UNIVERSIDADE DO VALE DO TAQUARI
CURSO DE ENGENHARIA DA COMPUTAÇÃO

CONTROLE DE ACESSO A EVENTOS UTILIZANDO RECONHECIMENTO FACIAL

João Pedro Basso Audibert

Lajeado, novembro de 2023

João Pedro Basso Audibert

CONTROLE DE ACESSO A EVENTOS UTILIZANDO RECONHECIMENTO FACIAL

Monografia apresentada na disciplina de Trabalho de Conclusão de Curso II, do curso de Engenharia da Computação, da Universidade do Vale do Taquari - Univates, como parte da exigência para a obtenção do título de bacharel em Engenharia da Computação.

Orientadora(o): Prof. Me. Fabrício Pretto.

Lajeado, novembro de 2023

João Pedro Basso Audibert

CONTROLE DE ACESSO A EVENTOS UTILIZANDO RECONHECIMENTO FACIAL

A Banca examinadora abaixo aprova a Monografia apresentada no componente curricular Trabalho de Conclusão de Curso II, do Curso de Engenharia de Computação, da Universidade do Vale do Taquari - Univates, como parte da exigência para a obtenção do título de Bacharel em Engenharia de Computação:

Prof. Me. Fabrício Pretto – orientador
Universidade do Vale do Taquari

Prof. Dr. Mouriac Halen Diemer
Universidade do Vale do Taquari

Prof. Dr. Alexandre Stürmer Wolf
Universidade do Vale do Taquari

Lajeado/RS, 13 de dezembro de 2023

RESUMO

O objetivo deste trabalho foi desenvolver uma aplicação de reconhecimento facial para dispositivos móveis para controle de acesso e reconhecimento automatizado de pessoas. A justificativa do desenvolvimento do sistema se deve à comum existência de filas em eventos sociais. Soma-se isso, a possibilidade de falsificação de documentos e a perda dos métodos garantidores de acesso. A autenticação é a base para o desenvolvimento da pesquisa e é feita por biometria, neste caso identificação facial. O estudo, conduzido de forma exploratória, se direciona à procura de técnicas para a implementação de uma ferramenta acessível e de código aberto. Focou-se na área de reconhecimento facial para controle de acesso. Foi criada uma solução utilizando as linguagens de programação Python, para o processamento, a qual apoia-se na biblioteca *face_recognition* para realizar o reconhecimento. Para o *mobile*, foi utilizado React Native com auxílio do *framework* Expo para detecção de faces e estruturação do aplicativo. Com o desenvolvimento completo, realizaram-se testagens piloto e de campo, as quais apontaram as falhas iniciais e posteriormente, corrigidas, os resultados. Esses, por sua vez, são muito satisfatórios, o algoritmo reconheceu corretamente cerca de 90% das vezes e em tempo hábil, de aproximadamente cinco segundos.

Palavras chave: Autenticação; mobile; eventos.

ABSTRACT

The objective of this work was to develop a facial recognition application for mobile devices for access control and automated people recognition. The justification for the development of the system is due to the common existence of queues at social events. Additionally, there is the possibility of document forgery and the loss of access assurance methods. Authentication is the foundation for the research development and is done through biometrics, in this case, facial identification. The study, conducted in an exploratory manner, focuses on finding techniques for the implementation of an accessible and open-source tool. It focused on the area of facial recognition for access control. A solution was created using the Python programming language for processing, which relies on the face_recognition library for recognition. For the mobile aspect, React Native was used with the support of the Expo framework for face detection and application structuring. With the complete development, pilot and field testing were conducted, which identified initial flaws that were later corrected, yielding very satisfactory results. The algorithm correctly recognized faces about 90% of the time and in a timely manner, approximately five seconds.

Keywords: Authentication; mobile; events.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE ABREVIATURA E SIGLAS

API	Application programming interface
APK	Android Application Package
CDN	Content Delivery Network
CSS	Cascading Style Sheet
CSV	Comma-Separated Values
EER	Equal Error Rate
FAR	False Acceptance Rate
FERET	Face recognition technology
FRGC	The Face Recognition Grand Challenge
FRR	False Rejection Rate
HOG	Histogram of Oriented Gradients
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JS	JavaScript
LGPD	Lei Geral de Proteção de Dados
NPM	Node Package Manager
PCA	Principal Component Analysis
PNG	Portable Network Graphics
REST	Representational State Transfer
RFID	Radio Frequency Identification
RGB	Red Green Blue
SDK	Software Development Kit
UUID	Unique Universal Identifier
VM	Virtual Machine

SUMÁRIO

1 INTRODUÇÃO

O reconhecimento facial sempre foi um processo importante para os indivíduos que vivem em sociedade. Para o ser humano, saber com quem se fala é parte central da comunicação, isso influencia no modo, vocabulário e até no tom de voz utilizado.

No âmbito dos sistemas de computadores, o processo de reconhecimento é feito através de autenticação. Magalhães e Santos (2003) apontam que a autenticação é a tarefa de permitir ou negar determinada ação ou acesso a alguém. Já a identificação é o processo de definir quem está acessando, e por conseguinte, atribuir os níveis de acesso que o indivíduo possui.

A tarefa de validação de identidade pode ser executada de diversas maneiras, sendo elas, senhas, crachás, ingressos e biometria. Este último tópico também possui diversos métodos, dentre eles impressões digitais, íris, formato de mão e reconhecimento facial. Costa, Obelheiro, Fraga (2006) definem biometria como um traço pessoal individual, robusto e difícil de ser imitado.

O uso da técnica de reconhecimento facial, foco deste trabalho, é observado em *smartphones*, como o FaceID da Apple, Mastercard Selfie Pay, e até para confirmação de compras em países asiáticos (ADJABI *et al.* 2020). Verifica-se que a técnica está sempre atrelada a empresas ou entidades governamentais.

Este trabalho toma como definição de evento a reunião de pessoas com o intuito de comparecer a determinado local que disponibilizará alguma atração de interesse desse grupo. Esses podem ser festas, palestras, congressos, dentre outros. Sendo eles organizados por uma instituição, produtora ou empresa o que influencia na quantidade de recursos disponíveis.

Algumas empresas de grande porte como Amazon e Microsoft dispõem de tecnologias de reconhecimento facial, respectivamente, Amazon Rekognition e Azure

AI Face Service. De fato, esses algoritmos trazem grandes bases de treinamento, e suporte a operação, que permitem bons resultados. No entanto, aponta-se que essas ferramentas fazem parte de um plano pago, o que, a depender da escala do evento, está fora de cogitação.

O acesso a eventos geralmente é uma tarefa custosa, que pode ocorrer em situações adversas como chuva e frio. O processo possui demora, haja vista que as pessoas devem possuir a forma de acesso, ingressos, crachás, além de uma identificação válida em território nacional. A obtenção desses documentos no momento citado, gera o problema, devido à busca em cima da hora dentro de carteiras e bolsas. Adiciona-se a essa problemática, a possibilidade de falsificação de documentos, prática comumente observada em festas de público jovem.

Adjabi *et al.* (2020) explica que o processo de identificação biométrica por reconhecimento facial é o de maior praticidade. Isso é evidenciado pela baixa ou nenhuma necessidade de cooperação que o método requer. Rapidamente a face do indivíduo pode ser capturada e enviada para processamento.

Levando em conta o exposto, uma aplicação capaz de identificar pessoas em eventos públicos e privados é de grande importância. A tecnologia ajuda a evitar contratempos, seja de procura, falsificação ou esquecimento de documentos ou forma de acesso. Além disso, fornece uma identificação visual para caso algo não funcione como planejado.

1.1 Problema de pesquisa e/ou hipótese

A implementação de um sistema de reconhecimento facial apresenta diversas etapas e desafios. O primeiro se encontra em como fazer a detecção de uma maneira eficiente e assertiva. Em sequência, o envio dos dados para processamento pelo servidor, e, por último, a importância do correto reconhecimento realizado pelas plataformas.

No mercado de tecnologia já existem soluções propostas para o problema a ser resolvido, no entanto, estão disponíveis apenas em serviços de *cloud*, que requerem grandes investimentos por parte dos clientes finais ou são de código fechado, impossibilitando as publicações sem uso de marca.

Os *smartphones*, estão difundidos em toda a sociedade, além de a cada dia que passa possuírem mais capacidade de processamento e comunicação. Soma-se a isso, a necessidade de reconhecimento para plataformas gerais.

O acesso a eventos também está no grupo das adversidades. O processo geralmente é custoso, deve-se apresentar documentação, forma de acesso (ingressos, crachás), o que acaba gerando filas e adicionando pessoal para conferência da validade dos elementos de identificação.

A partir dos princípios citados, surgem as seguintes questões: qual a assertividade de uma solução de reconhecimento facial para plataforma móvel, com equipamentos de reuso? Além disso, como a performance do reconhecedor se comporta em ambiente de produção?

1.2. Objetivos

Objetivou-se, de maneira geral, automatizar e simplificar o processo de entrada em eventos e com custo acessível. Com esse fim, realizou-se o desenvolvimento de uma solução de reconhecimento facial para dispositivos móveis com equipamentos de reuso que controlasse o acesso. Soma-se a isso, a implementação desacoplada entre as frentes de processamento e *mobile*. Visa-se isso com o intuito de possibilitar a adição de outras plataformas de maneira facilitada.

De maneira específica, buscou-se o entendimento dos métodos que a ferramenta de reconhecimento selecionada opera. Pretende-se também, trazer uma solução de custo reduzido e que aproveite material descartado ou obsoleto.

1.3 Estrutura do trabalho

O presente trabalho está disposto em seis capítulos. Esses, de forma resumida, serão responsáveis por introduzir, definir a base teórica, apresentar publicações da comunidade acadêmica, descrever as tecnologias e a forma que o projeto foi desenvolvido, apresentar as testagens realizadas e, por fim, pontuar as conclusões observadas.

Dessa forma, o primeiro capítulo, introdução, apresenta a contextualização do material. Tem a função de apresentar a relevância da pesquisa bem como suas justificativas e motivações.

O segundo capítulo, fundamentação teórica, descreve a base de conhecimento necessária para entendimento da pesquisa. Apresenta conceitos de autenticação, biometria e reconhecimento facial, descrevendo detalhadamente, sua importância, processo, técnicas, aplicações e impasses que a prática gera.

Já o terceiro capítulo, trabalhos relacionados, tem a finalidade de apresentar trabalhos acadêmicos ao leitor. Ajuda a complementar a presente monografia, pois traz publicações científicas para conhecimento, as quais auxiliaram na definição da arquitetura e método de implementação.

O quarto, materiais e métodos, descreve a forma que a pesquisa foi feita. Além disso, estende-se a descrever as tecnologias utilizadas conforme publicações científicas e suas próprias documentações. Apresenta, também, o desenvolvimento do projeto, como foi configurado e implementado.

O quinto capítulo demonstra as testagens realizadas. Divide-se em três partes, responsáveis por relatar o processo, as falhas observadas e seus processos de correção, além dos resultados obtidos e a discussão desses.

Por fim, apresenta as considerações finais, capítulo no qual são listadas as observações obtidas com a experimentação. Neste capítulo também são expostas melhorias observadas que visam a evolução do projeto.

2 FUNDAMENTAÇÃO TEÓRICA

Este trabalho apoia-se nos conceitos de autenticação, biometria, detecção e reconhecimento facial e os processos relacionados. O entendimento desses conceitos é fundamental para a compreensão do trabalho e dos motivos que levaram ele a se estruturar de tal maneira. Dessa forma, este capítulo tem a função de apresentar e explicar as bases teóricas abordadas no decorrer desta produção, por meio de materiais científicos.

2.1 Autenticação

Conforme descrito por Makhsud (2021), autenticação é o processo em que a autenticidade de determinado sistema, processo ou pessoa é validada. Isso ocorre com a parte examinadora solicitando para a parte auditada uma informação única, que a diferencie das demais. Essa informação pode ser uma senha, certificado, crachá, biometria, entre outros.

O processo de autenticação é dividido em três etapas. Inicialmente é realizada a identificação. Consiste no ato de se informar algo para que posteriormente seja autenticado. Para tal, tem-se o exemplo de login e senha (IDRUS et al., 2013).

O seguinte passo, conforme Kumar e Farik (2016), é a autenticação. Maneira com a qual o computador confirma quem está acessando determinado sistema ou local. Ela pode se basear em três formas: algo que aquele quem acessa lembra (senhas), algo que tenha (chip com RFID) ou algo que seja (biometria).

A autenticação é de alta importância para o funcionamento das aplicações, pois garante que o acesso aos dados e funcionalidades seja permitido apenas a agentes validados. Visto isso, no Brasil, foi desenvolvida a Lei Geral de Proteção de Dados.

Conforme determinado pela Emenda Constitucional Nº 115, de 2022, atrelada à LGPD, são assegurados os dados pessoais, estendendo-se para os meios digitais. Para tal, os acessos devem passar por um sistema de autenticação e nível de acesso, a fim de evitar o vazamento de informações pessoais sensíveis (BRASIL, 2022).

Por fim, com a identidade confirmada, é realizada a autorização. Segundo Idrus et al. (2013), na etapa da autorização, o usuário é liberado para usufruir do que estava tentando acessar. Além disso, são definidos os limites ou permissões as quais deverão ser respeitados, como setores privados, permissões de execução de aplicativos, entre outros, determinados como níveis de acesso.

2.2 Biometria

De acordo com Magalhães e Santos (2003), a biometria é definida pela identificação de determinado usuário por meio de suas características físicas. A ação é baseada nas características que o indivíduo possui, sendo as mais comuns: rosto, voz, íris e impressões digitais.

O uso da biometria nos sistemas computacionais é amparado pelos três pilares que eles se apoiam: confidencialidade, integridade e disponibilidade. A prática é uma boa alternativa para os processos de autenticação, pois é capaz de identificar um usuário com suas próprias características físicas. Essas, por sua vez, devem ser únicas, difíceis de compartilhar e de serem alteradas (COSTA; OBELHEIRO; FRAGA, 2006).

Dessa forma, Babich (2012), reitera as constatações feitas anteriormente. Salienta em sua pesquisa os seguintes termos, sumarizando a ideia:

- Universalidade - todos humanos devem possuir;
- Unicidade - capazes de diferenciar duas pessoas;
- Constância - se manterem mesmo com alterações naturais do organismo;
- Coletabilidade - devem ser fáceis de coletar em um curto espaço de tempo;
- Performance - a diferenciação deve ser possível de ser identificada em tempo hábil;
- Segurança - devem ser difíceis de se copiar.

Costa, Obelheiro, Fraga (2006) salientam que existem dois processos para autenticação, verificação e identificação. Ambos utilizam o método de amostra e referência, o qual recebe uma informação e dirige-se a algum local para buscar uma

correspondência. O primeiro processo é utilizado em validações que utilizam senhas ou *tokens*, já o segundo é a base da autenticação biométrica.

O processo de identificação biométrica ocorre da seguinte forma: o usuário fornece suas informações, neste caso, suas características físicas e compete ao sistema fazer a conferência. Este processo é uma busca de um para muitos, o qual não consegue trazer com exatidão total quem é a pessoa, seu resultado é sempre uma aproximação. Com a ampliação do poder computacional e da qualidade dos algoritmos, a confiança dos resultados chega próximo dos 100% (COSTA; OBELHEIRO; FRAGA, 2006).

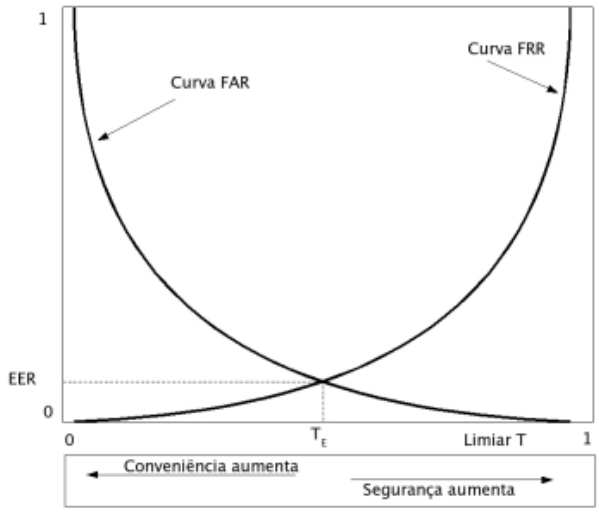
2.2.1 Tolerância

Malik et al. (2014) define que as possíveis combinações são analisadas para a determinação da autenticidade do usuário, definindo o limiar da aplicação. Em suma, é feita uma comparação da semelhança encontrada entre a amostra e a referência, a qual deve satisfazer um valor mínimo para assegurar a identidade do indivíduo.

Desse processo, surgem as siglas FAR (False Acceptance Rate) e FRR (False Rejection Rate), as quais são, respectivamente, a quantidade de vezes que uma pessoa não autorizada é aceita; e a frequência que uma pessoa legítima é barrada (MALIK et al., 2014).

As incidências dos erros de falsa aceitação e falsa rejeição são opostos entre si. Esses índices são resultados da configuração do sistema e operam entre a relação de conveniência e segurança. Sistemas convenientes são mais fáceis de utilizar e disseminar, enquanto sistemas seguros podem barrar usuários autorizados até intervenção futura. Essa situação é ilustrada na figura 1, a qual demonstra o EER (Equal Error Rate), taxa em que os erros de aceitação e rejeição se encontram, demonstrando o sistema ideal de autenticação (COSTA; OBELHEIRO; FRAGA, 2006).

Figura 1 - Curvas das taxas de erro FAR e FRR demonstrando a tendência entre conveniência e segurança.



Fonte: Costa, Obelheiro, Fraga (2006).

2.3 Reconhecimento facial

Reconhecimento facial, segundo Tolba, El-Baz, El-Harby (2006), é uma abordagem biométrica a qual utiliza sistemas automatizados para validar a identidade de determinada pessoa baseado em suas características faciais. Dos métodos de identificação por meio de atributos humanos, é o menos invasivo, a pessoa não precisa realizar muitas ações ou fornecer diversos dados para poder ser reconhecida.

O método biométrico apresenta três vantagens principais sobre os demais, como citado anteriormente, não é invasivo, portanto, não é necessário tocar um sensor, como na coleta de impressões digitais. Além disso, não carece de grande cooperação do usuário, ou seja, a captura do material de análise é rápida. Por fim, observa-se a naturalidade, pois o cérebro humano também apoia-se nessa técnica para determinar quem é a pessoa analisada (ADJABI et al., 2020).

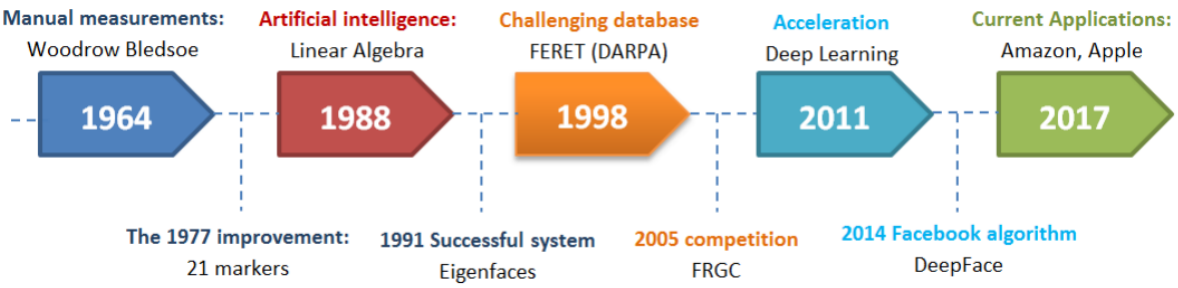
Sua utilização ocorre em diversas áreas, como controle de acesso, identificação de assaltantes, monitoramento, além de ser ponto central na comunicação humana. A prática iniciou no século XIX, a qual consistia na captura das imagens de perfil de um grupo de pessoas e classificando as curvas observadas em grupo. Esse processo passou para situações reais e seu desenvolvimento foi acelerado, haja visto os avanços computacionais e a existência de grandes bases para treinamento (TOLBA; EL-BAZ; EL-HARBY, 2006).

No âmbito computacional, segundo Adjabi et al. (2020), os estudos de reconhecimento facial iniciaram em 1964 com Woodrow Bledsoe e sua equipe, os quais imaginaram um sistema semi automático que realizava o reconhecimento através de vinte medidas da face do usuário, informadas manualmente. No ano de 1977 foram adicionadas outras vinte e uma outras marcações para análise.

Devido à complexidade do processamento, Adjabi et al. (2020), destaca que, em 1988, foi adicionado o auxílio da inteligência artificial. Ela, por meio de álgebra linear, simplificava a interpretação das marcações identificadas nos rostos analisados. O primeiro algoritmo de sucesso, o Eigenfaces, foi publicado em 1991, o qual observava os principais pontos do rosto do indivíduo. Já em 1998, foi apresentado o FERET (Face recognition technology), por parte do DARPA (Defense Advanced Research Projects Agency). Tornando-se a primeira base de dados sólida para treinamentos de sistemas de reconhecimento facial.

Ainda conforme Adjabi et al. (2020), entrando no século XXI, em 2005 foi realizado um grande evento do tema, o FRGC (The Face Recognition Grand Challenge) com o intuito de evoluir rapidamente o assunto. Isso foi observado ao entrar nos anos de 2010, quando estratégias de deep learning foram adicionadas ao processo. Isso permitiu o desenvolvimento de algoritmos mais precisos, haja visto o DeepFace do Facebook, de 2014. De 2017 em diante, tem-se observado a adesão da tecnologia em diversos processos, tais como o Selfie Pay da Mastercard, a checagem de faces na realização de compras na China e o FaceID, da Apple. A figura 2 apresenta a sumarização da linha do tempo apresentada.

Figura 2 - Linha do tempo do desenvolvimento de algoritmos de reconhecimento facial.

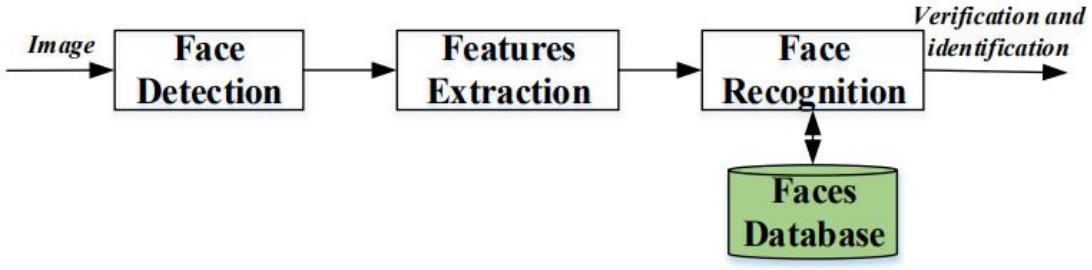


Fonte: Adjabi et al. (2020).

2.4 Processo de reconhecimento facial

Os processos de reconhecimento facial geralmente convergem para três passos essenciais que podem ser subdivididos posteriormente. São eles detecção de faces, extração de características e reconhecimento de fato. A figura 3 demonstra o processo macro de identificação por meio da face (KORTLI *et al.*, 2020).

Figura 3 - Processo simplificado de reconhecimento facial com os passos.



Fonte: Kortli *et al.* (2020).

Woodward Junior *et al.* (2003), descreve que o passo de detecção facial inicia pela obtenção de alguma imagem para análise, seja via carregamento ou captura em vídeo por alguma câmera. Em seguida, um algoritmo deverá executar a busca de faces no retrato. Esse procedimento busca padrões como uma forma oval contendo algo semelhante a dois olhos e uma boca. Beham, Roomi (2013), salientam que juntamente ocorre a vetorização da imagem, processo crítico para o sucesso do sistema.

O segundo passo do processo, extração de características, conforme descrito por Kortli *et al.* (2020), consiste na identificação das medidas presentes na face do usuário analisado. Neste passo, busca-se identificar a geometria facial, a qual será ponto central para o último passo. Nessa etapa, existem diversas técnicas que podem ser empregadas, tais como Holística, Cascata e Eigenface.

O último passo é o reconhecimento facial propriamente dito. Woodward Junior *et al.* (2003) aponta que nesta etapa, os vetores obtidos anteriormente são comparados com bases já analisadas. O resultado obtido determina se há ou não combinações. A assertividade mínima do sistema é determinada pela sua configuração e se baseia na aplicação deste.

Beham, Roomi (2013) assinalam a utilização de inteligência artificial durante o processo. Dessa forma, são adicionados os algoritmos de inteligência artificial, treinados para reconhecimento e classificação de padrões, seja na extração de características ou no reconhecimento. Essa estratégia melhora a assertividade final bem como acelera o processo de identificação, haja visto a qualidade empregada na obtenção dos pontos de interesse para análise.

2.5 Técnicas de reconhecimento facial

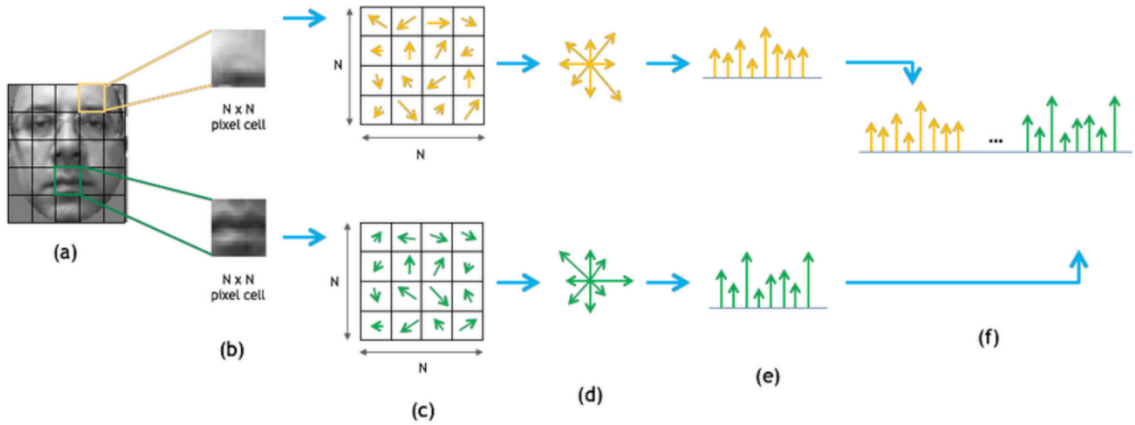
O processo de identificação biométrica de uma pessoa por meio da face pode ser executado de diversas maneiras. Nesta seção serão apresentadas algumas técnicas utilizadas no mercado para reconhecimento facial.

2.5.1 HOG (*histogram of oriented gradients*)

Segundo Eng *et al.* (2019), HOG ou em portugês, histograma de gradientes orientados, é um algoritmo de visão computacional direcionado para a detecção de pessoas, objetos, entre outros. Trabalha, em termos amplos, reconhecendo as magnitudes, intensidade de iluminação, e orientação de imagens.

Primeiramente o software, ainda conforme Eng *et al.* (2019), verifica a orientação da figura. Posteriormente, ocorre a divisão do retrato em partes iguais, chamadas células de N x N *pixels*. Desses componentes, são extraídos, quantificados e acumulados os vetores de magnitude, para qual lado a luz está apontando, em um histograma. Salientam que o processamento geralmente ocorre em imagens em preto e branco. Isso se deve ao fato de que as cores não fazem diferença no processo, baseado apenas em luz e sombra, além de adicionar mais *bytes* para serem transitados. A figura 4 ilustra a extração de características por meio da técnica.

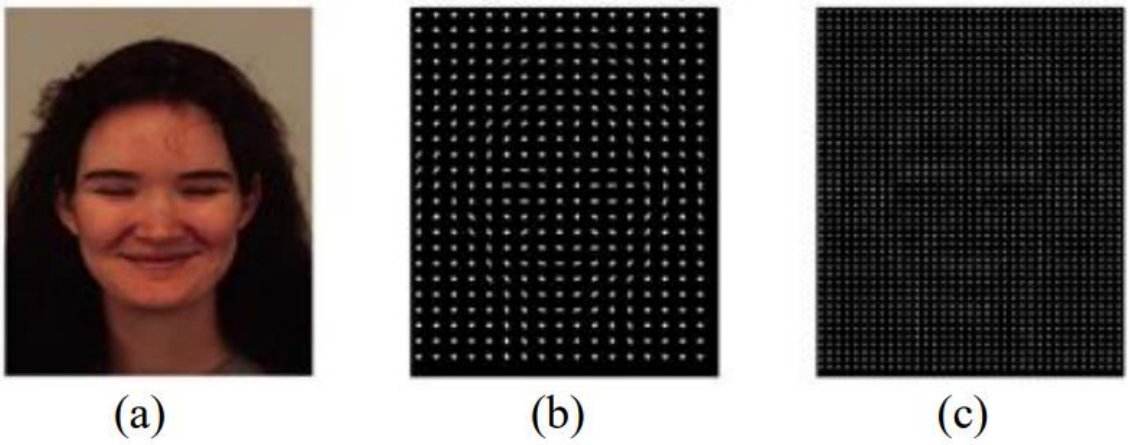
Figura 4 - Processo de extração de características por meio de histograma de gradientes orientados.



Fonte: Eng et al. (2019).

Eng *et al.* (2019) ainda explica que os principais parâmetros da prática são a quantidade de vetores e o tamanho das células. A figura 5, trazida pelos autores, exemplifica a granularidade aplicada, sendo “a” a imagem analisada, “b” uma análise executada com células de 32 x 32 *pixels* e, por fim, “c” uma análise de 16 x 16 *pixels*, mais precisa. Ela demonstra que a quantidade de detalhes extraídos aumenta conforme o tamanho das partes é diminuído. No entanto, a maior extração de características e quantidade de vetores implica na maior geração de dados, acarretando em mais tempo de processamento. Dessa forma, cabe ao desenvolvedor configurar, conforme o nível de precisão desejado, o tamanho dos objetos de análise.

Figura 5 - Demonstração da parametrização do tamanho das células e o resultado gerado.



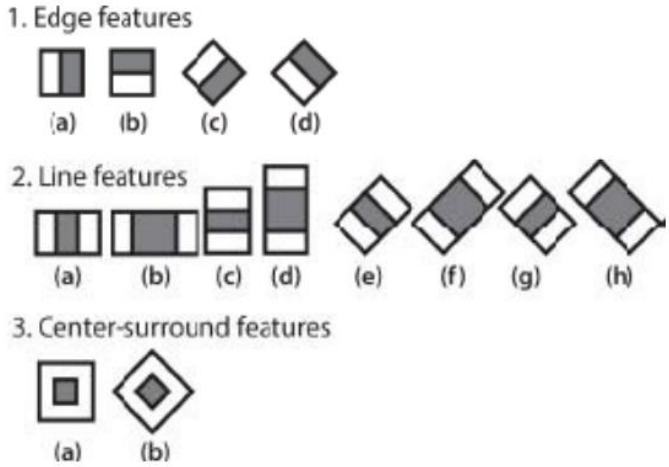
Fonte: Eng et al. (2019).

2.5.2 Haar cascade classifier

As funções de Haar foram introduzidas pelo matemático Alfred Haar. Elas se apoiam no conceito de claro e escuro, observando a variação de luminosidade entre eles (GOYANI; PATEL, 2018). Para observação dos contrastes, foi proposto o LBP (*Local Binary Pattern*) o qual descrevia as oposições entre zeros e uns, sendo zero branco e um preto (CUIMEI *et al.*, 2017).

A figura 6 apresenta o padrão descrito por Haar. Ela demonstra a segmentação da imagem analisada, são evidenciadas as linhas, bordas e pontos, os quais determinarão os limites do objeto.

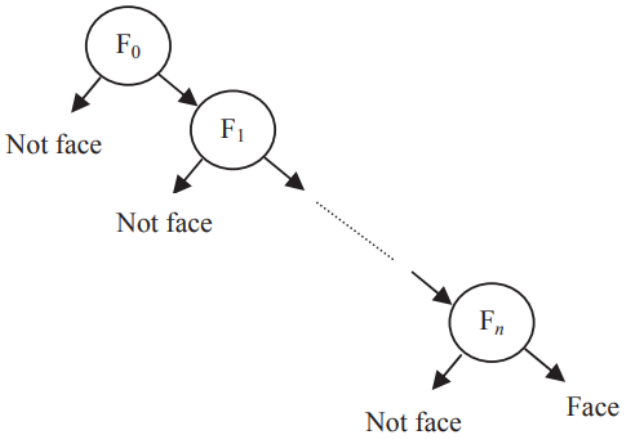
Figura 6 - Demonstração do padrão Haar.



Fonte: Cuimei *et al.* (2017).

Viola e Jones criaram o conceito de comparação em cascata. Conforme Cuimei *et al.* (2017), o método se baseia na classificação eliminatória de seções da imagem que podem ou não compor um rosto. Inicia no objeto de análise zero e segue até o *n*, determinado pela configuração do sistema. A figura 7 expõe o conceito de eliminação executado pelo algoritmo, o qual segue por uma árvore de decisão entre face e não face.

Figura 7 - Demonstração do funcionamento do algoritmo Viola-Jones de detecção facial.



Fonte: Cuimei *et al.* (2017).

Cuimei *et al.* (2017) aponta também que, apesar da grande taxa de detecção que a técnica apresenta, cerca de 99,9%, são observados alguns falsos positivos. Isso pode ser resolvido com o aumento do treinamento ou da quantidade de nodos possíveis. Dessa maneira, a avaliação acaba tomando mais tempo, mas a assertividade aproxima-se de 98%.

2.5.3 Eigenfaces

O algoritmo de Eigenfaces, conforme descrito por Hapsari, Mutiara, Tarigan (2019), tem a função de identificar faces humanas através de outras imagens previamente existentes e com a detecção já testada. Ou seja, neste modelo, existem imagens de referência que determinam o que é um rosto.

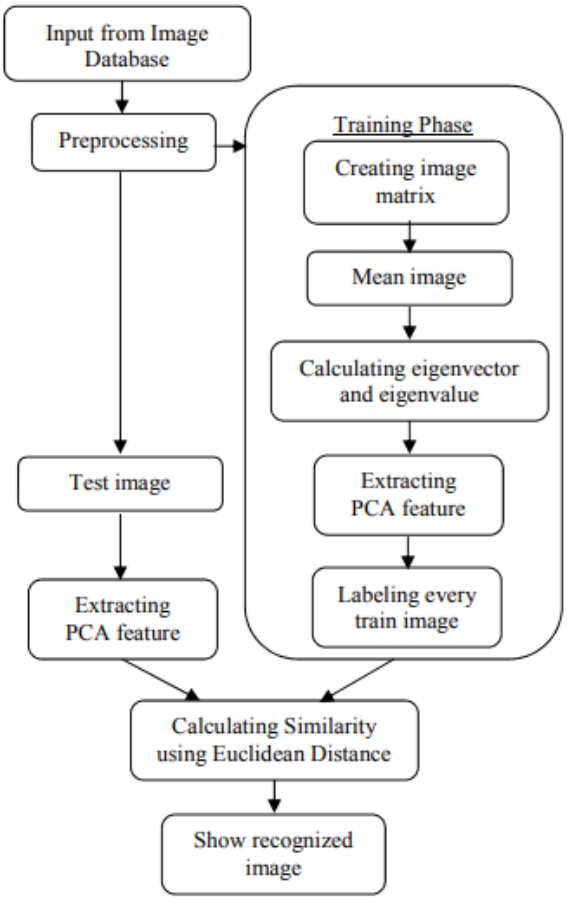
Conforme Mulyono et al. (2019), segue o modelo de reconhecimento por componentes de análise principais, PCA (*principal component analysis*). O autor também reforça a ideia do parágrafo anterior, que o algoritmo é responsável apenas por fazer a detecção de rostos em uma imagem apresentada.

Mulyono et al. (2019) aponta que Eigenfaces é reconhecido por ter uma acurácia elevada, até mesmo em situações não ideais. Devido a isso, tem ampla utilização nas áreas de estudo relativas a expressões faciais, detecção de rostos e reconstrução de faces em imagens.

O processamento, ilustrado na figura 8, inicia com a entrada de uma imagem para análise e o pré-processamento, o qual converte o retrato de RGB (*Red Green Blue*) para escala de cinza, pois reduz a complexidade para um terço. A imagem é validada e tem as características extraídas. A similaridade é determinada pelo algoritmo de distância euclidiana e, por fim, a imagem reconhecida ou grau de semelhança são mostrados (MULYONO et al. 2019).

Caso seja a referência, ela passa pela etapa de treinamento. Essa parte inicia obtendo os vetores da imagem, após isso, é obtido a média e essa é subtraída do vetor original. Do resultado, são calculados os valores de *eigenvector* e *eigenvalue*. Esses números são utilizados na extração de características, a correspondência entre valor e vetor seja maior que um, o componente será utilizado para constituição do *eigenface*. Por fim, o treinamento encerra adicionando os rótulos para identificação (MULYONO et al. 2019).

Figura 8 - Demonstração do funcionamento de um algoritmo de reconhecimento facial utilizando Eigenfaces.



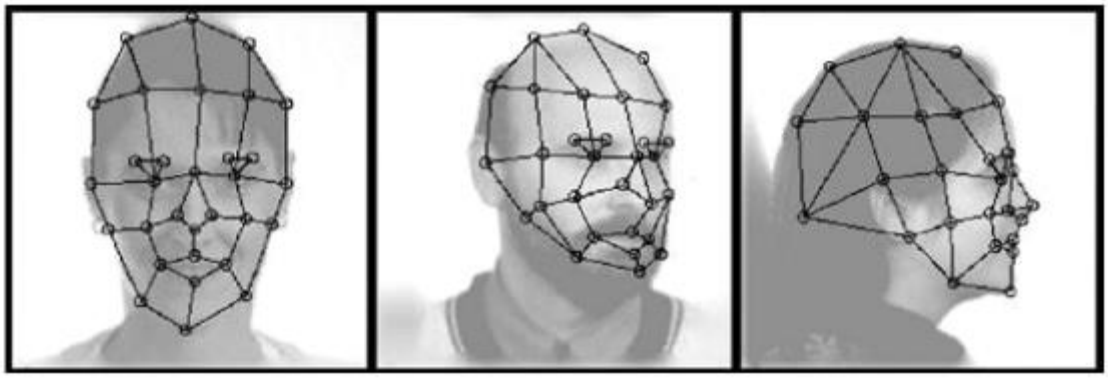
Fonte: Mulyono et al. (2019).

2.5.4 *Feature-Based*

Essa abordagem de reconhecimento facial, conforme Beham, Roomi (2013), se difere das baseadas em gradientes, análise de pontos ou vetores. Ela opera fazendo uso de características concretas do indivíduo, tais como olhos, nariz e boca.

Beham, Roomi (2013) apontam o experimento *Elastic Bunch Graph Matching* como um exemplo do uso da técnica. Nele, os pontos de interesse, ou seja, elementos que identificam o usuário, são posicionados em nodos que são interconectados. A figura 9 demonstra a estruturação dos dados coletados.

Figura 9 - Grafos gerados pelo experimento Elastic Bunch Graph Matching.



Fonte: Beham, Roomi (2013).

Os autores, Beham, Roomi (2013), ressaltam a eficiência no reconhecimento por parte dos algoritmos que utilizam as características como ponto chave. Os sistemas apresentam bons resultados, identificação e performance, até mesmo com pouco treinamento.

2.6 Redes neurais convolucionais

As redes neurais convolucionais são peças fundamentais no desenvolvimento de aplicações de visão computacional. A partir dela, permitiu-se a criação de diversos algoritmos, dentre os mais conhecidos estão reconhecimento facial e veículos autônomos (LI *et al.*, 2021).

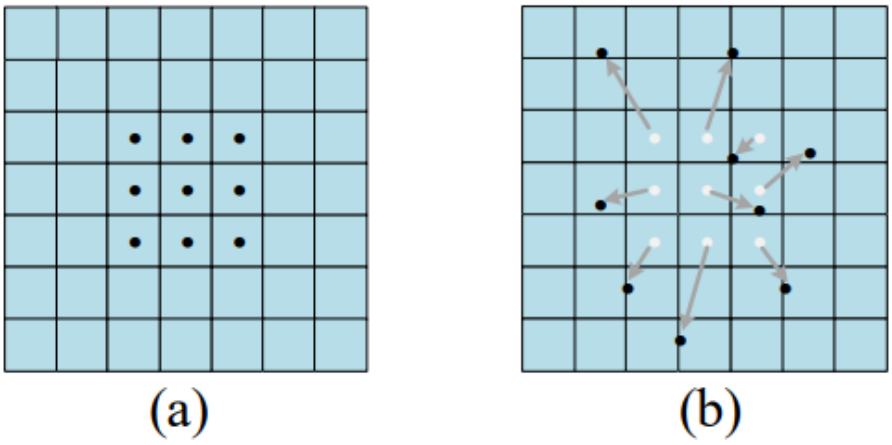
Li *et al.* (2021) ainda definem que, seu funcionamento se difere das demais, pois seu processo de extração de características se apoia na percepção visual. Opera

com a lógica de *kernels*, receptores que podem receber diversos tipos de informações, os quais se conectam apenas com uma certa quantidade de neurônios anteriores, a fim de reduzir os parâmetros e acelerar o processo de convergência. A tecnologia, visando a obter resultados mais rápidos, ainda pode realizar a distribuição de carga entre suas células.

Ainda segundo o mesmo autor, essas redes neurais, aplicam o conceito de diminuição do tamanho da amostra por dimensionamento. Para tal, são adicionados zeros entre os pontos de interesse. Essa prática permite a diminuição dos parâmetros, mas segue mantendo a quantidade de informações. Por terem seu foco na visão computacional, permitem também, adaptar a disposição dos pontos de maneira irregular.

A figura 10, elaborada por Li *et al.* (2021), demonstra o funcionamento da redução de amostras por dimensionamento operando de maneira deformada. O item “a” corresponde ao comportamento padrão o qual possui uma grande densidade na sua composição. Já o item “b” apresenta a dispersão dos pontos, com o intuito de cobrir todo o quadrante respeitando as irregularidades apresentadas em uma imagem de mundo real.

Figura 10 - Deformação dos pontos de interesse em redes neurais convolucionais



Fonte: Li et al. (2021).

2.7 Aplicações de reconhecimento facial

O reconhecimento facial, de fato, possui diversas aplicações. Woodward Junior *et al.* (2003) destacam, além dos sistemas de autenticação, a busca por pessoas desaparecidas e identificação de criminosos ou terroristas.

Seguindo com Woodward Junior *et al.* (2003), essa característica se deve pela facilidade de operação de um sistema de identificação facial. Ele não requer colaboração do usuário, ou seja, a captura das imagens pode ser feita de forma rápida e até mesmo sem consentimento. Isso implica em uma grande possibilidade de aumento da base de treinamento.

A aplicação no setor da investigação criminal é de extrema importância. O processo é muito semelhante ao de autenticação, primeiro a imagem é obtida e a face detectada, o rosto é enviado para uma central para normalização e processamento e por fim, haja visto a criticidade da situação, oficiais treinados determinam se a correspondência é legítima e enviam, ou não, o alerta para as autoridades em patrulha (WOODWARD JUNIOR *et al.*, 2003).

Galterio, Shavit, Hayajneh (2018) apresentam o uso no setor comercial, em que seu foco está em identificar clientes com grande potencial de compra. Para tal, os autores sugerem a construção de um sistema que identifica e atribui as compras realizadas pelo indivíduo à sua face, identificada através do sistema de vigilância. Dessa forma, direcionando atendentes com melhor histórico, a fim de garantir maior sucesso nas vendas.

Adjabi et al. (2020) aponta a criação do produto Selfie Pay, o qual verifica a identidade do usuário ao fazer uma compra no cartão de crédito. Além disso, Apple estendeu seu sistema de reconhecimento para autenticação em sistemas bancários e de revenda.

Galterio, Shavit, Hayajneh (2018) descrevem também, a existência de sistemas de segurança dentro dos celulares Android. Citam *FaceLock* e *AppLock*. O primeiro, possui a função de autenticação do dispositivo como um todo, além da verificação de usuário em ações críticas. Já o segundo, tem a função de bloquear o acesso a aplicativos. Ambos realizam a comparação de imagens previamente capturadas pelo dispositivo para garantir autenticidade.

2.8 Problemas no reconhecimento facial

O reconhecimento facial, dentre todos os métodos de identificação biométrica, é o menos invasivo e aquele que exige o menor índice de colaboração do indivíduo. Dessa forma, referências para o sistema, ou seja, imagens de rostos, podem ser capturadas muito facilmente, até mesmo sem consentimento do usuário (TOLBA; EL-BAZ; EL-HARBY, 2006).

Woodward Junior *et al.* (2003), reforçam que para todo o sistema biométrico funcionar, ele deve possuir alguma referência para comparação posterior. Para tal, orienta que a coleta de bases para treinamento, em casos de órgão públicos ou de pesquisa, seja feito em áreas públicas para diminuir os problemas legais que a prática pode acarretar.

No entanto, conforme Conger, Fausset, Kovaleski, (2019), San Francisco baniu a tecnologia de reconhecimento facial em público. Com quase unanimidade, o quadro de supervisores da cidade votaram pelo banimento da tecnologia, alegando o abuso de imagem.

Além disso, é observado a limitação no uso de retratos de crianças. Conforme Barret (2020), crianças são vulneráveis e devem ter sua privacidade respeitada. O maior problema verificado é a insegurança dos sistemas que executam a função de reconhecimento, além de sua falta de precisão com pessoas desse grupo. A autora salienta que a exposição pode prejudicar o desenvolvimento do indivíduo, pois tende a limitar a livre expressão dos pequenos.

Além dos impasses legais, existem as dificuldades presentes para que o processo ocorra de forma correta. Os três pilares da tecnologia, detecção, extração de características e reconhecimento facial precisam de um ambiente minimamente ideal para eliminar os problemas descritos a seguir.

Segundo Anwarul e Dahiya (2019), apontam que os sistemas de identificação facial sofrem influência de fatores intrínsecos e extrínsecos. Os primeiros são relativos à face da pessoa, já os segundos, ao ambiente ou razões externas. No primeiro grupo, estão envelhecimento, expressão facial e cirurgias plásticas, todos capazes de alterar significativamente o rosto analisado. Já no segundo grupo, tem-se oclusão, baixa resolução, ruído, iluminação e variação de pose, como exemplos. Estes, por sua vez, excluem ou reduzem a capacidade das aplicações executarem suas funções.

3 TRABALHOS RELACIONADOS

Esta seção apresenta trabalhos já realizados pela comunidade científica, os quais orientaram e definiram os padrões e técnicas de desenvolvimento do presente trabalho. O intuito é trazer trabalhos recentes, dos últimos 10 anos, para além de ampliar os conhecimentos sobre o assunto, verificar e validar o que foi construído.

3.1 Real Time Automatic Attendance System for Face Recognition Using Face API and OpenCV

Conforme Khan, Akram, Usman (2020), o trabalho desenvolvido tem como objetivo validar a presença dos alunos em uma sala de aula. A problemática inicial se dá pela demora no uso dos métodos biométricos, da possibilidade de fraude no uso de itens de identificação (RFID) e da perda ou dano dos elementos físicos.

Para solucionar os problemas apresentados anteriormente, os desenvolvedores fizeram uso de reconhecimento facial por meio de uma câmera instalada na sala de aula e na biblioteca *OpenCV*. O processamento se divide em duas etapas, detecção de face e reconhecimento facial, sendo feito por *YOLO V3* (*You Only Look Once*) e *Azure Face API*.

A primeira ferramenta, YOLO V3, é um algoritmo em linguagem Python designado para a detecção de objetos e pode ser configurado para localizar um ou mais rostos em uma determinada imagem. Já a Azure Face API é responsável pela obtenção dos pontos de interesse e reconhecimento facial.

O sistema inicia capturando uma foto da sala de aula, inicia a detecção de faces e corta a imagem, evidenciando o rosto do estudante. Após isso, ele separa entre conhecidos ou não e gera a primeira planilha. No passo seguinte, é feito o reconhecimento facial pela API de faces da Azure. Depois disso, os alunos são

reclassificados entre conhecidos ou não, os dados de presença são realizados e registrados em planilhas e sistemas internos, os quais são enviados aos professores, pais e responsáveis.

3.2 Facial Recognition using the OpenCV Libraries of Python for the Pictures of Human Faces Wearing Face Masks during the COVID-19 Pandemic

Vadlapati, Velan e Varghese (2021) apresentam os possíveis meios de reconhecimento de pessoas. Sendo eles, uso de digital, leitura de íris, reconhecimento vocal e facial.

Além disso, concordam na existência de cenários não ideais para o funcionamento dos sistemas listados. A falta de iluminação ideal, tempo de processamento, elementos móveis na imagem e qualidade da captura, são algumas situações não ideais (KHAN; AKRAM; USMAN, 2020).

Recentemente, devido à pandemia de Covid-19, foi adicionada a essa lista, a presença de máscaras de proteção individual. Elas cobrem pontos importantes para o reconhecimento facial. O problema se deve ao treinamento dos sistemas, os quais dependem do rosto completo para detecção da face ou reconhecimento facial.

A pesquisa se desenvolve ao redor da biblioteca, desenvolvida em Python, nomeada de *face_recognition*, por meio do método *face_landmarks()*, inicia obtendo os pontos de referência das imagens previamente salvas. Posteriormente, o mesmo processo é obtido com a fotografia de testagem. Os pontos de interesse indicam as dimensões e posições da face e se estruturam como evidenciado na figura 11.

Figura 11 - Demonstração dos pontos obtidos por face_landmarks().

```
[{
    'chin': [(46, 47), (45, 54), (44, 62), (44, 69), (44,
    'left_eyebrow': [(51, 42), (54, 39), (58, 39), (63, 4
    'right_eyebrow': [(75, 44), (80, 44), (86, 44), (90,
    'nose_bridge': [(70, 48), (68, 52), (67, 56), (66, 6
    'nose_tip': [(60, 64), (62, 65), (65, 67), (68, 66),
    'left_eye': [(55, 47), (57, 45), (61, 46), (63, 48),
    'right_eye': [(77, 51), (80, 50), (84, 51), (86, 54),
    'top_lip': [(54, 75), (58, 72), (61, 72), (64, 73), (
    'bottom_lip': [(73, 80), (68, 81), (64, 81), (62, 80)
}]
```

Fonte: Vadlapati, Velan e Varghese (2021).

Essas marcações foram comparadas com o algoritmo treinado para fazer a verificação. Previamente, foram adicionadas imagens de Joe Biden e Elon Musk, sem máscara, para que as referências fossem criadas e salvas. Em um segundo momento, as imagens dos citados, desta vez com máscaras cirúrgicas, eram expostas ao programa, que foi capaz de identificá-los com boa precisão, entre 70% e 85%.

O sistema desenvolvido segue em evolução. Foram identificados diversos pontos de melhoria, tais quais: identificações de emoções em pessoas usando máscaras e performance. São identificados 128 pontos de cada rosto, os quais demoram cerca de 30 segundos para apresentar o resultado, mesmo em um escopo pequeno. Para tal, visa-se melhorar processos e técnicas para que o reconhecimento siga com a qualidade apresentada e melhore as lacunas listadas.

3.3 A Comprehensive Review on Face Recognition Methods and Factors Affecting Facial Recognition Accuracy

Os autores iniciam relatando a importância de um sistema de reconhecimento eficiente, sendo ele facial ou biométrico. Essa necessidade é evidenciada nos sistemas de fraude, vigilância, área forense, identificação de criminosos, etc. (ANWARUL; DAHIYA, 2019).

A pesquisa segue apresentando possíveis agentes que podem criar dificuldades para que os processos de detecção, extração de características e

reconhecimento facial ocorram, impactando na precisão dos resultados. Os fatores podem ser intrínsecos, ou seja, que incidem diretamente na face ou que influenciam na detecção dessa.

Como exemplos intrínsecos tem-se:

- Idade: o rosto humano varia inteiramente nesse processo. Formato, pintas, sardas e marcas de expressão são exemplos que alteram o processo de reconhecimento;
- Expressão facial: o treinamento ocorre, comumente, com uma face neutra. A alteração do posicionamento de certos elementos, podem gerar incerteza por parte do sistema;
- Cirurgias plásticas: os procedimentos estéticos têm a capacidade de alterar radicalmente atributos da face analisada. Devido a isso, é um artifício usado por criminosos para driblar sistemas de reconhecimento.

Já para os elementos extrínsecos, fala-se em:

- Oclusão: se define pela obstrução de certas características faciais. Pode ser causada por cabelo, barba, adereços,entre outros;
- Baixa resolução: imagens com baixa resolução dificultam a obtenção dos pontos de referência, utilizados para análise;
- Ruído: podem ser introduzidos via programas de manipulação de imagens, os quais também atrapalham na coleta das informações;
- Iluminação: esse elemento é um dos principais para a captura de uma imagem. Pode alterar a qualidade, saturação, contraste, favorecer elementos faciais e vice-versa. Uma iluminação constante é essencial para o processo de reconhecimento facial;
- Variação de pose: influencia diretamente nos pontos de referência que o algoritmo obtém durante o processo de detecção.

Conforme os autores, existem três tipos de sistemas de reconhecimento facial no mercado. São eles baseados em aparência, em características e híbridos. O primeiro grupo apresenta desempenho superior, no entanto uma confiança menor na entrega. O segundo supre as dificuldades do primeiro, no entanto, ocupa mais memória computacional, carece de uma base de dados maior e um aprendizado supervisionado mais extenso, acarretando em uma resposta mais lenta. O terceiro combina os ônus e bônus dos dois métodos anteriores.

Por fim, algoritmos de aparência não performam bem em imagens de baixa resolução e imagens com variação de pose. Já sistemas baseados em características possuem resultados mais satisfatórios, no entanto, precisam de aprendizado supervisionado para funcionar corretamente. Os autores encerram revalidando a necessidade da elaboração de um algoritmo de reconhecimento facial capaz de performar até mesmo em ambientes não ideais.

3.4 Face recognition accuracy of forensic examiners, super recognizers, and face recognition algorithms

O reconhecimento facial na ciência forense é de suma importância. Além de apresentar uma identificação quando necessário, pode acarretar em consequências muito sérias na vida de um determinado indivíduo. Para avaliar a acurácia das identificações, foram selecionados grupos de estudos compostos por examinadores e revisores forenses, super reconhecedores, analistas de impressões digitais e estudantes. Foram também adicionados ao grupo sistemas de reconhecimento facial elaborados com redes neurais convolucionais profundas, (PHILLIPS *et al.* 2018).

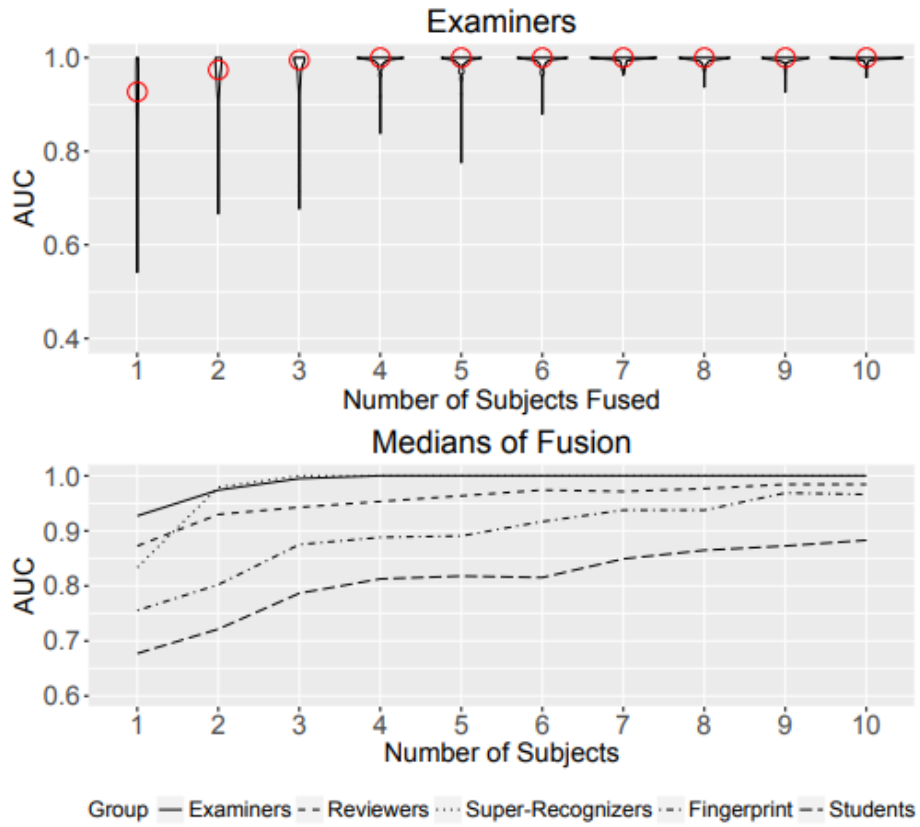
O estudo foi conduzido da seguinte forma: para cada integrante era apresentada uma imagem de uma pessoa que deveria ser comparada com a do lado. A escala de avaliação era de +3 até -3, sendo o número positivo responsável por indicar que era a mesma pessoa e o negativo o contrário.

Foram feitas duas baterias de testes, nas quais os grupos trabalhavam por conta, ou seja, os examinadores faziam suas análises na escala, bem como estudantes, super reconhecedores, sistemas de reconhecimento, etc.

Na sequência, a assertividade foi avaliada de 0 a 1. Nas análises humanas, o máximo foi de 0.93, por parte dos examinadores forenses e o mínimo, por parte dos estudantes, 0.68. Nos sistemas de reconhecimento facial, a evolução conforme os anos foi notada, a rede neural mais velha, A2015, apresentou acurácia de 0.68, já a A2017b, 0.96, superando os resultados humanos.

Após a verificação dos resultados individuais foram elaboradas análises combinadas entre 2 a 10 participantes, dessa maneira, os apontamentos trabalhavam com a mediana. Com essa estratégia, o grupo ideal, examinadores, passaram a atingir 100% de assertividade já na terceira fusão. Já os estudantes melhoraram seus resultados para 88% na última combinação.

Figura 12 - Resultado da mediana dos examinadores e demais grupos após combinação de análises.



Fonte: Phillips et al. 2018.

A combinação das análises humanas com as de máquina também obtiveram grande melhora, atingindo o máximo de acurácia na maioria dos casos. No entanto, o que foi evidenciado na combinação é que a possibilidade de afirmação ou negação foi aumentada com menor probabilidade de erro. Dessa maneira, as afirmações ficaram mais próximas de +3 ou -3, até mesmo para os cientistas forenses, os quais não realizaram afirmações plenas na maior parte do experimento. Serviu também para corrigir as estimativas dos estudantes.

O autor conclui que a análise humana pode ser variada conforme experiência e também validação. Na ciência forense a verificação e validação de resultados é uma prática comum. Dessa forma, a capacidade humana de reconhecimento facial pode ser aumentada se combinada com os esforços computacionais, bem como a precisão da análise.

3.5 Facial Detection and Recognition System Using Python

O autor inicia apresentando a diferença entre detecção e reconhecimento facial. O primeiro item refere-se a como o computador, por meio da tecnologia escolhida, vai fazer a separação da face identificada do resto da imagem. Já o segundo, é relacionado com o processo de recognição do indivíduo (RAO *et al.* 2022).

Para tal, os pesquisadores, visando trabalhar com tecnologias acessíveis e bem documentadas, escolheram Python e OpenCV. A linguagem de programação, Python, foi selecionada devido à facilidade apresentada enquanto sendo escrita e operada. Já o protocolo do OpenCV, se faz presente devido à grande utilização já apresentada.

O conjunto trabalha primeiramente separando a face identificada do resto da imagem, por meio do recorte de um retângulo. Importante ressaltar que a imagem deve estar em um ambiente ideal de luminosidade e a pessoa deve estar de frente para a câmera. Desse processo, são salvas imagens que servem para posterior comparação.

Após o processo de obtenção, explicado anteriormente, vem o de normalização. Nessa etapa o rosto é normalizado e a orientação é corrigida. Em sequência ocorre o reconhecimento, o qual baseia-se num escopo de dados pré-existente. Caso a situação seja diferente da mencionada, novos recortes são criados para a pessoa não identificada, ampliando a base.

Os autores encerram apresentando as considerações finais, que seguem o mesmo caminho apresentado em outros trabalhos. Deve-se melhorar o algoritmo para ser capaz de trabalhar fora de ambientes ideias, seja relativo à performance, falta de iluminação ou baixa resolução.

4 MATERIAIS E MÉTODOS

O presente capítulo tem como objetivo apresentar os meios em que a pesquisa foi realizada. Para tal, serão descritas as tecnologias utilizadas, bem como suas aplicações e seus resultados.

A pesquisa desenvolvida tem abordagem qualitativa. Conforme Zanella *et al.* (2006), esse procedimento busca entender a realidade a partir da ótica dos participantes. Foca-se no processo e não nos resultados obtidos com ele. Comumente acaba gerando um estudo que segue o método indutivo.

O presente trabalho segue o método dialético e indutivo. Apresenta questões e situações problema que da realidade, que geram outras problemáticas e possibilitam a realização do estudo. Dessa forma, as hipóteses podem surgir em diferentes momentos e elas determinam as possíveis conclusões (MORESI *et al.*, 2003).

Esta produção é uma pesquisa exploratória e de campo. Segundo Santos e Parra Filho (2012), ela parte de um pressupostos e, por meio de experimentos reais, tenta confirmar e validar essas variáveis. Para tal, realizou-se a construição de uma ferramenta com o intuito de confirmar as hipóteses levantadas.

A pesquisa consistiu no desenvolvimento de uma aplicação *mobile* para detecção facial e uma API para reconhecimento facial. A primeira aplicação é um aplicativo desenvolvido com React Native sendo suportado pelo *framework* Expo. Já a segunda, compõe-se de uma interface de programação desenvolvida em Python com FastAPI, apoiada em OpenCV, Dlib e a biblioteca Face_recognition.

4.1 Tecnologias

Neste capítulo serão descritas as tecnologias utilizadas para o desenvolvimento do presente trabalho. A estrutura está dividida por ordem de relevância. Por primeiro, são apresentadas as tecnologias centrais da aplicação e por fim, serão descritas as ferramentas auxiliares do processo.

4.1.1 OpenCV

De acordo com sua documentação, OpenCV (2023) define que *Open Source Computer Vision Library,* comumente chamado de OpenCV, é uma biblioteca *open source* de diversos algoritmos de visão computacional. Possui scripts estáticos e compartilhados, tendo portanto uma estrutura modular.

A biblioteca iniciou como um projeto de pesquisa dentro da Intel em 1998 e foi difundido no ano 2000. Seu foco era solucionar problemas de identificação de padrão, tal qual detecção de características, faces e pedestres. Em 2010 recebeu a aceleração de GPU, sendo implementado no padrão CUDA, da NVIDIA. Isso possibilitou um salto em performance, facilidade de utilização, pois passava a ser otimizado por padrão e total controle dos dados por parte do usuário (PULLI *et al.*, 2012).

A relevância desta biblioteca se dá pelo processo de democratização dos conceitos de visão computacional e, também, de desenvolvimento de trabalhos com a tecnologia, antes presentes apenas em grandes centros de pesquisa. Alcançou isso provendo uma base inicial já pronta de algoritmos confiáveis, capazes de detectar e reconhecer padrões, através de *machine learning* e processamento de imagem (BRADSKI; KAEHLER, 2008).

OpenCV (2023) ainda descreve, sua construção é feita em C++, no entanto, possui interfaces de conexão, APIs, que permitem a conexão em outras linguagens, bem como MATLAB, Java e Python, sendo essa última, a de maior interesse para este trabalho. A biblioteca exporta essas possibilidades de conexão através de pacotes instaláveis, presentes em cada uma das linguagens citadas acima.

4.1.2 Python

Python é uma linguagem de alto nível, *open source* e multiplataforma (LYNCH, 2023). Dessa forma, trabalha com comandos próximos à linguagem humana, é de uso aberto e capaz de rodar em qualquer sistema operacional, seja Linux, Unix, Mac OS X e Windows. Também se organiza por meio de classes, sendo definida como orientada a objetos (PYTHON, 2023).

Sua criação data do ano de 1991, por Guido van Rossum. Já no ano 2000, teve sua versão 2.0 lançada, a qual perdurou diversos anos. Em 2003, foi lançada a versão 3.0, a qual, é importante salientar, que não substituiu a versão antiga, ambas existiram paralelamente até 2020, quando a segunda distribuição foi descontinuada (LYNCH, 2023). No momento do desenvolvimento deste trabalho, a ferramenta encontra-se na versão 3.11.

A linguagem possui uma extensa comunidade, haja visto a grande possibilidade de utilização. Observa-se Python em jogos, aplicações para internet, desenvolvimento de ferramentas, introdução à programação, pesquisa, automação, entre outros (MATTHES, 2016).

Sua sintaxe é simplificada, abre-se mão dos pontos e vírgulas existentes nas linguagens de programação do tipo *C-based*, que seguem o padrão de organização do código idêntico ao C. Além disso, as chaves são trocadas por dois pontos e o conteúdo do bloco deve ser indentado uma tabulação para dentro. A ferramenta também trabalha com polimorfismo, bem como Java e C++ e faz uso de tipagem dinâmica, a qual pode trazer complicações quando se deseja trabalhar com sobrecargas (PYTHON, 2023).

4.1.2.1 FastAPI

Segundo Ramírez (2023), criador do FastAPI, cita na documentação da ferramenta que, o recurso é um *framework* de desenvolvimento *web* desenvolvido em Python, de versão igual ou superior a 3.7. Foca-se na *performance*, comparável ao NodeJS e GO, e superando os tradicionais frameworks, Django e Flask, velocidade de criação de código e reduzido número de bugs.

É de grande robustez, baseia-se nos padrões de desenvolvimento abertos, ou seja, visa compatibilidade com o maior número de aplicações possível. Apresenta

documentação automática, a qual descreve as rotas criadas pelo usuário. Por fim, funciona com múltiplos editores de código, podendo citar o VSCode e PyCharm (RAMÍREZ, 2023).

4.1.2.2 Dlib

Trata-se de um conjunto de ferramentas, desenvolvidas em C++, e fornece um apanhado de algoritmos para *machine learning* e resolução de problemas da vida real. Seu uso pode ser observado em diversas áreas de pesquisa e desenvolvimento (DLIB, 2022).

Tem seu foco direcionado para o desenvolvimento em C++, com arquitetura de contrato e componentização, portanto, disponibiliza testagem e *debugging* por padrão. Além disso, apresenta APIs, dessa forma apoia o funcionamento de scripts em outras linguagens de programação (KING, 2009). Essa característica é observada na biblioteca *face_recognition*, presente no desenvolvimento deste trabalho.

4.1.2.3 Face_recognition

Segundo Vadlapati, Velan e Varghese (2021), *face_recognition* é uma biblioteca Python, e conforme Geitgey, (2018), disponibiliza uma interface de linha de comando e tem seu funcionamento baseado no Dlib *toolkit*. Possui a função de detectar e reconhecer faces e faz isso através de pontos de interesse detectados.

O processo de identificação inicia com o recebimento de uma imagem contendo rostos, seja via requisição ou upload para armazenamento e realização de futuras comparações. No momento que é disparado o evento de reconhecimento, no novo retrato recebido é executada a função *face_landmarks* para identificar as marcações do rosto e convertê-las para NumPy. Elas consistem nas posições e dimensões que, por exemplo, olhos, boca e nariz se encontram na pessoa analisada. Concluindo, as duas imagens têm seus pontos extraídos e suas marcações comparadas, a fim de determinar a identidade do usuário.

4.1.2.4 Pickle

Segundo a documentação Python (2023), o ato de *pickling* é semelhante a uma serialização. Neste processo, a estrutura do objeto é convertida em um *byte stream* e gera o arquivo com formato *pickle*. Para sua deserialização, o termo definido é *unpickling*.

4.1.2.5 Postman

Segundo documentação da plataforma, Postman, Inc. (2023) define a ferramenta como um sistema de software para facilitar os processos que envolvem o desenvolvimento de APIs. Possui ferramentas de testagem interna com verificação de retornos, integração com plataformas de entrega e integração contínuas e estrutura de cloud. Apoia-se fortemente no conceito de API *first* e disponibiliza, também, a possibilidade de colaboração entre membros da equipe.

4.1.3 Javascript

Pode-se definir como a linguagem de programação para web. Sua adoção por parte de quase todos os navegadores é notável, os quais dependem de HTML (*Hypertext Markup Language*), CSS (*Cascading Style Sheet*) e JS (*JavaScript*) para seu funcionamento (FLANAGAN, 2011). Apresenta sintaxe *C based*, com ponto e vírgula e chaves, devido ao espelhamento na linguagem Java (WIRFS-BROCK; EICH, 2020).

O autor, Flanagan (2011), expõe que a ferramenta é de alto nível, trabalha com comandos próximos aos humanos, interpretada, não é compilada para bits (zeros e uns), sem tipagem e pode ser adequada tanto para programação orientada a objetos como funcional. Lista, também, os diversos tipos de uso que ela pode ter, seja na criação de scripts, programação web, programação backend, entre outros, com performance apoiada pelo NodeJS.

Seguindo com Flanagan (2011), a tecnologia apresenta apenas uma simples interface de programação para que seja possível trabalhar com textos, datas, *arrays* e expressões regulares. As demais funções ficam a cargo do local que ela está implementada.

Javascript, originalmente LiveScript, como descrito por Wirfs-Brock e Eich (2020), foi publicada em 1995, acoplada ao extinto navegador NetScape. Tinha a proposta de ser uma linguagem de scripts dinâmica, a qual poderia alterar o comportamento dos componentes Java sem a necessidade da geração ou carregamento de uma nova página.

A tecnologia encontra seu maior problema quando precisa ser utilizada e mantida em sistemas de grande escala, muitos colaboradores e linhas. Observando isso, foi criado o TypeScript. Ele consiste em um *superset* para o EcmaScript, real nome da linguagem, na sua quinta versão (BIERMAN; ABADI; TORGERSEN, 2014).

O sistema de tipagem funciona adicionando a possibilidade aos desenvolvedores de definir o tipo, estaticamente, de suas variáveis, parâmetros, retornos e fazer uso de *code annotations*. Por fim, o código é transpilado para JS, ou seja, convertido para a linguagem original. Esse comportamento acontece devido ao NodeJS, motor de funcionamento, que só interpreta Javascript.

4.1.3.1 NodeJS

De acordo com sua documentação, NodeJs é runtime de JavaScript, é uma plataforma na qual o JS das aplicações será interpretado. Opera na arquitetura *single thread*, a qual faz pouca ou nenhuma ativação das funções de *input* e *output* do sistema operacional, possibilitando o funcionamento no modelo *non-blocking* (OPENJS FOUNDATION, 2023).

Sua construção é baseada num *loop* de eventos ao invés de uma sequência de bibliotecas e seu comportamento é definido a partir de callbacks. Disponibiliza amplo suporte para a estrutura HTTP (*Hypertext Transfer Protocol*), haja visto o passado do seu motor de funcionamento, V8, do *Google Chrome*. Possui também, apesar do modelo de núcleo único, a funcionalidade de divisão de processos, para aumentar o paralelismo e o balanceamento de carga, quando necessário (OPENJS FOUNDATION, 2023).

De acordo com Herron (2013), junto com o *runtime*, a ferramenta disponibiliza o NPM (*Node Package Manager*). Ele provê ao conjunto uma maneira de distribuição de pacotes JS, ou seja, códigos prontos, tal qual o funcionamento do React Native e Expo, utilizado neste trabalho. Isso facilita na integração de novos recursos nas aplicações, haja vista que apenas a configuração básica é necessária.

Herron (2013) aponta também que o gerenciamento de pacotes ocorre no arquivo *package.json*, que determinará o nome, a versão e se ele pertence à instância de desenvolvimento ou produção, a figura 13 demonstra o funcionamento do arquivo. Essa estratégia permite o não versionamento dos códigos dos pacotes, diminuindo o tamanho final do produto para publicação.

Figura 13 - Estrutura de um arquivo package.json.

Fonte: Elaborado pelo autor (2023).

4.1.3.2 React Native

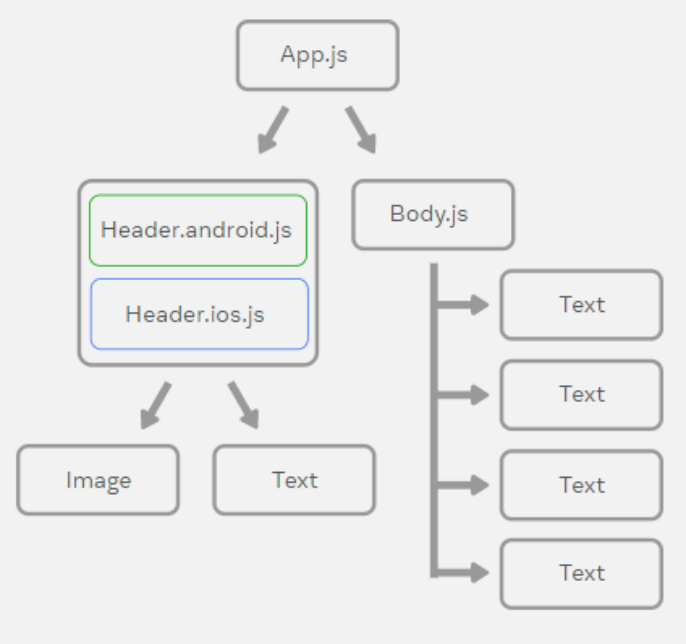
A tecnologia surgiu após o sucesso do ReactJS como biblioteca para desenvolvimento front-end web. Para tal, o Facebook, hoje Meta, realizou conferências para desenvolvimento da ferramenta, com a premissa de criar um SDK (*Software Development Kit)* capaz de compilar código nativo para ambas plataformas dominantes do mercado, Android e IOS (WU, 2018).

Segundo sua documentação, Meta Platforms, Inc. (2023) define que, React Native é uma biblioteca capaz de converter código JSX (*JavaScript Xtension*) para

código mobile nativo. Sustenta-se fortemente na característica de reaproveitamento da lógica presente no ReactJS, simplificando a curva de aprendizado.

Além disso, permite o desenvolvimento paliativo ou total, ou seja, pode-se, gradativamente, substituir aplicativos existentes, em código nativo, para a tecnologia do Native. Outro atributo destacado pelo autor, é a capacidade de gerar código nativo, para Android e IOS, com apenas uma sintaxe (META PLATFORMS, INC., 2023). A figura 14 demonstra o isolamento dos cabeçalhos presentes nas plataformas para qual a ferramenta é capaz de compilar e a execução da comunicação, via API para com os componentes JSX.

Figura 14 - Demonstração do isolamento dos componentes específicos de plataforma e comunicação com os componentes React Native.



Fonte: Meta Platforms, Inc. (2023).

Outro ponto positivo, é a grande comunidade existente para a tecnologia. Em 2018, como descrito por Wu (2018), o repositório que contém o código do React Native era aquele com maior número de estrelas no Github. O sucesso pode ser observado pela contribuição de grandes companhias de software no desenvolvimento da ferramenta. Isso se deve, também, seguindo com o autor, pela capacidade de integrar padrões de desenvolvimento web no mobile e fazer o uso de itens nativos dos aparelhos, como câmera e armazenamento. Isso, como exposto anteriormente, diminui a curva de aprendizagem e atrai um maior número de colaboradores.

4.1.3.3 JSX

JSX, de forma simples, é a combinação de JavaScript com estrutura HTML. Isso fornece maior dinamismo na criação de aplicações e componentização. Isso permite estender as tags de marcação e criar novos componentes, os quais podem agrupar diversas *tags* ou até mesmo outros componentes, além de adicionar lógica computacional à estrutura (CHINNATHAMBI, 2017).

Chinnathambi (2017) ainda explica, a lógica de componentização, por fim, é transformada para JS nativo, o qual o navegador ou plataforma que está interpretando a linguagem, conseguem entender. Essa prática otimiza a construção da interface dos sistemas, pois visa acelerar o entendimento da lógica por parte do *runtime.*

4.1.3.4 Expo

Conforme definido por sua documentação, Expo (2023), é um *framework* para desenvolvimento *mobile* e *web open source* em JavaScript ou TypeScript. Ela provê a estrutura inicial, bem como diversos itens que podem ser úteis durante o desenvolvimento de aplicações.

Conta também com uma interface de linha de comando, a qual permite que o desenvolvedor acione determinadas ações as quais a tecnologia provê. Soma-se a isso, a existência de *builds* pré prontos para *deploy* nas lojas de aplicativos, *Play Store* e *App Store*, os quais facilitam a entrega para essas plataformas (EXPO, 2023).

4.1.3.5 ML Kit

O ML Kit (Machine Learning Kit), da Google, é um SDK para aparelhos móveis. Ele, através do API Vision, fornece uma gama de recursos de visão computacional. O autor, Google (2003), salienta que os códigos são executados diretamente no dispositivo, permitindo a utilização performática da ferramenta, até mesmo *off-line*.

4.1.3.6 Base64

Segundo Josefsson (2006), Base64 é uma maneira de simplificar a escrita de valores binários. Para tal, utiliza-se a tabela ASCII em combinação com o resultado de um valor binário de seis bits. Dessa forma, o valor decimal convertido determinará qual letra será posicionada. Por exemplo, o valor binário 010010, resulta em 18 decimal, portanto, assumirá o valor de S no padrão Base64.

4.1.4 Ubuntu

A disponibilização do ambiente *backend* desenvolvido foi feito através da plataforma Linux Ubuntu 22.04 LTS *Jellyfish*. Conforme apresentado em sua documentação, Canonical LTD. (2023), a ferramenta é um sistema operacional moderno capaz de operar nas frentes de *desktop*, servidor, computação em *cloud* e *internet* das coisas. Observa-se sua crescente utilização em diversas empresas, tais como Amazon, Microsoft, Google e Univates.

4.2 Desenvolvimento

O projeto consiste na criação de um aplicativo para dispositivos móveis, Android e IOS, para reconhecimento facial. A concepção se deu devido à necessidade de registro da presença de pessoas em eventos de forma prática, a qual, a partir de um cadastro prévio, seja capturada a imagem da pessoa e posteriormente, sob responsabilidade do evento, a validação seja feita. Isso reduz a necessidade de porte de identificações e aplicativos para o visitante.

A arquitetura, conforme evidenciado pela figura 15, apoia-se em uma arquitetura desacoplada, separada entre *front end*, responsável por detectar a face do indivíduo a ser autenticado e o *back end*, que faz o processamento. As seções seguintes apresentarão o fluxo da aplicação baseado em sua arquitetura.

Figura 15 - Arquitetura da aplicação.

Fonte: Elaborada pelo autor (2023).

O aplicativo, construído com Expo e React Native é responsável pela detecção primária através do ML Kit. Posteriormente, é feita uma chamada HTTP para a API Python realizar o reconhecimento. Nesse processo, o conteúdo da imagem é enviado em formato Base64, para possibilitar o tráfego. O processamento é realizado em Python que está estruturado no *framework* FastAPI. Este utiliza a *lib face_recognition* que possui dependências da Dlib. Com o resultado de reconhecimento obtido, envia-se este ao aplicativo para que sejam feitas as confirmações necessárias.

4.2.1 Estrutura para coleta de rostos

Nesta seção o objetivo é ilustrar a estrutura da coleta de dados. Conforme apresentado anteriormente, o processo inicia-se na coleta de uma imagem contendo o rosto da pessoa a ser reconhecida. Neste passo, é importante pontuar o funcionamento da ferramenta de reconhecimento facial, a qual utiliza histograma orientado a gradientes (HOG), portanto, depende de como a luz reflete no rosto analisado.

Tendo isso em vista, utilizou-se uma base com um clipe para se acoplar à mesa e que disponibilizasse um suporte para o aparelho celular, bem como um *ring light* para garantir iluminação padronizada e razoável no ambiente. Estes aparelhos tinham a função de eliminar os possíveis ruídos relativos à iluminação. A figura 16 demonstra a estrutura montada para testagem em campo.

A base contava com alguns *presets* de iluminação. Conforme verificação prévia, evidenciou-se que a luz amarela, mais natural, apresentava um funcionamento melhor se comparado com a branca. Isso pode estar relacionado com a grande iluminação branca já existente nos locais de testagem.

Figura 16 - Base de apoio do celular para capturar os rostos.

Fonte: Elaborado pelo autor (2023).

4.2.2 Aplicativo de celular

A aplicação para dispositivos móveis, dentro do escopo da arquitetura é a responsável pela detecção inicial de faces e posteriormente, o envio da imagem capturada para processamento no servidor. Além disso, há também a função de retornar o resultado apresentado para o usuário e aguardar sua confirmação.

Para o entendimento das seções seguintes, relativas à implementação do aplicativo *mobile*, faz-se necessário o entendimento do que é um objeto de estado. De maneira simplificada, é uma forma de manter um valor em memória durante o tempo de execução do app. Compõe-se de duas variáveis, uma que é o próprio valor e outra é o nome da função que fará a alteração do seu conteúdo, por exemplo nome e *setNome*. É de suma importância que os valores de estado sejam manipulados através do método definido, a fim de evitar *memory leaks*.

A detecção é realizada pela ferramenta Expo Camera, a qual faz utilização da tecnologia ML Kit da empresa Google. A figura 17 apresenta a configuração do recurso

dentro do app. Para tal, operou-se com o modo rápido, sem registro dos pontos de interesse da face, bem como suas classificações e acompanhamento do rosto na tela do dispositivo. Essa abordagem se deve pois o aplicativo tem apenas a função de validar se há a presença de um rosto na imagem e, dessa forma, otimizar o tempo resposta.

Figura 17 - Configuração do Expo Camera.

```
<Camera
  ref={cameraRef}
  style={styles.camera}
  type={CameraType.front}
  onFacesDetected={handleFacesDetected}
  faceDetectorSettings={{
    mode: FaceDetector.FaceDetectorMode.fast,
    detectLandmarks: FaceDetector.FaceDetectorLandmarks.none,
    runClassifications: FaceDetector.FaceDetectorClassifications.none,
    minDetectionInterval: detectionInterval,
    tracking: false,
  }}
/>
```

Fonte: Elaborado pelo autor (2023).

No momento da localização de um rosto, é feito o uso do recurso do React Native de REF, a qual permite o acesso aos elementos mostrados na tela, para realizar a captura da imagem presente na tela. O método desenvolvido fornece uma imagem para ser enviada para o servidor.

A figura 18 apresenta a forma que o REF foi abordado no desenvolvimento da aplicação. O arquivo gerado segue as seguintes características: formato PNG (*Portable Network Graphics*); qualidade 0.8; altura e largura de 720 por 720 *pixels*; com resultado em base64. Aponta-se o cálculo realizado para adequar o tamanho da imagem para o dispositivo utilizado, para tal, faz-se a divisão do tamanho desejado pela quantidade total de *pixels* que o *smartphone* possui.

Figura 18 - Método de captura da imagem com uso de ref.

```
const takePic = useCallback(async (): Promise<string> => {
  try {
    return await captureRef(cameraRef, {
      format: 'png',
      quality: 0.8,
      result: 'base64',
      width: 720 / PixelRatio.get(),
      height: 720 / PixelRatio.get(),
    })
  } catch (ex) {
    console.error(ex, 'Some error have occurred capturing the face image')
  }
}, [])
```

Fonte: Elaborado pelo autor (2023).

Em sequência é iniciado o processo de envio dos dados de imagem para processamento no servidor. Para tal, requisita-se a rota "*identify"*, responsável pelo reconhecimento do usuário. Enviam-se o *base64* bem como o nível de tolerância para análise. Optou-se por manter o nível de tolerância no aplicativo *mobile* devido aos recursos de estado, *useState*, que o React Native fornece. Esta técnica será descrita em conjuntura com o método de confirmação. Envia-se também, os headers de "*Content-Type: multipart/form-data*", haja visto que o servidor espera os dados por meio de um *form-data*.

Com o resultado obtido, é populado o objeto de estado de identidade, bem como é estendido o intervalo entre requisições para 50 segundos, nos métodos *setIdentity* e *setDetectionInterval*, respectivamente. O primeiro item contém os dados do indivíduo, sendo eles: *ID* do reconhecimento; nome; porcentagem de semelhança e um valor booleano que indica se houve uma combinação. O valor de identidade é testado no momento em que um rosto é constatado, pois o envio de dados deve apenas ocorrer caso haja um rosto e não houver alguém já identificado. No código, isso ocorre na instrução "*if (face && !identity)*". A figura 19 demonstra a implementação da requisição.

Figura 19 - Implementação da requisição de reconhecimento.

```
const handleFacesDetected = useCallback(
  async ({ faces }: FaceDetectionResult) => {
    const face = faces[0] as FaceDetector.FaceFeature
    if (face && !identity) {
      const capturedSnapshot = await takePic()
      try {
        const { data } = await api.post<IRecognitionResult>(
          'identify',
          {
            data: capturedSnapshot,
            tolerance,
          },
          {
            headers: {
              'Content-Type': 'multipart/form-data',
            },
          },
        )
        if (!data.match_status) {
          console.log('no match found')
          return
        }
        setIdentity(data)
        setDetectionInterval(50000)
      } catch (error) {
        console.log(error)
      }
    }
  },
  [identity, takePic, tolerance],
)
```

Fonte: Elaborado pelo autor (2023).

Após receber os dados de reconhecimento e fazer a população do objeto de identidade, são renderizados os elementos de tela que possibilitam a confirmação do reconhecimento. Para tal, há a mensagem que evidencia o nome do indivíduo e dois botões para confirmar ou negar a identificação. A figura 20 demonstra a implementação da renderização condicional.

Figura 20 - Renderização condicional da seção de confirmação de identidade.

```
{identity && (
  <View style={styles.confirmation}>
    <View style={styles.confirmationContainerText}>
      <Text style={styles.confirmationText}>
        You are {identity.name}. Is it correct?
      </Text>
    </View>
    <View style={styles.confirmationBox}>
      <TouchableOpacity
        style={styles.confirmationButton}
        onPress={() => {
          handleConfirmation(true)
        }}
      >
        <Text style={styles.confirmationButtonText}>Yes!</Text>
      </TouchableOpacity>
      <TouchableOpacity
        style={styles.confirmationButtonNegation}
        onPress={() => {
          handleConfirmation(false)
        }}
      >
        <Text style={styles.confirmationButtonText}>No.</Text>
      </TouchableOpacity>
    </View>
  </View>
)}
```

Fonte: Elaborado pelo autor (2023).

Os botões de confirmação acionam a função para confirmação de identidade. O método faz o envio para a rota “*confirmation*” com os seguintes dados: ID da identificação, nome do indivíduo e confirmação. Por fazer utilização de *form-data*, também é utilizado o cabeçalho “*Content-Type: multipart/form-data*”.

Na sequência, ao completar o envio, observa-se o valor de confirmação, caso falso, a tolerância é reduzida em 0.05, a cada requisição, até atingir 0.4, a fim de exigir uma análise mais precisa, do contrário, é restaurada para o valor padrão, 0.55. Por fim, sempre são retornados os valores de identidade e intervalo de identificação para os padrões, nulo e 2 segundos, para que a tela de detecção seja restabelecida e o processo reinicie. A figura 21 ilustra como a função de confirmação de identidade foi construída.

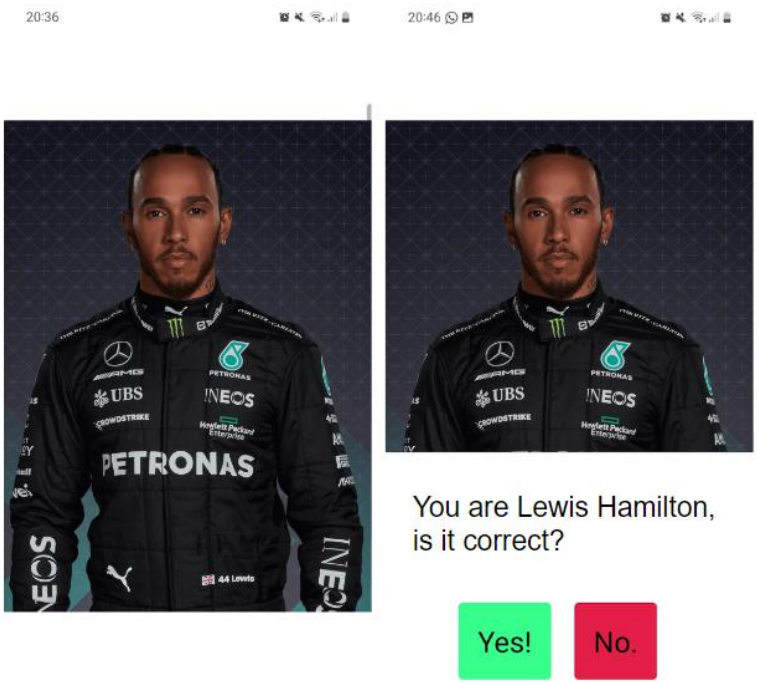
Figura 21 - Função de confirmação de identidade.

```
const handleConfirmation = useCallback(
  async (confirmation: boolean) => {
    if (!identity) return
    await api.post(
      'confirmation',
      {
        id: identity.id,
        name: identity.name,
        confirmation: confirmation ? 'yes' : 'no',
      },
      {
        headers: {
          'Content-Type': 'multipart/form-data',
        },
      },
    )
    if (!confirmation) {
      const reducedTolerance = tolerance - 0.05
      setTolerance(reducedTolerance)
    } else {
      setTolerance(initialTolerance)
    }
    setIdentity(null)
    setDetectionInterval(initialDetectionInterval)
  },
  [identity, tolerance],
)
```

Fonte: Elaborado pelo autor (2023).

Por fim, o resultado apresentado para o usuário é o demonstrado na figura 22. À esquerda está a imagem da tela do celular executando a detecção. Já à direita, é mostrada a tela de confirmação dos resultados de reconhecimento fornecidos pelo servidor.

Figura 22 - Telas da aplicação mobile.



Fonte: Elaborado pelo autor (2023).

4.2.3 Aplicação de reconhecimento

A aplicação de reconhecimento facial foi desenvolvida utilizando Python e o padrão FastAPI. A linguagem apresenta diversos métodos que agilizam o desenvolvimento, assim como o *framework* escolhido, o qual por meio de anotações permite a definição do tipo de método HTTP implementado e suas rotas.

A escolha dessas tecnologias ocorreu também pela performance apresentada na sua combinação. Além disso, a API *face_recognition* é implementada nesta linguagem de programação, portanto, há a facilidade de integração entre elas.

Como base de dados para análise, foram utilizados arquivos físicos, com extensão PNG e seus pontos de interesse, explicitados posteriormente, em formato *pickle*. Os dados de reconhecimento e confirmação foram salvos em arquivos CSV. Adotou-se essa estratégia tendo em vista a simplicidade e o ganho de performance, pois tudo está sendo criado ou recuperado do próprio diretório da aplicação.

4.2.3.1 Rota de identificação

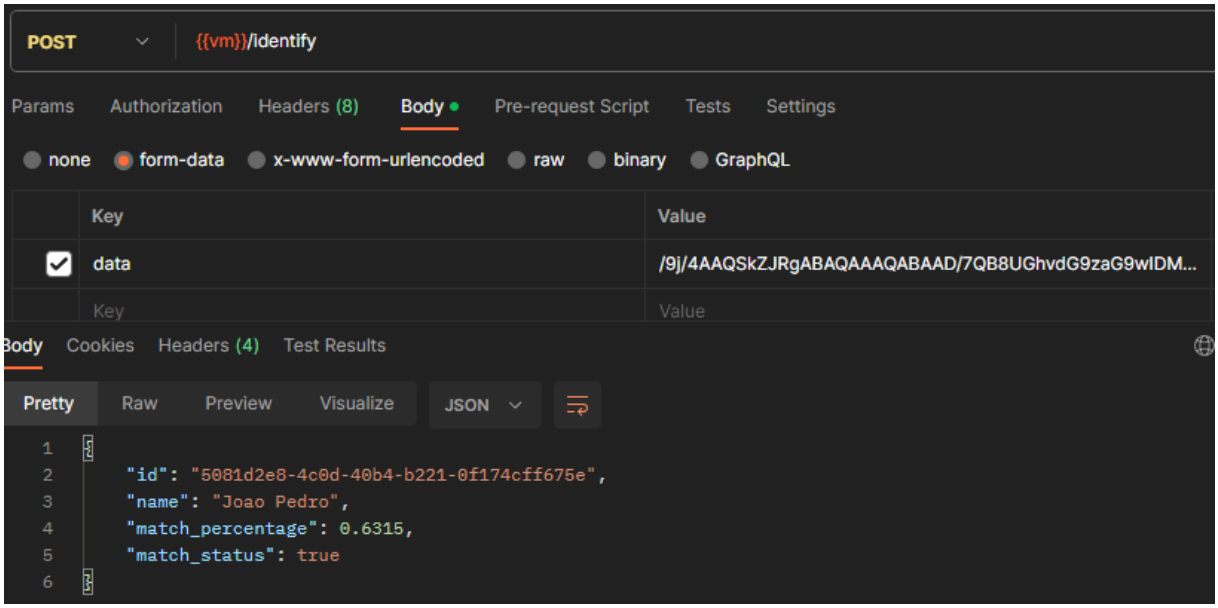
A rota *identify* é a primeira a ser requisitada pelo aplicativo. Em um primeiro momento, é gerado uma identificação para o reconhecimento, esta será responsável por fazer a conexão entre resultado e sua confirmação. Posteriormente um arquivo de imagem contendo os dados recebidos no corpo da requisição é construído. Este por sua vez é enviado para o método de reconhecimento em conjunto com a tolerância, também recebida.

O método, por sua vez, inicia obtendo os dados de magnitudes da imagem criada, através da função *face_encodings* da biblioteca *face_recognition*. Este processo é realizado através de uma rede neural convolucional, a qual já é pré-configurada, por meio do pacote *face-recognition-models*, na versão 0.3.0, com modelos para detecção de rostos. O passo de constatação de faces é repetido com o intuito de trazer os dados no formato especificado pela biblioteca.

Na sequência a base de rostos é iterada por inteiro. Esta abordagem tem o intuito de mitigar falsos positivos, que podem aparecer caso a iteração seja realizada somente até uma combinação positiva. Em cada iteração, são comparados os histogramas de gradientes de cada imagem salva com a de análise. No momento em que uma correspondência é detectada, calcula-se a proximidade entre a amostra e a referência. Caso o resultado seja menor que o anterior, os valores de nome e porcentagem de similaridade são alterados, do contrário, o fluxo segue.

Os dados retornados são salvos em um arquivo CSV referente ao dia da análise. Por fim, são retornados ao cliente os dados de identificação da requisição, nome da pessoa identificada, porcentagem de correspondência e se houve uma correspondência. A figura 23 demonstra a execução da requisição ao servidor de hospedagem, a partir do cliente de API Postman.

Figura 23 - Requisição para a rota identify.



Fonte: Elaborado pelo autor (2023).

4.2.3.2 Rota de confirmação

A rota de confirmação apresenta um funcionamento mais simplificado e tem apenas a função de registrar se o reconhecimento feito anteriormente foi considerado correto. Por sua vez, requer os seguintes dados ao cliente que fará a requisição: ID do reconhecimento; nome do indivíduo; resultado da confirmação. Por fim, realiza o salvamento dos dados em um arquivo CSV.

4.2.3.3 Rotas de cadastro

Para que o reconhecimento seja possível é necessário que os rostos sejam cadastrados previamente. Para cumprir tal função, foram elaboradas duas rotas. A primeira, que trabalha de maneira individual, tem a finalidade de ser utilizada na evolução do projeto, pois permite que as ações sejam operadas pelo usuário, através do *front end* do app. Já a segunda, focada no ambiente de testagem, permite o cadastro de um grande número de referências na base.

O funcionamento dessas se difere na forma que iniciam o processo. A primeira recebe os dados por meio do corpo da requisição, como parâmetros. Já a segunda busca os arquivos no diretório *batch*, criado na raiz da aplicação.

Na sequência, os processos tornam-se iguais. São criados os arquivos de imagem na pasta *db*. Para tal, é utilizada a extensão PNG. Para finalizar, com o intuito de acelerar o processo de reconhecimento, executado posteriormente, são obtidos os histogramas de gradiente, ou seja, os vetores de iluminação dos usuários que estão sendo cadastrados e esses são salvos em um arquivo *pickle*.

4.2.4 Implantação

Ambas aplicações foram disponibilizadas para acesso além da máquina de desenvolvimento. Para tal, no aplicativo móvel, utilizou-se o motor de construção que o próprio Expo disponibiliza. Com ele, foi construído um aplicativo Android em modo preview, com a extensão APK (*Android Application Package*).

Já para o servidor de reconhecimento, foi utilizada uma máquina virtual fornecida pela Univates. Nela, utilizou-se a infraestrutura Linux Ubuntu 22.04 *Jellyfish* para instalação dos pacotes necessários, cita-se Cmake e Dlib, bem como a criação do ambiente Python para a execução da aplicação.

O envio de arquivos, tanto da máquina local para o servidor, quanto o caminho contrário, foi executado através do comando SCP. A figura 24 demonstra um envio de imagens para a realização do cadastro em massa. Para acesso à VM (*Virtual Machine*) foi utilizado o comando SSH.

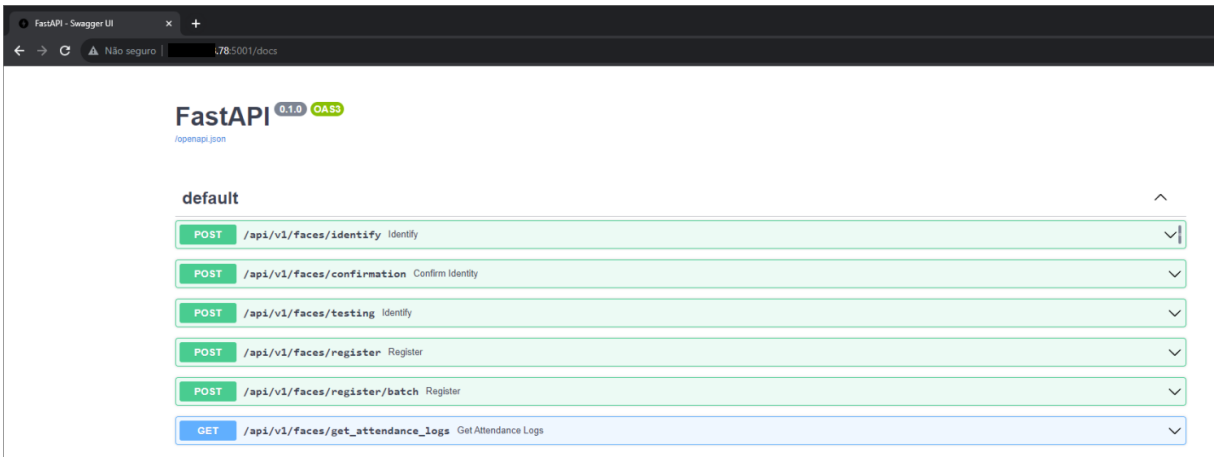
Figura 24 - Utilização do comando SCP.

Fonte: Elaborado pelo autor (2023).

Após a configuração do ambiente e o envio do projeto para a máquina virtual, executou-se o comando para iniciar a aplicação: “python3 main.py”. Essa ação faz com que o *back end* fique aguardando a chamada das rotas definidas a partir do Front end. A figura 25 demonstra as rotas existentes por meio da documentação gerada

pela ferramenta Swagger, no caminho: “endereço/docs”, criada por padrão pelo *framework* FastAPI.

Figura 25 - Rotas da aplicação



Fonte: Elaborado pelo autor (2023).

A fim de manter o desenvolvimento organizado e permitir o versionamento, tendo em vista o desenvolvimento evolutivo, o código criado foi enviado para plataforma Github. O projeto está disponível no seguinte endereço: https://github.com/JpAudibert/WhoRU.

5 TESTES E ANÁLISE DOS RESULTADOS

O presente capítulo apresenta a descrição dos processos utilizados e os resultados dos testes feitos, bem como a discussão desses. A testagem, será dividida em três seções: teste piloto, teste de campo e discussão dos resultados obtidos.

5.1 Teste piloto

Após a construção da primeira versão da aplicação de reconhecimento, foi executado um teste piloto. O intuito era colocar a prova o que foi construído em um ambiente controlado e com a possibilidade de erros.

Para tal, foi utilizada a aula de Computação Gráfica do professor Fabrício Pretto. Previamente, foi solicitado aos alunos que, facultativamente, realizassem seus cadastros através do formulário Google que solicitava nome; número de matrícula e duas imagens do rosto. No total, 10 alunos participaram. O cadastro foi realizado por meio do cadastro em remessas. No total, o ambiente contava com 240 faces aleatórias, com a finalidade de testar a sua performance.

A testagem iniciou com certo sucesso, com respostas positivas com o tempo médio em 2,5 segundos. No entanto, devido à implementação daquele momento, começou a gerar diversos falsos positivos. O erro, evidenciado pela figura 26, consistia na verificação do processo de iteração. A base de rostos era iterada até que uma correspondência era identificada. Isso é um problema grave para o reconhecedor, tendo em vista que nem todos os rostos eram validados e somente a primeira combinação era retornada. Soma-se a isso, a falta de dados salvos, o que impedia a análise de similaridade, bem como a relação entre confirmações dos reconhecimentos.

Figura 26 - Implementação errada da iteração da base.

```
while (not match) and (j < len(db_dir)):
    path_ = os.path.join(DB_PATH, db_dir[j])
    file = open(path_, "rb")

    if os.stat(file.name).st_size == 0:
        return "corrupted file", False

    embeddings = pickle.load(file)[0]
    match = face_recognition.compare_faces([embeddings], embeddings_unknown)[0]
    j += 1
```

Fonte: Elaborado pelo autor (2023).

Nas demais frentes, observou-se também um erro relativo ao uso da ferramenta Expo Camera. Segundo fontes de pesquisa, o erro é relativo ao uso da funcionalidade de detecção de faces e no momento das testagens, nos meses de setembro e outubro de 2023, ainda não havia sido resolvido. Também foi testada a estrutura de posicionamento do celular e iluminação, as quais funcionaram com êxito.

5.2 Teste de campo

Após evolução do projeto e melhorias de implementação, demonstrado na figura 27, realizou-se os testes de campo. Para esses, a base de dados passou a ser iterada independentemente de combinação positiva, os registros de reconhecimento foram aumentados, foram aumentados os dados retornados para o aplicativo móvel, além da construção da lógica de diminuição de tolerância caso houvesse um reconhecimento errado e polimentos na operação do reconhecedor.

Figura 27 - Melhorias para os testes de campo.

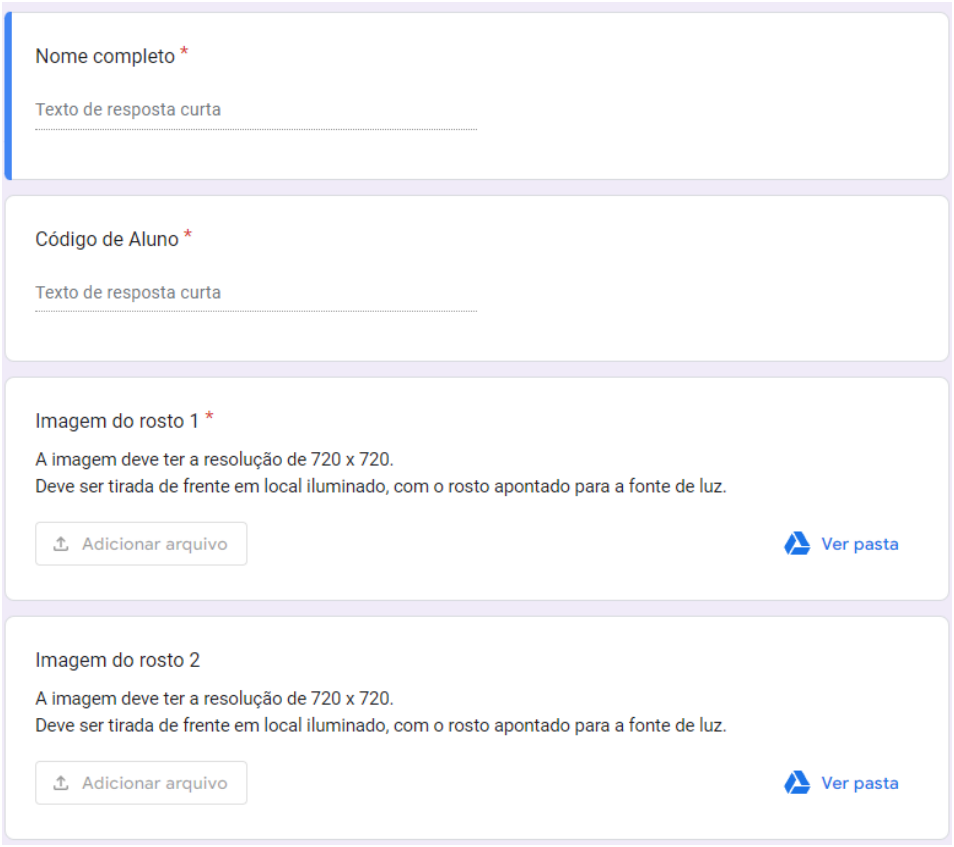
```
for j in range(len(db_dir)):
    path_ = os.path.join(DB_PATH, db_dir[j])
    file = open(path_, "rb")

    if os.stat(file.name).st_size == 0:
        return "corrupted file", match_percentage, match

    try:
        pickleEmbeddings = pickle.load(file)
        if len(pickleEmbeddings) > 0:
            embeddings = pickleEmbeddings[0]
    except:
        continue

    match = face_recognition.compare_faces([embeddings], embeddings_unknown, tolerance)[0]

    if match:
        new_face_distance = face_recognition.face_distance([embeddings], embeddings_unknown)[0]

        if new_face_distance < face_distance:
            person_name = db_dir[j][:-7]
            match_percentage = 1 - new_face_distance
            is_match = True

            face_distance = new_face_distance
```

Fonte: Elaborado pelo autor (2023).

A realização dos testes oficiais foi durante os eventos do CRIE CONEXÕES, realizados de 06/11/2023 a 09/11/2023. Como público-alvo para os testes, foram selecionados os minicursos, ministrados durante o evento. O ambiente, bem como no teste anterior, contava com 245 faces aleatórias para dificultar o trabalho do algoritmo.

O processo se iniciou pelo cadastro dos rostos dos participantes, este por sua vez, facultativo. A ação foi realizada através do preenchimento do mesmo formulário. Para este momento, foi criado um código QR para facilitar o acesso dos alunos. A figura 28 apresenta o formulário. Solicitou-se apenas imagens de 720 x 720 *pixels* com o intuito de acelerar o processo de cadastro.

Figura 28 - Formulário de cadastro



Fonte: Elaborado pelo autor (2023).

O questionário contou com um total de 31 respostas entre os dois momentos de testagem. Todas as imagens foram enviadas para a base de dados, pelo método de remessa, totalizando 276 rostos. Aponta-se que as imagens contavam com pessoas com barba, óculos, diferentes tipos de cabelo, adereços e posições de captura. Esses elementos contribuem para o aumento da dificuldade de reconhecimento.

Os testes de campo obtiveram grande sucesso. Os reconhecimentos eram realizados em um tempo adequado, cerca de 5 segundos. Esse tempo de resposta é bom, pois leva-se em conta todos os passos feitos pelos usuários para o funcionamento da aplicação. São eles: posicionamento; captura do rosto detectado; envio para análise e retornos; saída do espaço de detecção e toque na tela para confirmação.

Com relação às falhas, foram observadas apenas duas situações. A primeira, relativa a altura do suporte para o celular e iluminação, alunos mais altos obtiveram resultados errados em um primeiro momento, o erro foi resolvido ao solicitar que se

abaixassem. Um segundo problema identificado é a detecção de pessoas com pele preta. O algoritmo de histograma apresenta deficiência na captura dos pontos de interesse em pessoas com essa tonalidade de pele. A figura 29 apresenta um caso de teste que não obteve êxito. Observa-se que esse problema não é único da aplicação desenvolvida, ferramentas como Instagram também apresentam o mesmo comportamento.

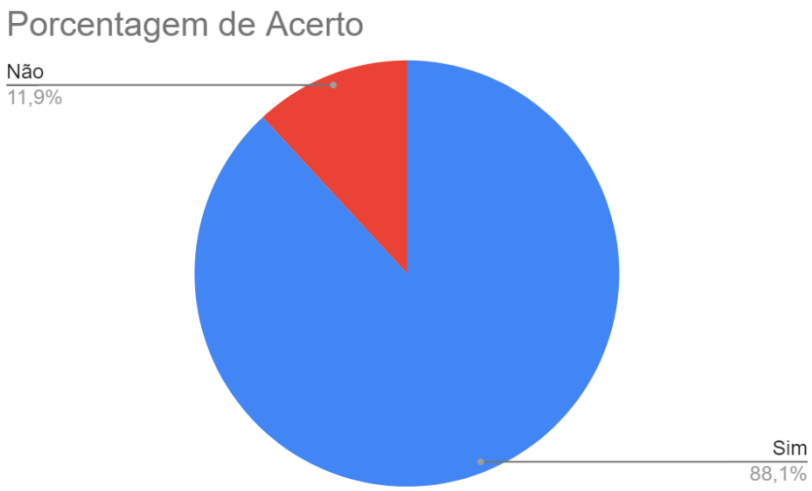
Figura 29 - Face em que o algoritmo de detecção apresentou falha.

Fonte: Greenwood Campbell Ltd (2023).

5.3 Resultados

Os resultados das testagens de campo são de grande sucesso. Totalizaram-se 42 reconhecimentos entre os dois dias. Aponta-se a disparidade entre o número cadastrado pois houve falsos positivos, bem como pessoas sendo reconhecidas mais de uma vez. Salienta-se que a participação era de cunho facultativo. O gráfico 1 demonstra as porcentagens de acertos. Evidencia-se a quantidade de 88,1% de correspondências corretas, a qual pode ser aumentada ao melhorar o posicionamento da câmera que realiza a coleta das imagens para análise. Do lado contrário, observa-se 11,9% de falsos positivos, totalizando cinco reconhecimentos.

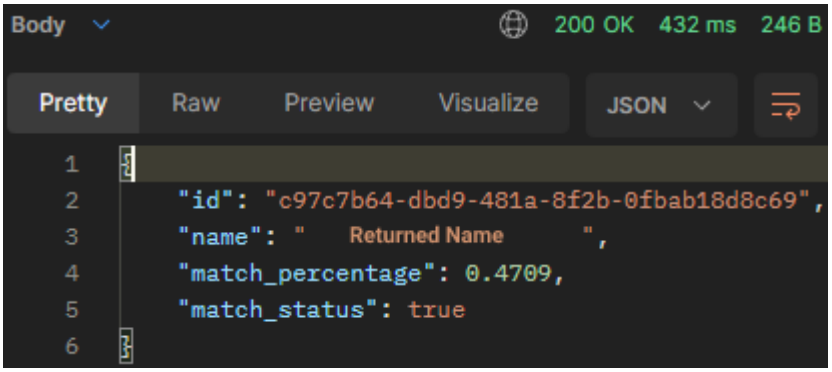
Gráfico 1 - Gráfico de confirmações de reconhecimentos.



Fonte: Elaborado pelo autor (2023).

Conforme citado anteriormente, utilizou-se uma base que totalizava 276 faces, sendo 31 cadastradas com o intuito de serem reconhecidas. Durante o teste piloto, obteve-se a performance de retorno de dados de 1,1 segundo, mesmo com a aplicação hospedada remotamente. Ao realizar as correções e polimentos necessários para evitar falsos positivos e acelerar o algoritmo, obteve-se uma performance média de 750 milissegundos. A figura 30 demonstra uma execução no cliente Postman realizando conexão na máquina virtual em que o *back end* está hospedado e recebendo retorno em 432 ms.

Figura 30 - Demonstração do tempo de resposta da aplicação final.



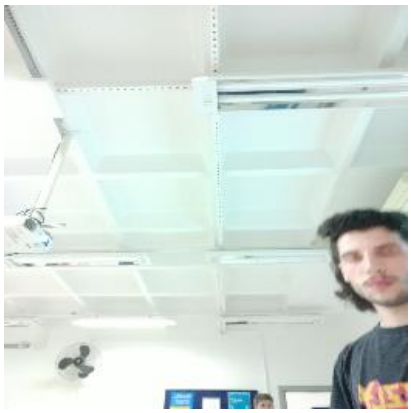
Fonte: Elaborado pelo autor (2023).

O tempo de resposta variável é relativo ao número de imagens que o usuário tem cadastrado na base, quanto maior o número, maiores são as chances de ser

reconhecido corretamente. Por outro lado, como há maior quantidade de correspondências, há maior quantidade de cálculos de proximidade de faces, aumentando o tempo de resposta.

Observou-se também a rápida detecção realizada pelo aplicativo de celular. As faces eram apontadas logo que entravam no quadrante em que a câmera capturava. Apesar do posicionamento não ideal, os resultados de reconhecimento eram corretos. A figura 31 ilustra, com o próprio autor, um exemplo dessa ocorrência.

Figura 31 - Captura de imagem no momento de entrada no enquadramento.



Fonte: Elaborado pelo autor (2023).

Como resultado disso, foi constatado um baixo índice de proximidade entre os rostos analisados e os de referência, de 47,9%. Isso se deve pois as faces capturadas estavam muito distantes da referência enviada. A ocorrência é devido a posição da câmera, luminosidade e pose que foi utilizada na foto de cadastro. O valor absoluto de proximidade não é expressivo, no entanto, para o contexto do experimento ele é suficiente, pois a assertividade da ferramenta permaneceu em um alto padrão, de 88,1%.

A realização das medições e construções do gráfico de assertividade apenas foi possível pois houve a criação de arquivos que registrassem as ocorrências. Eles são separados por data e são respectivos aos reconhecimentos e as confirmações. A figura 32, demonstra a combinação desses, a fim de facilitar a criação de gráficos e realização de análises.

Figura 32 - Arquivo combinado de reconhecimentos e suas confirmações

Id	Name	Match Percentage	Date	Confirmation
cf66c9f1-5a07-4444-9cb1-214e2b071e91	Name Surname	4.794	2023-11-08 00:04:14.195709	yes
7bed8dba-4b47-4d5c-a684-facd74eeb670	Name Surname	502	2023-11-08 00:04:29.919596	yes
9b547181-ff90-4c75-80f8-63b13b783379	Name Surname	5.851	2023-11-08 00:05:03.086881	yes
60642e63-57e2-448f-8f02-a1693d6c91b9	Name Surname	4.786	2023-11-08 00:05:27.307018	yes
88a448e6-a313-4f01-bea6-361e516f2717	Name Surname	5.072	2023-11-08 00:05:42.744439	yes
a52d263c-9d76-4fbc-8770-488e1465ac6b	Name Surname	4.595	2023-11-08 00:05:58.713896	yes
3bfb23ae-d5d2-4847-9ace-66aea738a0e5	Name Surname	555	2023-11-08 00:06:08.740962	yes
1f5b8682-b271-426c-9289-678c6a08d03c	Name Surname	4.585	2023-11-08 00:06:23.800863	yes
74038687-ebad-449a-b964-3dc375f4c697	Name Surname	5.168	2023-11-08 00:07:11.614623	yes
737f4fcf-309a-4c2f-b138-f0ff80022247	Name Surname	4.658	2023-11-08 00:07:34.512461	yes
86368fe5-cb8a-41ac-b370-960574e42e09	Name Surname	547	2023-11-08 00:08:15.581685	yes
1b05d554-5ac2-417d-97af-f80e1f0fae97	Name Surname	5.825	2023-11-08 00:20:49.582306	yes
9f743fc2-8845-4605-a9d0-02c0a668e897	Name Surname	523	2023-11-08 00:23:46.567568	yes

Fonte: Elaborado pelo autor (2023).

6 CONSIDERAÇÕES FINAIS

O reconhecimento facial é um método biométrico que apresenta grande potencial de uso em contextos gerais, sobretudo na autenticação em eventos. O processo, além de não invasivo, possui grande rapidez na sua execução, sendo extremamente benéfico para os organizadores.

Para tal, o projeto desenvolvido apresentou êxito. Levando em consideração a testagem e os resultados obtidos, a quantidade de acertos é grande se comparado com a de erros. Além disso, a maneira que foi implementado, permite o acoplamento rápido de outras ferramentas, por exemplo, *web*. Além da facilidade na distribuição do aplicativo.

Existem outras tecnologias que podem trazer resultados ainda melhores, pois se beneficiam de uma base de treinamento maior, bem como maior suporte para suas operações. Destacam-se neste quesito o Amazon *Rekognition* e Azure *AI Face Service*. No entanto, ambos são disponibilizados gratuitamente somente por um período de tempo ou quantidade de usuários.

Para a presente aplicação, objetivou-se, desenvolver uma solução que tivesse um custo baixo e que aproveitasse recursos já existentes. Com esse intuito, o projeto foi inteiramente desenvolvido com ferramentas de código aberto, além de fazer uso de um aparelho celular que havia sido descartado. Aponta-se que a qualidade da câmera do dispositivo não é alta, nas amostras coletadas, ficou evidente a suavização aplicada nos rostos capturados.

Como medidas de expansão e transformação do produto, carinhosamente nomeado de WhoRU, deverá ser possível que os usuários façam seu próprio cadastro. Essa melhoria permitirá que mais dados sejam armazenados na base,

facilitando a autenticação caso o reconhecimento venha a falhar. Cita-se CPF e número de identidade, ambos utilizados para conferência de documentos.

Deve-se buscar informar ao usuário que o processo de reconhecimento está ocorrendo, como uma espécie de tela de carregamento. Pode-se trazer a imagem cadastrada para a tela a fim evidenciar quem é o usuário encontrado na base. Tem-se essa abordagem, pois busca-se o funcionamento inteiramente sem interação por parte dos indivíduos, sem a necessidade de confirmação. Para melhoria na performance desta funcionalidade, a conexão com um CDN (*Content Delivery Network*) é essencial.

Seguindo com as futuras implementações, a aplicação deverá contar com um cadastro de organizadores de eventos, que, por sua vez, deverão poder cadastrar esses. Essa implementação, além de permitir que os usuários sejam atrelados àquilo que pretendem comparecer, auxiliará o algoritmo de reconhecimento, pois ao invés de trabalhar com a base inteira, reduzirá seu escopo somente àqueles que estão ligados ao evento. Por exemplo, durante as testagens, reduzir-se-ia o número de referências para cerca de 30, em oposição aos quase 300. Este cenário também seria reproduzido em eventos de maior escala, tendo em vista que o número de cadastros na base tende a aumentar.

Considera-se que o projeto desenvolvido cumpriu com as metas estipuladas no começo de seu desenvolvimento, trazer uma nova abordagem, por sua vez confiável e acessível, ao reconhecimento facial. Também, foram superadas, com êxito, as dificuldades encontradas tanto nas linguagens de programação, bem como na integração das ferramentas escolhidas.

REFERÊNCIAS BIBLIOGRÁFICAS

ADJABI, Insaf; OUAHABI, Abdeldjalil; BENZAOUI, Amir; TALEB-AHMED, Abdelmalik. **Past, present, and future of face recognition: A review. Electronics**, v. 9, n. 8, p. 1188, 2020. MDPI.

ANWARUL, Shahina; DAHIYA, Susheela. **A Comprehensive Review on Face Recognition Methods and Factors Affecting Facial Recognition Accuracy.** Lecture Notes In Electrical Engineering, [S.L.], p. 495-514, 22 nov. 2019. Springer International Publishing. http://dx.doi.org/10.1007/978-3-030-29407-6_36.

BABICH, Aleksandra. **Biometric Authentication. Types of biometric identifiers**. 2012. 56 f. TCC (Graduação) - Curso de Business, Haaga-Helia University Of Applied Sciences, Helsínquia, 2012. Disponível em: https://www.theseus.fi/bitstream/handle/10024/44684/Babich_Aleksandra.pdf. Acesso em: 04 maio 2023.

BARRETT, Lindsey. **Ban facial recognition technologies for children-and for everyone else**. BUJ Sci. & Tech. L., v. 26, p. 223, 2020.

BEHAM, M. Parisa; ROOMI, S. Mohamed Mansoor. **A REVIEW OF FACE RECOGNITION METHODS**. International Journal Of Pattern Recognition And Artificial Intelligence, [S.L.], v. 27, n. 04, p. 1356005, jun. 2013. World Scientific Pub Co Pte Lt. http://dx.doi.org/10.1142/s0218001413560053.

BIERMAN, Gavin; ABADI, Martín; TORGERSEN, Mads. **Understanding typescript**. In: ECOOP 2014–Object-Oriented Programming: 28th European Conference, Uppsala, Sweden, July 28–August 1, 2014. Proceedings 28. Springer Berlin Heidelberg, 2014. p. 257-281.

BRADSKI, Gary; KAEHLER, Adrian**. Learning OpenCV: Computer vision with the OpenCV library**. " O'Reilly Media, Inc.", 2008.

BRASIL. [Constituição (2022)]. **EMENDA CONSTITUCIONAL Nº 115. 30**. ed. [S. l.: s. n.], 2022. 2 p. Disponível em: https://www.in.gov.br/web/dou/-/emenda-constitucional-n-115-379516387. Acesso em: 21 maio 2023.CHINNATHAMBI, Kirupa. Learning React. Boston: Addison-Wesley, 2017. 35 p.

Canonical LTD. **Enterprise Open Source and Linux**. 2023. Disponível em: https://ubuntu.com/. Acesso em: 17 nov. 2023.

CONGER, Kate; FAUSSET, Richard; KOVALESKI, Serge F.. **San Francisco Bans Facial Recognition Technology**. The New York Times. Nova York, p. 1-3. 14 maio 2019. Disponível em: https://www.beaude.net/traces/downloads/San%20Francisco%20Bans%20Facial%20Recognition%20Technology%20-%20The%20New%20York%20Times.pdf. Acesso em: 04 maio 2023.

COSTA, Luciano R.; OBELHEIRO, Rafael R.; FRAGA, Joni S. **Introdução à biometria**. Sociedade Brasileira de Computação, 2006.

DAUGMAN, John. New Methods in Iris Recognition. Ieee **Transactions On Systems, Man And Cybernetics**, Part B (Cybernetics), [S.L.], v. 37, n. 5, p. 1167-1175, out. 2007. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/tsmcb.2007.903540.

DLIB. **Dlib C++ Library**. 2022. Disponível em: http://dlib.net. Acesso em: 30 maio 2023.

ENG, S K; ALI, H; CHEAH, A y; CHONG, Y F. **Facial expression recognition in JAFFE and KDEF Datasets using histogram of oriented gradients and support vector machine**. Iop Conference Series: Materials Science and Engineering, [S.L.], v. 705, n. 1, p. 012031, 1 nov. 2019. IOP Publishing. http://dx.doi.org/10.1088/1757-899x/705/1/012031.
EXPO. Overview. 2023. Disponível em: https://docs.expo.dev/overview/. Acesso em: 04 jun. 2023.

FLANAGAN, David. **JavaScript: o guia definitivo**. 6. ed. Porto Alegre: Bookman, 2011. Tradução de: João Eduardo Nóbrega Tortello.

GALTERIO, Mary Grace; SHAVIT, Simi Angelic; HAYAJNEH, Thaier**. A review of facial biometrics security for smart devices**. Computers, v. 7, n. 3, p. 37, 2018.

GEITGEY, Adam. **Face recognition documentation**. Release, v. 1, n. 3, p. 3-37, 2018.

GOOGLE. **ML Kit**. 2023. Disponível em: https://developers.google.com/ml-kit/guides?hl=pt-br. Acesso em: 17 nov. 2023.

GOYANI, Mahesh M; PATEL, Narendra M. **Evaluation of various classifiers for expression recognition using multi level Haar features**. Int. J. Next-Gener. Comput., v. 9, n. 2, p. 131-150, 2018.

HAPSARI, Gita Indah; MUTIARA, Giva Andriana; TARIGAN, Husein. **Face recognition smart cane using haar-like features and eigenfaces**.

TELKOMNIKA (Telecommunication Computing Electronics and Control), v. 17, n. 2, p. 973-980, 2019.

HERRON, David. **Node Web Development: second edition**. 2. ed. Birmingham: Packt Publishing Ltd, 2013. 248 p.

IDRUS, Syed Zulkarnain Syed; CHERRIER, Estelle; ROSENBERGER, Christophe; SCHWARTZMANN, Jean-Jacques. **A Review on Authentication Methods**. A Review On Authentication Methods. S.I., p. 95-107. dez. 2013. Disponível em: https://hal.science/hal-00912435/document. Acesso em: 01 maio 2023.

JOSEFSSON, Simon. **The base16, base32, and base64 data encodings**. 2006.

KHAN, Sikandar; AKRAM, Adeel; USMAN, Nighat. **Real Time Automatic Attendance System for Face Recognition Using Face API and OpenCV**. Wireless Personal Communications, [S.L.], v. 113, n. 1, p. 469-480, 31 mar. 2020. Springer Science and Business Media LLC. http://dx.doi.org/10.1007/s11277-020-07224-2.

KING, Davis E. **Dlib-ml: A machine learning toolkit**. The Journal of Machine Learning Research, v. 10, p. 1755-1758, 2009.

KORTLI, Yassin; JRIDI, Maher; FALOU, Ayman Al; ATRI, Mohamed. **Face Recognition Systems: a survey**. Sensors, [S.L.], v. 20, n. 2, p. 342, 7 jan. 2020. MDPI AG. http://dx.doi.org/10.3390/s20020342.

KUMAR, Kunal; FARIK, Mohammed. **A Review Of Multimodal Biometric Authentication Systems**. International Journal Of Scientific & Technology Research. S.I., p. 5-9. dez. 2016. Disponível em: http://www.ijstr.org/final-print/dec2016/A-Review-Of-Multimodal-Biometric-Authentication-Systems.pdf. Acesso em: 01 maio 2023.

LI, Cuimei; QI, Zhiliang; JIA, Nan; WU, Jianhua. **Human face detection algorithm via Haar cascade classifier combined with three additional classifiers**. In: 13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI), 2017, p. 483-487. IEEE.

LI, Zewen et al. **A survey of convolutional neural networks: analysis, applications, and prospects**. IEEE transactions on neural networks and learning systems, 2021.

LTD., Greenwood Campbell. **We learn and we teach**. 2023. Disponível em: https://www.thehumantech.agency/learn. Acesso em: 19 nov. 2023.

LYNCH, Stephen. **Python for Scientific Computing and Artificial Intelligence**. CRC Press, 2023.

MAGALHÃES, Paulo Sérgio Tenreiro; SANTOS, Henrique Dinis dos. **Biometria e autenticação**. 2003.

MAKHSUD, Usmonov. **Identification and Authentication**. International Journal Of Academic Pedagogical Research (Ijapr). [S. L.], p. 39. jan. 2021. Disponível em: http://ijeais.org/wp-content/uploads/2021/1/IJAPR210108.pdf. Acesso em: 15 maio 2023.

MALIK, Jyoti; GIRDHAR, Dhiraj; DAHIYA, Ratna; SAINARAYANAN, G.. **Reference Threshold Calculation for Biometric Authentication**. International Journal Of Image, Graphics And Signal Processing, [S.L.], v. 6, n. 2, p. 46-53, 8 jan. 2014. MECS Publisher. http://dx.doi.org/10.5815/ijigsp.2014.02.06. Disponível em: http://www.mecs-press.net/ijigsp/ijigsp-v6-n2/IJIGSP-V6-N2-6.pdf. Acesso em: 02 maio 2023.

MATTHES, Eric. **Python Crash Course: a hands-on, project - based introduction to programming**. San Francisco: No Starch Press, 2016. 562 p.

META PLATFORMS, INC. **React Native**. [S. l.], 2023. Disponível em: https://reactnative.dev/. Acesso em: 1 jun. 2023.

MORESI, Eduardo et al. **Metodologia da pesquisa**. Brasília: Universidade Católica de Brasília, v. 108, n. 24, p. 5, 2003.

MULYONO, Ibnu Utomo Wahyu; SUSANTO, Ajib; RACHMAWANTO, Eko Hari; FAHMI, Amiq; SETIADI, De Rosal Ignatius Moses; MULJONO. **Performance analysis of face recognition using eigenface approach**. In: 2019 International Seminar on Application for Technology of Information and Communication (iSemantic), 2019, p. 1-5. IEEE.

OPENCV. **Introduction**. Disponível em: https://docs.opencv.org/4.x/d1/dfb/intro.html. Acesso em: 29 maio 2023.
OPENJS FOUNDATION. About Node.js. 2023. Disponível em: https://nodejs.org/en/about. Acesso em: 03 jun. 2023.

PHILLIPS, P. Jonathon; YATES, Amy N.; HU, Ying; HAHN, Carina A.; NOYES, Eilidh; JACKSON, Kelsey; CAVAZOS, Jacqueline G.; JECKELN, Géraldine; RANJAN, Rajeev; SANKARANARAYANAN, Swami. **Face recognition accuracy of forensic examiners, superrecognizers, and face recognition algorithms**. Proceedings Of The National Academy Of Sciences, [S.L.], v. 115, n. 24, p. 6171-6176, 29 maio 2018. Proceedings of the National Academy of Sciences. http://dx.doi.org/10.1073/pnas.1721355115.

POSTMAN INC. **API platform overview**. 2023. Disponível em: https://www.postman.com/api-platform/. Acesso em: 03 jun. 2023.

PULLI, Kari; BAKSHEEV, Anatoly; KORNYAKOV, Kirill; ERUHIMOV, Victor. **Realtime Computer Vision with OpenCV**. Queue, [S.L.], v. 10, n. 4, p. 40-56, abr. 2012. Association for Computing Machinery (ACM). http://dx.doi.org/10.1145/2181796.2206309.

PYTHON. **MovingToPythonFromOtherLanguages**. 2023. Disponível em: https://wiki.python.org/moin/MovingToPythonFromOtherLanguages. Acesso em: 29 maio 2023.

PYTHON. **Overview**. Disponível em: https://wiki.python.org/moin/BeginnersGuide/Overview. Acesso em: 29 maio 2023.

PYTHON. **Pickle — Python object serialization**. 2023. Disponível em: https://docs.python.org/3/library/pickle.html. Acesso em: 22 nov. 2023.

RAMÍREZ, Sebastián. **FastAPI**. 2023. Disponível em: https://fastapi.tiangolo.com. Acesso em: 29 maio 2023.

RAO, Dwadasi Priyamvada; TAUNK, Shraddha; KANSARA, Nirali; GUPTA, Apurva. **Facial Detection and Recognition System Using Python**. International Journal Of Futuristic Innovation In Engineering, Science And Technology (Ijfiest), [S.L.], v. 1, n. 1, p. 16-18, 20 set. 2022. Inence Publications Pvt Ltd. http://dx.doi.org/10.59367/ijfiest.v1i1.9.

SANTOS, João Almeida dos; PARRA FILHO, Domingos. **Metodologia Científica**. 2. ed. São Paulo: Cengage Learning Ltda., 2012. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9788522112661. Acesso em: 11 jun. 2023.

TOLBA, A. S.; EL-BAZ, A. H.; EL-HARBY, A. A. **Face recognition: A literature review**. International Journal of Signal Processing, v. 2, n. 2, p. 88-103, 2006.

VADLAPATI, Jayanth; VELAN, S Senthil; VARGHESE, Ewin. **Facial Recognition using the OpenCV Libraries of Python for the Pictures of Human Faces Wearing Face Masks during the COVID-19 Pandemic**. 2021 12Th International Conference On Computing Communication And Networking Technologies (Icccnt), [S.L.], p. 1-5, 6 jul. 2021. IEEE. http://dx.doi.org/10.1109/icccnt51525.2021.9579712.

WIRFS-BROCK, Allen; EICH, Brendan. **JavaScript: the first 20 years**. Proceedings Of The Acm On Programming Languages, [S.L.], v. 4, n. , p. 1-189, 12 jun. 2020. Association for Computing Machinery (ACM). http://dx.doi.org/10.1145/3386327.

WOODWARD JUNIOR, John D.; HORN, Christopher; GATUNE, Julius; THOMAS, Aryn. **Biometrics: a look at facial recognition**. Arlington: Rand,

2003. 31 p. Disponível em: https://apps.dtic.mil/sti/pdfs/ADA414520.pdf. Acesso em: 04 maio 2023.

WU, Wenhao. **React Native vs Flutter, cross-platform mobile application frameworks**. 2018. 28 f. TCC (Doutorado) - Curso de Bachelor Of Engineering, Information Technology, Metropolia University Of Applied Sciences, Helsínquia, 2018.

ZANELLA, Liane Carly Hermes et al. **Metodologia da pesquisa**. SEAD/UFSC, 2006.