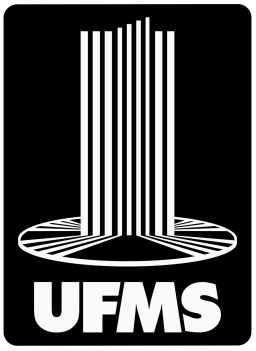


UNIVERSIDADE FEDERAL
DE MATO GROSSO DO SUL

FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA

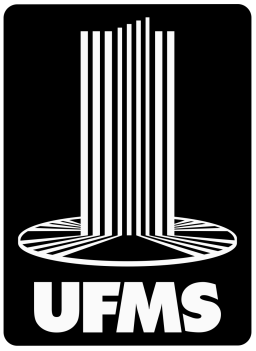
CURSO DE ENGENHARIA ELÉTRICA

INTEGRAÇÃO DE INTELIGÊNCIA ARTIFICIAL E SOLUÇÕES IOT PARA AUTOMATIZAÇÃO DOS SERVIÇOS DE EMPRÉSTIMO DE BICICLETAS ELÉTRICAS

HENRY FELIPE MAGRI

Campo Grande - MS

2024



UNIVERSIDADE FEDERAL
DE MATO GROSSO DO SUL

FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA

CURSO DE ENGENHARIA ELÉTRICA

INTEGRAÇÃO DE INTELIGÊNCIA ARTIFICIAL E SOLUÇÕES IOT PARA AUTOMATIZAÇÃO DOS SERVIÇOS DE EMPRÉSTIMO DE BICICLETAS ELÉTRICAS

HENRY FELIPE MAGRI

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharelado em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul - UFMS.

Orientador: Prof. Dr. Ruben Barros Godoy

Campo Grande - MS

2024

INTEGRAÇÃO DE INTELIGÊNCIA ARTIFICIAL E SOLUÇÕES IOT PARA AUTOMATIZAÇÃO DOS SERVIÇOS DE EMPRÉSTIMO DE BICICLETAS ELÉTRICAS

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharelado em Engenharia Elétrica da Universidade Federal de Mato Grosso do Sul – UFMS.

AUTOR: Henry Felipe Magri

ORIENTADOR: Prof. Dr. Ruben Barros Godoy

Banca Examinadora

_____

Prof. Dr. Ruben Barros Godoy

_____

Prof. Dr. Edson Antonio Batista

_____

Prof. Cristiano Quevedo Andrea

Campo Grande - MS, 2024.

"*Engenheiros gostam de resolver problemas. Se não há problemas à vista, eles criam seus próprios problemas.*"

(Scott Adams)

"*Não espere o futuro mudar sua vida. Porque o futuro será a consequência do presente.*"

(Racionais MC's)

Agradecimentos

Gostaria de expressar minha profunda gratidão a todos aqueles que contribuíram para a realização desta etapa da minha vida.

Aos meus pais, Luis Carlos Magri e Gislaine Cristina Nogueira, e ao meu irmão, Davi Miguel Magri, meu eterno agradecimento por todo amor, incentivo e apoio incondicional que sempre me ofereceram. Vocês foram a base sólida que me permitiu seguir com minhas escolhas com segurança. Aos meus avós, tias e primos, que sempre me motivaram. Sem o suporte de todos vocês, essa conquista não teria sido possível.

Também sou grato a todas as amizades que carrego comigo, sejam aquelas cultivadas durante a faculdade ou desde os tempos de escola. A presença de cada um, seja nas risadas compartilhadas presencialmente ou nos momentos de distância, especialmente nos últimos cinco anos, fez toda a diferença. Agradeço por todos os momentos que me proporcionaram leveza e alegria durante essa jornada.

Meu agradecimento se estende a todos os professores que tive o privilégio de encontrar ao longo da graduação, em especial ao meu orientador, Dr. Ruben Barros Godoy, pelo direcionamento e paciência ao longo do projeto, e ao professor Dr. Cristiano Quevedo Andrea, que primeiro despertou em mim o interesse pela engenharia elétrica. Reconheço, também, todos os demais docentes que compartilharam conhecimentos e oportunidades, sempre encorajando meu desenvolvimento.

A cada pessoa que fez parte dessa trajetória, deixo aqui meu mais sincero e profundo agradecimento.

Resumo

Este trabalho propõe a criação de uma plataforma automatizada para o empréstimo de bicicletas elétricas, integrando tecnologias de Inteligência Artificial (IA) e Internet das Coisas (IoT) para simplificar o uso e aumentar a segurança deste serviço. Utilizando um sistema de reconhecimento facial baseado em IA e conectado a um robô de WhatsApp, os usuários podem alugar bicicletas sem a necessidade de aplicativos adicionais. A revisão bibliográfica explora as principais arquiteturas de IA avaliadas para o projeto, incluindo a *Fully Convolutional Networks* (FCN), Mask R-CNN, U-Net e DeepLabV3+ para segmentação, e ResNet50, EfficientNetB3, Visual Geometry Group (VGG) e MobileNet para classificação de imagens, destacando suas aplicações e vantagens. A metodologia abrange o desenvolvimento do sistema em Python, com o uso de *frameworks* como FastAPI e bibliotecas de IA (TensorFlow, Scikit-learn, OpenCV e Keras) que utiliza um modelo robusto treinado para segmentação de faces e treina modelos personalizados de classificação para cada usuário. Os resultados apontam que a arquitetura de segmentação DeepLabV3+, que obteve 94,64% de precisão e 12,19% de perdas durante a validação, e uma assertividade geral de 93%, e o modelo de classificação ResNet50, que obteve 100% de precisão e 0,66% de perdas durante a validação, e manteve uma assertividade de 98% a 100% entre os modelos treinados para diferentes faces, ofereceram o melhor desempenho em termos de precisão e eficiência para o reconhecimento facial. O sistema principal foi capaz de realizar processos como o treinamento dos modelos personalizados dos usuários em uma média de 17 minutos, durante o cadastro, processo de validação em uma média de 10 segundos e processos secundários em 2 segundos. A integração entre o robô de WhatsApp e a API do sistema principal mostrou-se eficaz para gerenciar o fluxo de dados dos usuários, atendendo aos requisitos de segurança e praticidade do serviço. Conclui-se que o sistema desenvolvido contribui para a automatização segura e simplificada de serviços de mobilidade urbana, com potencial para futuras expansões em outras áreas de automação.

Palavras Chave: Inteligência Artificial (IA), Internet das Coisas (IoT), Serviços de Aluguel, Micromobilidade Urbana, Robô de WhatsApp.

Abstract

This work proposes the creation of an automated platform for electric bicycle rentals, integrating Artificial Intelligence (AI) and Internet of Things (IoT) technologies to simplify use and enhance the security of this service. By using an AI-based facial recognition system connected to a WhatsApp bot, users can rent bicycles without the need for additional apps. The literature review explores the main AI architectures evaluated for the project, including Fully Convolutional Networks (FCN), Mask R-CNN, U-Net, and DeepLabV3+ for segmentation, and ResNet50, EfficientNetB3, Visual Geometry Group (VGG), and MobileNet for image classification, highlighting their applications and advantages. The methodology covers the development of the system in Python, using frameworks like FastAPI and AI libraries (TensorFlow, Scikit-learn, OpenCV, and Keras), employing a robust model trained for face segmentation and training customized classification models for each user. The results show that the DeepLabV3+ segmentation architecture, with 94.64% accuracy and 12.19% loss during validation, and an overall accuracy of 93%, and the ResNet50 classification model, which achieved 100% accuracy and 0.66% loss during validation, maintaining an accuracy of 98% to 100% among models trained for different faces, offered the best performance in terms of accuracy and efficiency for facial recognition. The main system was able to perform processes such as training customized user models in an average of 17 minutes during registration, validation processes in an average of 10 seconds, and secondary processes in 2 seconds. The integration between the WhatsApp bot and the main system's API proved effective in managing user data flow, meeting the security and practicality requirements of the service. It is concluded that the developed system contributes to the secure and simplified automation of urban mobility services, with potential for future expansions into other areas of automation.

Keywords: Artificial Intelligence (AI), Internet of Things (IoT), Rental Services, Urban Micromobility, WhatsApp Bot.

Lista de ilustrações

Lista de tabelas

Lista de abreviamentos e siglas

IoT - *Internet of Things* (Internet das Coisas)

IA - Inteligência Artificial

BATLAB - Laboratório de Inteligência Artificial, Eletrônica de Potência e Sistemas Digitais da UFMS

UFMS - Universidade Federal de Mato Grosso do Sul

API - *Application Programming Interface* (Interface de Programação de Aplicações)

HTTP - *HyperText Transfer Protocol* (Protocolo de Transferência de Hipertexto)

M2M - *Machine to Machine* (Máquina para Máquina)

AM - Aprendizado de Máquina

AP - Aprendizado Profundo

CNN - *Convolutional Neural Network* (Rede Neural Convolucional)

RNN - *Recurrent Neural Network* (Rede Neural Recorrente)

FCN - *Fully Convolutional Network* (Rede Totalmente Convolucional)

VOC - *Visual Object Classes* (Classes de Objetos Visuais, um conjunto de dados popular para detecção de objetos)

NYU - *New York University* (Universidade de Nova York)

SIFT - *Scale-Invariant Feature Transform* (Transformada Invariante de Escala para Extração de Características)

R-CNN - *Regions with Convolutional Neural Networks* (Regiões com Redes Neurais Convolucionais)

YOLO - *You Only Look Once* (Um modelo de detecção de objetos)

NET - *Network* (Rede)

ISBI - *International Symposium on Biomedical Imaging* (Simpósio Internacional de Imagens Biomédicas)

ASPP - *Atrous Spatial Pyramid Pooling* (Agrupamento Piramidal Espacial com Atrous)

ResNet - *Residual Network* (Rede Residual)

ILSVRC - *ImageNet Large Scale Visual Recognition Challenge* (Desafio de Reconhecimento Visual em Grande Escala do ImageNet)

VGG - *Visual Geometry Group* (Grupo de Geometria Visual, refere-se a uma arquitetura de redes neurais desenvolvida por esse grupo)

ROC-curve - *Receiver Operating Characteristic Curve* (Curva Característica de Operação do Receptor)

2D - Duas Dimensões

3D - Três Dimensões

FEI - Fundação Educacional Inaciana

JSON - *JavaScript Object Notation* (Notação de Objetos JavaScript)

PNG - *Portable Network Graphics* (Gráficos de Rede Portátil)

CPF - Cadastro de Pessoa Física

CEP - Código de Endereçamento Postal

Sumário

1 Introdução

Nos últimos anos, o avanço das tecnologias de reconhecimento facial e de Internet das Coisas (IoT) tem impulsionado a criação de soluções inovadoras para diversos setores, incluindo mobilidade urbana. O aluguel ou empréstimo de bicicletas elétricas, uma alternativa sustentável e prática para o transporte em grandes centros urbanos, tem se beneficiado dessas inovações, permitindo um serviço mais seguro, eficiente e acessível, além de se tornar a base de muitos modelos de negócios emergentes. O conceito de IoT, pode ser compreendido como uma rede global de objetos físicos distintos interconectados por meio de sensores sem fio e identificadores de radiofrequência, e esta é considerada a próxima evolução da internet [1]. Entretanto, a maior parte das soluções disponíveis atualmente no mercado ainda depende de interações manuais, como a utilização de aplicativos móveis complexos para liberar os serviços de empréstimo ou aluguel, realizar pagamentos e resolver problemas. Além disso, a segurança e o controle de identidade em tais sistemas são questões desafiadoras, visto que o uso de senhas e códigos de acesso pode ser vulnerável a fraudes ou acessos indevidos.

Diante desse cenário, este trabalho propõe o desenvolvimento de uma plataforma automatizada para o empréstimo de bicicletas elétricas, utilizando inteligência artificial para reconhecimento facial, integrando o serviço a um sistema de IoT e oferecendo uma interface amigável via WhatsApp, que tem por finalidade evitar a necessidade da instalação de aplicativos adicionais. O objetivo principal é eliminar a necessidade de interações manuais e aumentar a segurança do processo de retirada, através de um sistema de reconhecimento facial personalizado para cada usuário, onde cada indivíduo possui um modelo treinado exclusivamente para sua identificação.

O projeto está sendo desenvolvido pelo BATLAB - Laboratório de Inteligência Artificial, Eletrônica de Potência e Sistemas Digitais da UFMS, onde atualmente existem docas instaladas tanto no eletroposto da universidade (localizado no campus de Campo Grande) quanto no próprio laboratório. Esses locais servem como pontos de retirada das bicicletas pelos usuários. A interação do usuário com o serviço é realizada através de um robô de WhatsApp, que permite ao usuário realizar o cadastro, alugar bicicletas, reportar defeitos e alterar dados cadastrais, de forma guiada e intuitiva. O sistema foi implementado e está sendo

validado em um grupo de aproximadamente 50 estudantes, em sua maioria da engenharia elétrica.

As informações coletadas durante o processo de cadastro são utilizadas para treinar redes neurais de classificação, responsável pela identificação do usuário. Duas APIs, desenvolvidas em Python com FastAPI, orquestram o treinamento dos modelos de Inteligência Artificial (IA) e gerenciam a comunicação com o usuário via WhatsApp, utilizando o serviço Twilio. O uso de robôs de *chat* é de extrema importância para disseminar informações de forma aceitável e direta para grandes blocos de usuários ao mesmo tempo, sobretudo em localidades onde existem barreiras técnicas, políticas ou de segurança, e o uso de VPN ainda é muito caro [2]. Nesse contexto, a extrapolação desta ferramenta para outras aplicações envolvendo grandes massas se torna uma solução mais prática, quando comparado com o desenvolvimento de aplicativos móveis.

Além disso, o sistema oferece a possibilidade de retreinamento dos modelos de reconhecimento facial caso o usuário sofra alterações significativas em sua aparência, e mecanismos para redefinição de senha, utilizada no processo de retreinamento dos modelos, e atualização de dados através de um código enviado por e-mail. Esse conjunto de funcionalidades visa não apenas otimizar a experiência de empréstimo de bicicletas elétricas, mas também proporcionar uma plataforma segura e de fácil uso, que pode ser adaptada a outros serviços de mobilidade e acesso automatizado.

Este trabalho tem como propósito apresentar o desenvolvimento desta solução, seus desafios, escolhas tecnológicas e potenciais impactos, tanto no setor de mobilidade urbana quanto em outras áreas que possam beneficiar-se da automação de processos baseados em reconhecimento facial.

2 Revisão bibliográfica

A evolução das tecnologias de Inteligência Artificial (IA) tem proporcionado avanços significativos em diversas áreas, especialmente no reconhecimento e segmentação de imagens. Essas arquiteturas de IA são essenciais para a identificação de padrões e a análise de dados visuais, sendo amplamente aplicadas em sistemas de segurança, reconhecimento facial e automação. Além disso, os conceitos de Aprendizado de Máquina (AM) e Aprendizado Profundo (AP) são fundamentais para compreender como os sistemas aprendem com os dados. Enquanto o AM permite que algoritmos se aprimorem automaticamente, o AP utiliza redes neurais profundas para resolver problemas complexos, possibilitando tarefas como a tradução automática e a geração de conteúdo.

Um estudo, revisado inicialmente, de 2014 sobre detecção e reconhecimento de faces utilizando MATLAB, demonstrou o potencial do uso de IA no reconhecimento facial, superando métodos convencionais testados à época. Esse trabalho impulsionou a busca por ferramentas e arquiteturas de IA mais atuais e robustas, capazes de oferecer maior precisão e eficiência em tarefas de segmentação e reconhecimento facial, conforme proposto neste projeto [3].

A integração da IA com a IoT também se destaca, conectando dispositivos e permitindo a coleta de dados em tempo real para otimizar processos e aumentar a eficiência. Essa interconexão é crucial para o desenvolvimento de soluções inteligentes que atendam a diversas demandas. Esta revisão bibliográfica, portanto, aborda três áreas principais — arquiteturas de IA, conceitos de AM e AP, e as aplicações da IoT — com o objetivo de estabelecer uma base teórica robusta para o desenvolvimento e implementação do trabalho em questão.

2.1 IoT: Internet das Coisas

A IoT é um conceito que se refere à interconexão de objetos físicos equipados com sensores, *software* e outras tecnologias que lhes permitem coletar e compartilhar dados pela internet. A IoT transforma objetos comuns em dispositivos inteligentes, permitindo que eles

se comuniquem entre si, processem informações e tomem decisões de forma autônoma. Desde sensores em carros e eletrodomésticos até grandes sistemas industriais, a IoT tem se expandido em diversas áreas, possibilitando melhorias na eficiência operacional, automação e análise de dados [4,5].

Para que os dispositivos conectados possam interagir eficientemente na IoT, são amplamente utilizadas *Application Programming Interfaces* (APIs), que atuam como intermediárias entre diferentes sistemas e dispositivos, permitindo a troca de informações de forma estruturada e segura. As APIs definem um conjunto de regras e protocolos que permitem a comunicação entre os dispositivos IoT e servidores ou outros dispositivos, garantindo que os dados possam ser enviados e recebidos sem que os sistemas envolvidos precisem conhecer o funcionamento interno uns dos outros.

As APIs geralmente trabalham com diferentes tipos de requisições *Hypertext Transfer Protocol* (HTTP), como *POST* e *GET*. A requisição *GET* é usada para solicitar dados de um servidor, enquanto a requisição *POST* é utilizada para enviar dados para serem processados ou armazenados. Por exemplo, em uma aplicação IoT, um sensor pode fazer uma requisição *POST* para enviar os dados de temperatura para um servidor, que processa essa informação e a armazena em um banco de dados. Em contrapartida, um aplicativo pode realizar uma requisição *GET* para consultar esses dados armazenados e exibi-los ao usuário final. Essas interações permitem que dispositivos IoT compartilhem dados em tempo real e façam integrações eficientes com sistemas maiores, como plataformas de análise ou automação [6].

Nos últimos anos, a IoT cresceu significativamente, em parte devido aos avanços em tecnologias de comunicação sem fio, identificadores de radiofrequência, e computação em nuvem, que permitem a troca eficiente de dados entre dispositivos conectados. Além disso, a adoção crescente de Máquina-a-Máquina (M2M) tem permitido que máquinas se comuniquem diretamente entre si, impulsionando o desenvolvimento da automação industrial e da infraestrutura urbana inteligente [5].

As tendências indicam que a IoT continuará a desempenhar um papel cada vez mais central na transformação digital de diversas indústrias. De acordo com previsões, o número de dispositivos conectados poderá alcançar dezenas de bilhões até 2025, abrangendo desde eletrodomésticos inteligentes até sistemas de transporte e cidades inteligentes. Essa tecnologia

tem o potencial de promover decisões mais rápidas e precisas em setores como saúde, transporte e manufatura [4,5].

Apesar do crescimento, a IoT enfrenta desafios significativos, como questões de segurança e privacidade dos dados, além da necessidade de interoperabilidade entre dispositivos de diferentes fabricantes. Esses desafios são amplamente discutidos na literatura, destacando a necessidade de desenvolvimento contínuo de protocolos e padrões que assegurem a operação eficiente e segura de redes IoT em larga escala [4].

2.2 Aprendizado de Máquina e Aprendizado Profundo

Aprendizado de Máquina (AM) é um campo da inteligência artificial onde sistemas de computador aprendem a realizar tarefas a partir de dados, sem a necessidade de serem explicitamente programados. O aprendizado ocorre por meio da identificação de padrões nos dados, criando modelos capazes de realizar previsões ou tomar decisões. Existem três principais tipos de aprendizado no AM: supervisionado, não supervisionado e por reforço. No aprendizado supervisionado, o modelo é treinado com base em dados rotulados, ou seja, cada entrada tem uma saída correspondente conhecida. Já o aprendizado não supervisionado trabalha com dados não rotulados, buscando padrões ou agrupamentos naturais nos dados. No aprendizado por reforço, o sistema aprende através de tentativa e erro, recebendo recompensas ou penalidades a cada ação, permitindo que otimize sua performance ao longo do tempo [7,8].

Aprendizado Profundo (AP), por outro lado, é um subconjunto do aprendizado de máquina que utiliza redes neurais profundas para realizar tarefas complexas. Diferente do AM tradicional, o aprendizado profundo não depende de extração manual de características, pois a rede neural aprende diretamente a partir dos dados brutos, como imagens ou textos. Isso é possível devido às múltiplas camadas de processamento que permitem ao modelo aprender representações em diferentes níveis de abstração. Redes neurais convolucionais (CNNs) e redes neurais recorrentes (RNNs) são alguns exemplos de arquiteturas de aprendizado profundo amplamente utilizadas. O AP é especialmente eficaz em tarefas que envolvem

grandes volumes de dados e problemas mais complexos, como reconhecimento de voz, segmentação de imagens e processamento de linguagem natural [8].

Enquanto o aprendizado de máquina tradicional é adequado para tarefas com conjuntos de dados estruturados e com extração manual de características, o AP se destaca em problemas que exigem o processamento de dados não estruturados em larga escala, como imagens e vídeos. Assim, o AP é amplamente utilizado em áreas que demandam alta precisão, como reconhecimento facial, carros autônomos e diagnósticos médicos [7,8].

2.3 Arquiteturas de Segmentação Estudadas

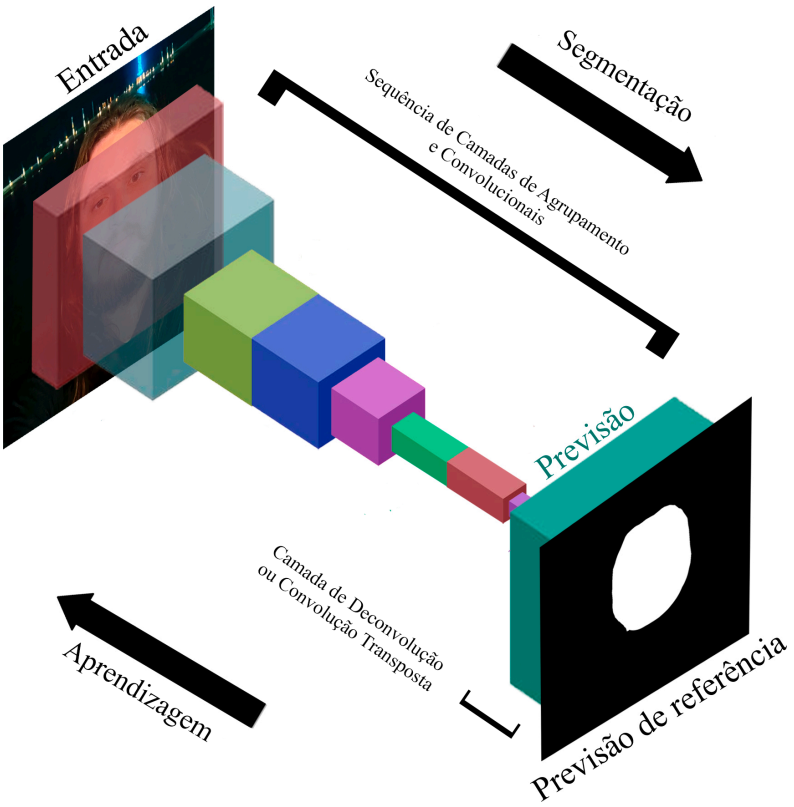
Este tópico apresenta uma revisão bibliográfica das arquiteturas de segmentação semântica estudadas e utilizadas neste trabalho. A segmentação de imagens é uma das tarefas fundamentais no campo da visão computacional, onde o objetivo é classificar cada pixel de uma imagem de acordo com a classe à qual ele pertence, permitindo a detecção precisa de objetos e suas fronteiras. Diferente de métodos de classificação tradicionais, que atribuem uma única classe a uma imagem inteira, a segmentação de imagens trata cada pixel individualmente, o que é essencial para aplicações como detecção de objetos, reconhecimento facial, diagnóstico por imagem, entre outras áreas [9].

2.3.1 *Fully Convolutional Networks*

A *Fully Convolutional Network* (FCN) é uma arquitetura de rede neural projetada para tarefas de segmentação semântica, onde cada pixel de uma imagem é classificado em uma categoria específica. Diferente das redes convolucionais tradicionais, a FCN se destaca por remover as camadas densamente conectadas que são comumente usadas em classificadores de imagens, substituindo-as por camadas convolucionais, que simulam o funcionamento de células visuais no cérebro, extraindo características importantes das imagens. Isso permite que a rede processe imagens de qualquer tamanho, fornecendo uma saída que mantém a correspondência espacial com a imagem de entrada [9].

O funcionamento da FCN baseia-se em uma sequência de camadas convolucionais que extraem características das imagens, seguidas de camadas de agrupamento, que reduzem progressivamente a dimensão espacial dos dados. Para restaurar a resolução original da imagem, a FCN utiliza camadas de deconvolução ou convoluções transpostas, que realizam o aumento da resolução, ampliando os mapas de características gerados nas camadas convolucionais para recuperar os detalhes espaciais perdidos durante a redução da resolução. Isso permite que a saída da rede tenha a mesma resolução da imagem de entrada, algo crucial para a precisão da segmentação semântica [9]. Esses processos estão exemplificados na Figura 1.

Figura 1 - Arquitetura FCN



Fonte: Elaborada pelo autor.

Além disso, a FCN introduz o conceito de pular conexões, que conectam camadas mais profundas da rede com camadas mais superficiais. Essa técnica permite combinar informações de alto nível, capturadas nas camadas mais profundas, com detalhes finos presentes nas camadas iniciais, resultando em uma segmentação mais precisa. Essas conexões ajudam a manter a precisão espacial ao longo do processo, melhorando a definição das bordas e detalhes dos objetos segmentados [9].

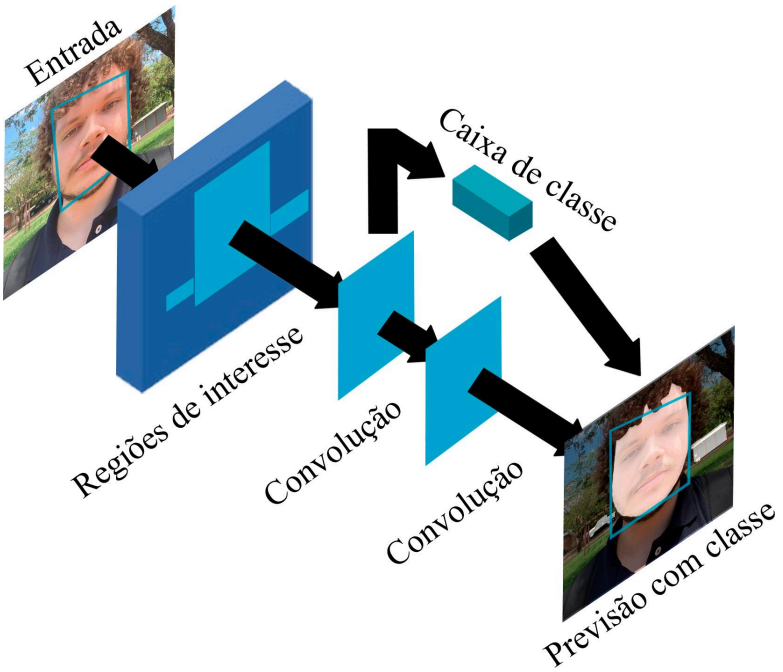
A FCN demonstrou resultados expressivos em *benchmarks* como o PASCAL *Visual Object Classes* (VOC), onde obteve melhorias significativas na segmentação de objetos, superando métodos tradicionais de segmentação. Além disso, sua arquitetura é versátil e pode ser aplicada em diversos outros conjuntos de dados de segmentação, como *New York University* (NYU) *Depth* V2 e *Scale-Invariant Feature Transform* (SIFT) *Flow*, sendo uma das arquiteturas pioneiras no uso de redes profundas para segmentação semântica [9].

Essa arquitetura abriu caminho para o desenvolvimento de várias outras abordagens mais recentes e avançadas em segmentação, devido à sua simplicidade e eficiência no uso de camadas totalmente convolucionais para preservação das informações espaciais durante o processamento [9].

2.3.2 *Mask Region-based Convolutional Neural Network*

A *Region-based Convolutional Neural Network* (R-CNN) é uma arquitetura voltada para a detecção e segmentação de objetos em imagens. Seu princípio central envolve o uso de Propostas Regionais, onde são geradas regiões de interesse que podem conter objetos, e em seguida, uma rede neural é aplicada para classificar e segmentar essas regiões. A *Mask* R-CNN, uma versão aprimorada da *Faster* R-CNN, adiciona uma terceira ramificação que prevê máscaras de segmentação, além de classificar e ajustar as caixas delimitadoras dos objetos [10]. O funcionamento do *Framework Mask* R-CNN está exemplificado na Figura 2.

Figura 2 - *Framework Mask* R-CNN



Fonte: Elaborada pelo autor.

A arquitetura *Mask* R-CNN realiza segmentação de instâncias ao prever máscaras para cada objeto identificado, além de reconhecer suas classes e localizações exatas. Para fazer isso, ela utiliza o método de alinhamento de regiões de interesse, que corrige o problema de alinhamento impreciso causado pelo agrupamento dessas regiões, garantindo que a extração de características seja espacialmente precisa, o que melhora a qualidade das máscaras de segmentação [10,11].

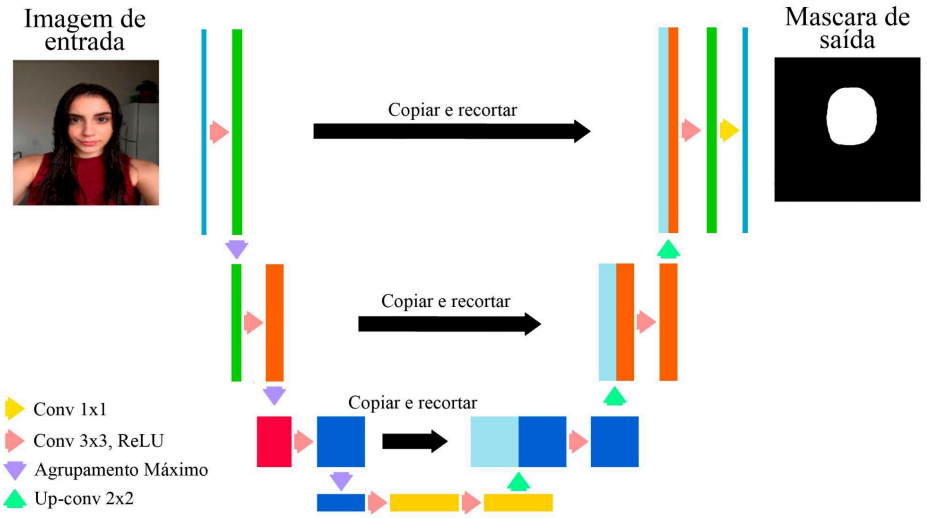
Embora a *Mask* R-CNN seja altamente eficaz na segmentação de múltiplas instâncias dentro de uma mesma imagem, o custo dessa precisão é uma maior complexidade computacional, o que torna o modelo mais lento em comparação com outras abordagens, como o *You Only Look Once* (YOLO), que foca na detecção em tempo real. A *Mask* R-CNN, por exemplo, pode processar uma imagem em cerca de 200 milissegundos, enquanto o YOLO pode fazer o mesmo em poucos milissegundos, sacrificando um pouco de precisão em favor da velocidade [11].

Em resumo, a *Mask* R-CNN é uma excelente escolha quando a precisão na segmentação de várias instâncias é essencial, mas apresenta uma desvantagem em termos de velocidade, sendo menos indicada para aplicações em tempo real sem otimizações adicionais [11].

2.3.3 U-Net

A U-Net foi projetada para segmentação semântica de imagens, com foco inicial em aplicações biomédicas. A arquitetura segue uma estrutura simétrica em forma de "U", composta por dois caminhos: um de contração e um de expansão, como mostrado na Figura 3. O caminho de contração funciona como um codificador, realizando convoluções seguidas por *pooling* para reduzir a dimensionalidade e extrair características de alto nível. Já o caminho de expansão atua como um decodificador, onde as operações de aumento de resolução (convoluções transpostas) restauram a resolução original da imagem [12].

Figura 3 - Arquitetura U-NET



Fonte: Elaborada pelo autor.

Uma característica dominante da U-Net é a concatenação entre as camadas correspondentes dos caminhos de contração e expansão. Essa combinação permite que informações de alta resolução (da fase de redução da resolução) sejam recuperadas e preservadas, o que é essencial para garantir uma segmentação precisa, especialmente em imagens com detalhes finos e complexos, como aquelas de estruturas celulares [12].

A U-Net também se destaca pela sua capacidade de generalização com dados limitados, o que é essencial em áreas como a biomédica, onde é difícil obter grandes quantidades de dados anotados manualmente. Para mitigar essa limitação, a U-Net faz uso de técnicas extensivas de aumento de dados, como rotações, espelhamentos e deformações elásticas, para aumentar a variabilidade dos dados de treinamento e melhorar a robustez do modelo [12].

Aplicações da U-Net em desafios de segmentação biomédica mostraram resultados impressionantes. No desafio *International Symposium on Biomedical Imaging* (ISBI) de segmentação celular, por exemplo, a U-Net superou outras abordagens ao oferecer precisão superior na segmentação de células em imagens microscópicas, consolidando-se como uma das arquiteturas mais eficientes para tarefas de segmentação em contextos com variação de escala e textura [12].

Essas características tornam a U-Net uma escolha amplamente adotada em diversos campos que exigem segmentação precisa e eficiente, especialmente quando o número de dados disponíveis para treinamento é limitado [12].

2.3.4 DeepLabV3+

A arquitetura DeepLabV3+ se destaca no campo da segmentação semântica, tendo sido desenvolvida para resolver desafios significativos que dificultam as tarefas de previsão densas. Seu funcionamento incorpora convoluções dilatadas, que expandem o campo de visão dos filtros, permitindo a captura de informações contextuais em múltiplas escalas, sem aumentar o custo computacional. O modelo foi projetado para enfrentar três desafios principais: a resolução de recursos reduzida, resultante da combinação repetida de operações de agrupamento máximo e redução da resolução em camadas sequenciais; a presença de

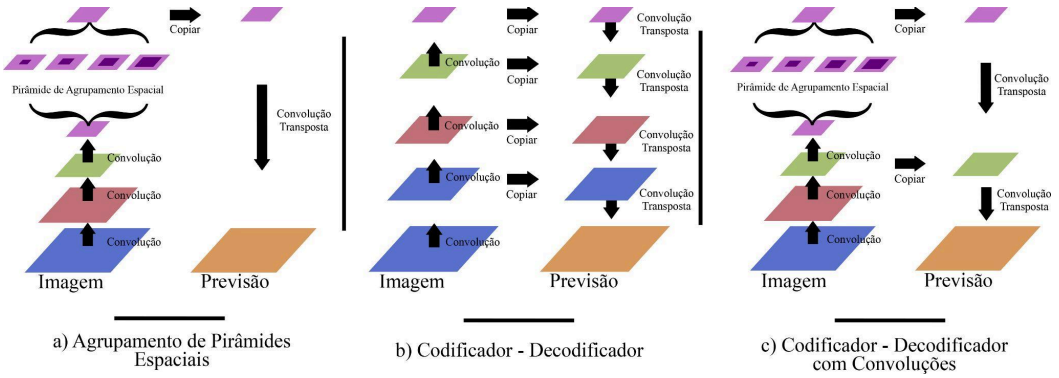
objetos em múltiplas escalas; e a diminuição da precisão de localização causada pela invariância [13,14].

O DeepLabV3+ representa uma evolução do DeepLabV3, incorporando um módulo de Agrupamento de Pirâmide Espacial (ASPP) e uma estrutura Codificador-Decodificador. O módulo codificador extrai informações semânticas ricas, enquanto o módulo decodificador é responsável por recuperar os contornos dos objetos nas imagens. Essa combinação garante que a arquitetura não apenas compreenda o contexto geral, mas também preserve detalhes cruciais da segmentação [13].

O módulo codificador é particularmente eficaz, pois permite a extração de características em resoluções arbitrárias através das convoluções dilatadas, otimizando assim a segmentação sem perda de detalhes essenciais. As convoluções dilatadas permitem que a rede mantenha a resolução espacial enquanto amplia seu campo de visão, o que é fundamental para a detecção de objetos em diferentes escalas. Essa técnica é crucial, especialmente em aplicações que exigem alta precisão, como na segmentação de imagens médicas ou em sistemas de reconhecimento facial [13,14].

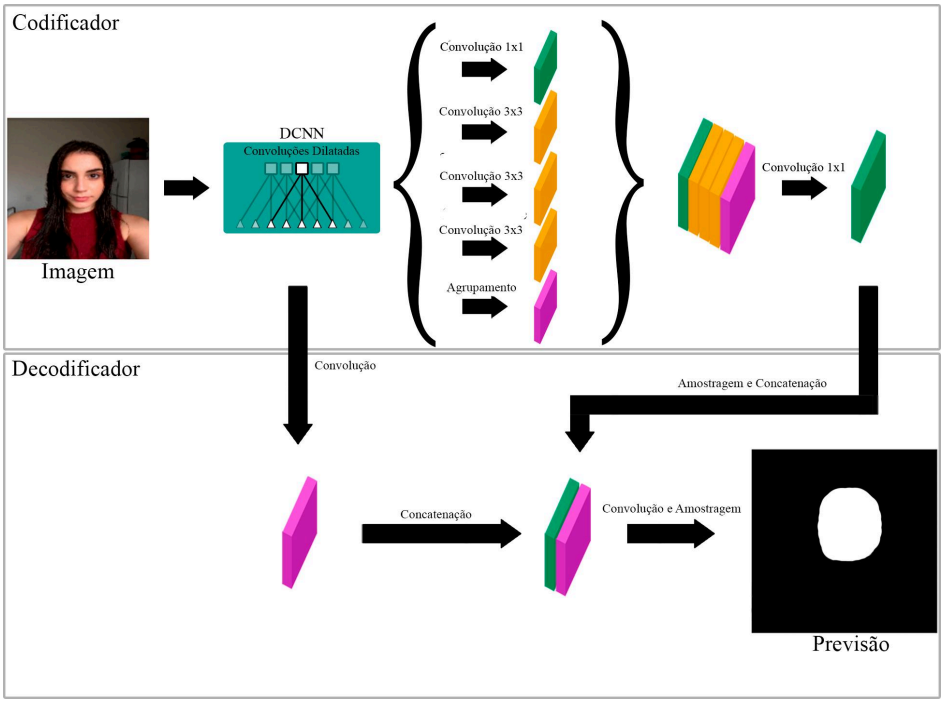
As técnicas aplicadas no funcionamento do DeepLabV3+ são ilustradas na Figura 4, enquanto a estrutura completa da arquitetura pode ser visualizada na Figura 5. Estudos demonstram que o DeepLabV3+ alcançou resultados excepcionais em *benchmarks* como PASCAL VOC e *Cityscapes*, superando outras arquiteturas em termos de precisão de segmentação e robustez em ambientes complexos [13,14].

Figura 4 - Ferramentas DeepLabV3+



Fonte: Elaborada pelo autor.

Figura 5 - Funcionamento completo DeepLabV3+



Fonte: Elaborada pelo autor.

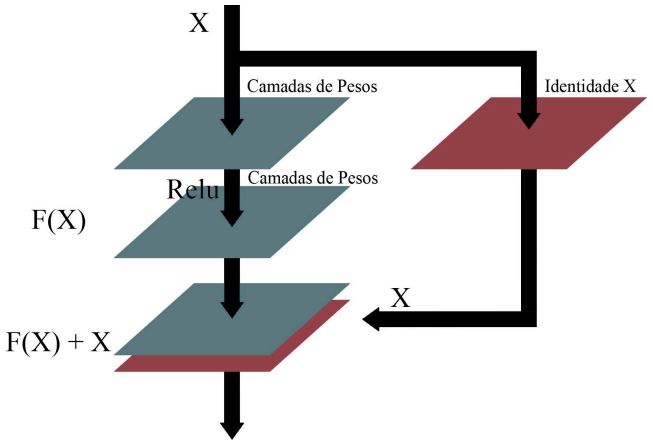
2.4 Arquiteturas de Classificação Estudadas

O presente capítulo oferece uma revisão bibliográfica das principais arquiteturas de classificação de imagens empregadas neste trabalho. A classificação de imagens é uma tarefa central na área de visão computacional, cujo objetivo é atribuir rótulos ou categorias a uma imagem completa. Diferente da segmentação, que trabalha em nível de pixel, a classificação foca em identificar a classe principal de uma imagem com base nas características extraídas. As CNNs desempenham um papel crucial nesse contexto, revolucionando o campo com sua capacidade de aprender automaticamente as melhores representações das imagens.

2.4.1 ResNet50

A arquitetura *Residual Network* (ResNet) foi desenvolvida para lidar com a dificuldade de treinamento de redes neurais profundas, introduzindo o conceito de "conexões residuais". Em redes tradicionais, à medida que a profundidade aumenta, surgem problemas de degradação de desempenho, em que o erro de treinamento aumenta à medida que mais camadas são adicionadas. A ResNet50 supera esse desafio ao reformular a rede para aprender funções residuais, em vez de funções diretas. Isso é feito através de atalhos que pulam uma ou mais camadas, permitindo que o aprendizado se concentre em ajustes residuais ao invés de tentar modelar diretamente a saída esperada [15]. Essa implementação está resumida na Figura 6.

Figura 6 - Aprendizado de funções residuais.



Fonte: Elaborada pelo autor.

O funcionamento da ResNet50 baseia-se em blocos de resíduos, onde a entrada original é preservada e somada à saída das camadas convolucionais subsequentes. Dessa forma, se a saída das camadas for pequena ou próxima de zero, a rede ainda será capaz de passar a entrada original, evitando que informações importantes sejam perdidas. Essa configuração não adiciona complexidade computacional significativa, mas melhora consideravelmente o desempenho em tarefas de classificação e reconhecimento de imagens. A estrutura básica da ResNet50 utiliza múltiplos blocos de convolução e agrupamento, seguidos de um agrupamento global e camadas totalmente conectadas para a classificação final [15,16].

No caso específico da ResNet50, que é uma das variantes mais populares, a rede utiliza 50 camadas organizadas em blocos de convolução de garrafa, onde convoluções de 1x1 são usadas para reduzir e restaurar a dimensão, intercaladas com convoluções de 3x3. Essa estrutura reduz a complexidade computacional enquanto mantém a precisão e eficiência no treinamento da rede. O uso de atalhos de identidade permite que a rede seja treinada de forma mais eficiente, resultando em menor taxa de erro em *benchmarks* como *ImageNet* [15,16].

Essa arquitetura mostrou resultados de ponta em tarefas como a classificação no ImageNet, vencendo competições como o *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) 2015, onde uma versão com 152 camadas alcançou um erro de top-5 de apenas 3,57%. A ResNet50 é amplamente utilizada em diversas aplicações de visão

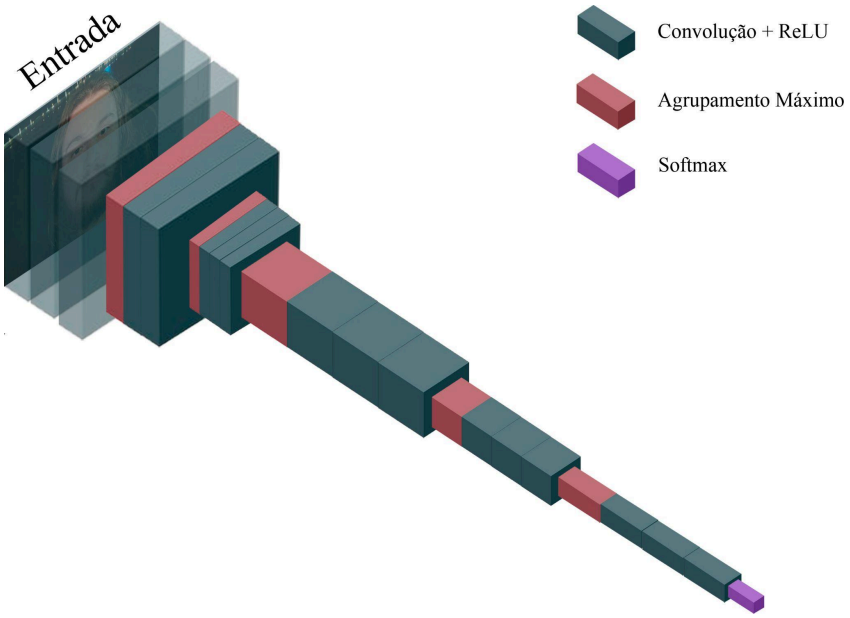
computacional, como detecção de objetos, segmentação de imagens e reconhecimento facial, devido à sua capacidade de aprender representações profundas e generalizar bem em diferentes conjuntos de dados [15].

2.4.2 *Visual Geometry Group Network* (VGG)

A arquitetura *Visual Geometry Group* (VGG) é uma rede neural convolucional desenvolvida para tarefas de classificação de imagens. Uma das principais características dessa arquitetura é o uso de convoluções de pequeno tamanho (3x3), em profundidade, para extrair características das imagens. A VGG introduziu uma estrutura simplificada que, ao empilhar essas camadas de convolução, permite que a rede capture padrões cada vez mais complexos à medida que a profundidade da rede aumenta. O modelo é organizado em blocos, onde cada bloco é composto por duas ou três camadas convolucionais seguidas por uma operação de agrupamento para redução da dimensionalidade [17].

O funcionamento da VGG é baseado em uma estrutura profunda que pode variar entre 16 a 19 camadas, sendo amplamente utilizada para tarefas de reconhecimento e classificação de imagens. No final das camadas convolucionais, há uma série de camadas densamente conectadas, seguidas de uma camada softmax para a classificação final. A profundidade da rede permite que a VGG obtenha um desempenho superior em *benchmarks* como o *ImageNet*, uma vez que as camadas convolucionais capturam características mais abstratas à medida que se aprofundam [17]. O funcionamento descrito acima está exemplificado na Figura 7.

Figura 7 - Arquitetura VGG-*face*.



Fonte: Elaborada pelo autor.

Apesar do seu excelente desempenho em termos de precisão, a VGG possui algumas desvantagens, como seu elevado custo computacional e o grande uso de memória devido à quantidade significativa de parâmetros, especialmente nas camadas totalmente conectadas. Isso pode tornar o treinamento mais lento e exigir mais recursos computacionais em comparação a arquiteturas mais modernas. Mesmo assim, a VGG permanece amplamente utilizada, especialmente quando se emprega o método de aprendizado por transferência, aproveitando seus pesos pré-treinados para acelerar o desenvolvimento de novos modelos [17].

2.4.3 MobileNet

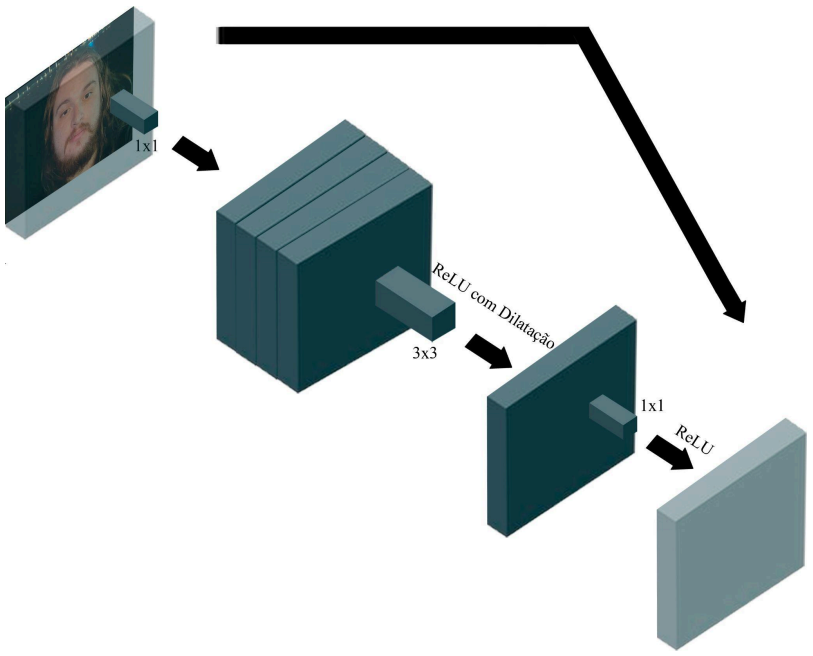
A arquitetura MobileNet foi desenvolvida para ser eficiente e leve, permitindo o uso em dispositivos móveis e embarcados, com recursos computacionais limitados. Um dos principais componentes que tornam o MobileNet tão eficiente é o uso de convoluções separáveis em profundidade. Esse tipo de convolução reduz drasticamente o custo computacional, dividindo a operação de convolução tradicional em duas etapas: uma

convolução espacial que opera separadamente em cada canal de entrada e, em seguida, uma convolução ponto a ponto (1x1) para combinar os resultados dos canais [18,19].

A arquitetura também introduz dois hiperparâmetros, o multiplicador de largura e o multiplicador de resolução, que permitem controlar o número de filtros em cada camada e a resolução das imagens de entrada. Isso oferece flexibilidade para ajustar o equilíbrio entre desempenho e precisão, dependendo das restrições de recursos e dos requisitos da aplicação [19].

O MobileNetV2 trouxe avanços com a introdução de estruturas residuais invertidas e o uso de convolução de garrafa lineares. Essa estrutura melhorou ainda mais a eficiência ao expandir as características internamente antes de realizar a projeção de volta para um espaço de características compactas, mantendo a capacidade expressiva das transformações não lineares por canal [18]. A versão V2 da *MobileNet* está representada na Figura 8.

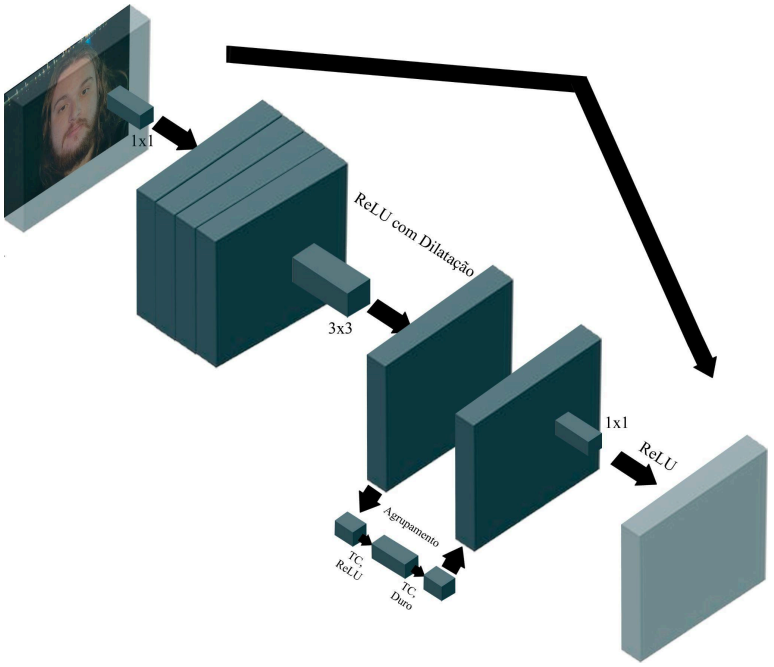
Figura 8 - Arquitetura MobileNetV2.



Fonte: Elaborada pelo autor.

Já o MobileNetV3, que é uma evolução das versões anteriores, combina técnicas de busca de arquitetura automatizada com avanços na configuração arquitetônico, como o uso de h-swish, uma variante otimizada da função de ativação de balanço, que melhora a eficiência em dispositivos móveis. Além disso, ele introduz módulos de aperto e excitação para ajustar dinamicamente a importância dos diferentes canais de característica [18,19]. A versão V3 da MobileNet está representada na Figura 8.

Figura 9 - Arquitetura MobileNetV3.



Fonte: Elaborada pelo autor.

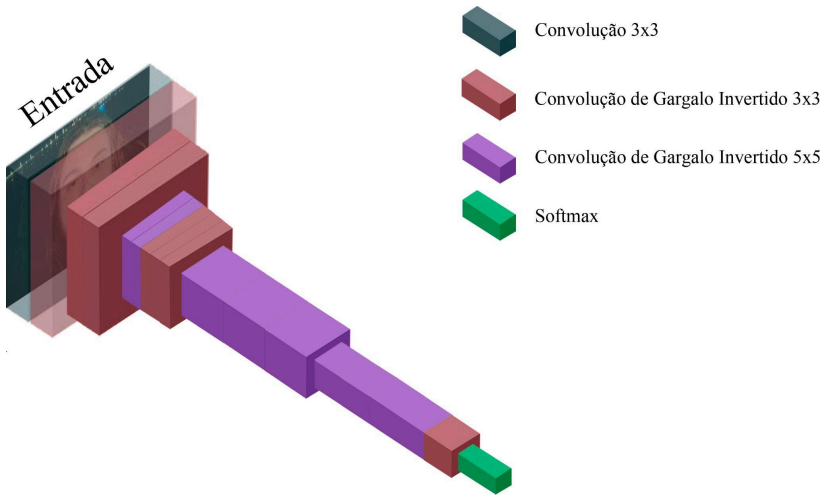
Essas características tornam o MobileNet adequado para tarefas de visão computacional em dispositivos com restrições de hardware, oferecendo um bom equilíbrio entre precisão e eficiência.

2.4.4 EfficientNet

A arquitetura EfficientNet foi desenvolvida para otimizar o equilíbrio entre precisão e eficiência no treinamento de redes neurais profundas. O principal diferencial da EfficientNet é o uso de escala composta, uma técnica que ajusta simultaneamente a profundidade, largura e resolução da rede, com o objetivo de melhorar o desempenho sem aumentar de forma desnecessária o custo computacional. Diferente de outras arquiteturas que aumentam arbitrariamente um único parâmetro (como profundidade), a EfficientNet otimiza todos os aspectos da rede de maneira conjunta, permitindo um uso mais eficiente dos recursos computacionais [20] .

A versão base, conhecida como EfficientNetB0, serve como ponto de partida, e a partir dela são geradas variantes como EfficientNetB1 até EfficientNetB7, cada uma com maior capacidade de processamento e precisão. A EfficientNet utiliza blocos MBConv, uma variante dos blocos de resíduo invertido, que foram introduzidos no MobileNetV2, otimizando a eficiência na extração de características. Além disso, a rede incorpora a técnica de normalização em lote e funções de ativação como balanço, que ajudam a manter a estabilidade do treinamento, mesmo em redes mais profundas [20]. A arquitetura da EfficientNet está representada na Figura 10.

Figura 10 - Arquitetura EfficientNetB0.



Fonte: Elaborada pelo autor.

No campo do reconhecimento de expressões faciais, a EfficientNet demonstrou ser capaz de superar outras arquiteturas profundas, em termos de precisão. Estudos mostram que variantes como a EfficientNetB3 oferecem alta acurácia (acima de 94%) na classificação de emoções a partir de expressões faciais. Além disso, a *Receiver Operating Characteristic curve* (ROC-*curve*) revela que a EfficientNetB3 tem excelente desempenho na detecção de emoções como surpresa e felicidade, com uma pontuação ROC média superior a 0.9, destacando sua capacidade de reconhecimento em tarefas de visão computacional complexas[20] .

Em resumo, a EfficientNet é uma escolha eficaz para aplicações que requerem um equilíbrio entre alto desempenho e uso otimizado de recursos, sendo amplamente utilizada em tarefas como reconhecimento de imagens e expressões faciais, onde eficiência e precisão são essenciais.

3 Metodologia

O capítulo de Metodologia descreve as etapas e os procedimentos adotados para a realização deste trabalho, fundamentado na revisão bibliográfica previamente realizada sobre as ferramentas e tecnologias selecionadas. A metodologia foi estruturada de forma a aplicar de maneira eficiente os conceitos e técnicas explorados na pesquisa teórica, garantindo a execução e validação dos experimentos propostos. Serão apresentadas as etapas de desenvolvimento, ferramentas de software utilizadas, além dos métodos de avaliação que permitiram a análise dos resultados.

3.1 Ajuste de Necessidades e Objetivos

O projeto começou com a necessidade de desenvolver uma ferramenta automatizada, eficiente e segura para o gerenciamento do serviço de empréstimo de bicicletas elétricas do BATLAB - UFMS. Diante dessa demanda, a primeira etapa metodológica envolveu a definição clara dos objetivos e o ajuste das necessidades do sistema, que deveria integrar tecnologias de IA e reconhecimento facial para facilitar a automação do processo de empréstimo, garantindo a segurança.

O processo metodológico teve início com um estudo sobre o funcionamento prático das redes neurais, especificamente no ambiente de programação MATLAB. O *software* desempenha um papel fundamental no ambiente de pesquisa, especialmente em áreas que envolvem cálculos matemáticos, modelagem, simulação e visualização de dados. Sua importância está ligada à capacidade de realizar cálculos numéricos de forma eficiente, sendo ideal para resolver problemas de álgebra linear, sistemas diferenciais e estatísticas, o que permite a implementação e teste rápido de algoritmos. Além disso, o MATLAB oferece várias ferramentas para simulação e modelagem de sistemas dinâmicos, tornando-se essencial em áreas como engenharia, controle e física, onde simular o comportamento de sistemas em tempo real é uma necessidade frequente.

No processamento de sinais e imagens, o MATLAB é amplamente utilizado devido às suas bibliotecas robustas, que facilitam a análise, manipulação e visualização de dados. Isso o

torna uma escolha comum em projetos de reconhecimento de padrões, análise de áudio e vídeo, e segmentação de imagens. Outro ponto importante é a capacidade do MATLAB de gerar gráficos interativos em 2D e 3D, o que facilita a interpretação e a comunicação dos resultados das pesquisas.

O MATLAB também se destaca por suas "*toolboxes*", pacotes de ferramentas especializadas para áreas específicas de pesquisa. Estas incluem AM, AP, controle, robótica e, por meio do Simulink, simulações de sistemas complexos em tempo real. Isso oferece aos pesquisadores flexibilidade e adaptabilidade para desenvolver soluções sob medida para seus projetos.

Inicialmente, o foco foi direcionado ao desenvolvimento e treinamento de modelos de segmentação, responsáveis por localizar faces nas imagens, utilizando pequenos lotes de dados provenientes de bancos de imagens públicos, como o FEI *Dataset* e o *Caltech Face Dataset* 1999. Esses conjuntos foram essenciais para rotular e identificar as áreas de interesse nas imagens, viabilizando o ajuste inicial do modelo de segmentação.

O processo de rotulação, também conhecido como *labeling*, consiste na atribuição de rótulos ou categorias a dados, um passo fundamental em tarefas de aprendizado supervisionado. No contexto de processamento de imagens, rotular significa identificar e marcar elementos específicos da imagem, como objetos ou regiões, com uma classe correspondente, como "carro", "pessoa" ou "face". No caso deste projeto, o rótulo “face” foi atribuído às regiões de interesse em cada imagem, conforme ilustrado na Figura 11. Esse processo é crucial para o desempenho do modelo, pois a precisão e qualidade das rotulações influenciam diretamente a eficácia do modelo treinado, especialmente em algoritmos supervisionados.

Figura 11 - Atribuição do rótulo “face” a imagem realizada no MATLAB.

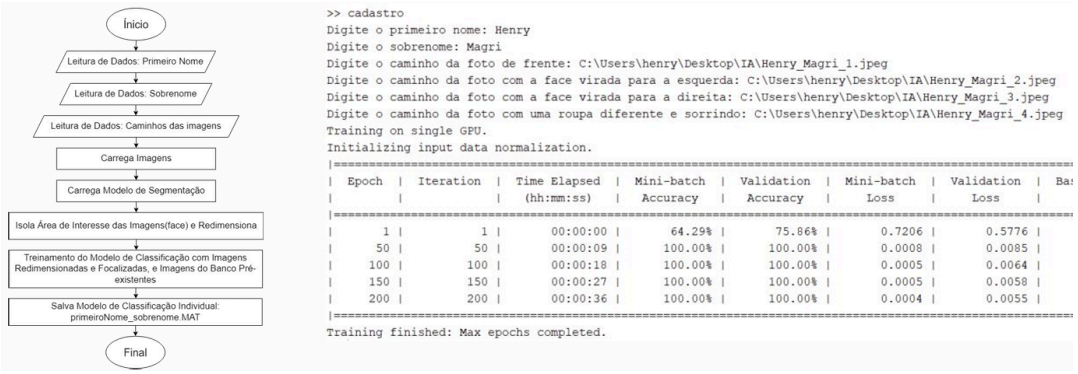
Fonte: Elaborada pelo autor.

Como o modelo de segmentação é estático e único para todo o sistema, foi possível alocar mais tempo e poder de processamento em seu treinamento inicial para obter um resultado altamente preciso. Esse alto nível de precisão na segmentação é essencial, pois garante uma base confiável para a etapa de classificação, facilitando o processo de identificação e economizando tempo e recursos do sistema. Ao fornecer ao modelo de classificação imagens já segmentadas, o treinamento desse segundo modelo pode ser realizado de forma mais rápida e com menor exigência de processamento, pois a etapa de segmentação já teria isolado as regiões de interesse, tornando o processo de cadastro mais ágil e eficiente para o usuário. Além disso, ao reduzir o tempo de treinamento dos modelos individuais de classificação, o servidor economiza capacidade de processamento, mantendo um desempenho consistente e eficaz.

Após a rotulação das imagens, o primeiro modelo foi treinado por meio de aprendizado supervisionado, sendo responsável pela segmentação das imagens e pela localização e isolamento das faces. Em seguida, foi desenvolvido um segundo modelo, um classificador, para identificar se a face capturada correspondia a um usuário específico ou não. Cada usuário possui um modelo de classificação individual, oferecendo maior segurança e personalização.

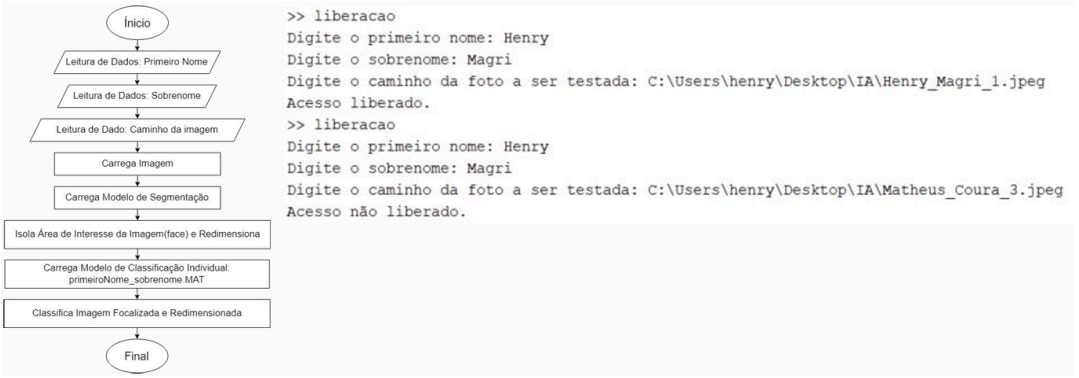
Com o MATLAB, foi possível ajustar parâmetros, validar e otimizar os modelos de IA, garantindo precisão tanto na segmentação quanto na classificação. Como resultado dessa etapa de estudo e desenvolvimento, dois algoritmos principais foram criados: um simulando um processo de cadastro simplificado e outro destinado à validação e liberação das bicicletas. Esses *scripts* foram cruciais para idealizar e validar o funcionamento do sistema em um ambiente de teste, assegurando que ele atendesse aos requisitos de segurança, praticidade e eficiência estabelecidos pelo BATLAB para o serviço de empréstimo de bicicletas elétricas. Ambos os fluxogramas e terminais de funcionamento estão presentes nas Figuras 12 e 13.

Figura 12 -Processo de Cadastro Simplificado.



Fonte: Elaborada pelo autor.

Figura 13 -Processo de Validação Simplificado.



Fonte: Elaborada pelo autor.

3.2 Escolha da Linguagem de Programação

A linguagem de programação Python foi escolhida para o desenvolvimento do *backend* do sistema devido à sua simplicidade e legibilidade. A sintaxe clara e intuitiva de Python facilita a escrita e compreensão do código, permitindo que o desenvolvimento se concentre na lógica do problema, sem se preocupar com a complexidade da linguagem. Essa característica torna o desenvolvimento mais ágil, especialmente em projetos que envolvem a implementação de soluções complexas, como inteligência artificial e aprendizado de máquina.

Além disso, Python conta com uma ampla variedade de bibliotecas e *frameworks* especializados que o tornam ideal para o desenvolvimento de sistemas baseados em IA. Bibliotecas como o TensorFlow, desenvolvido pelo Google, oferecem uma plataforma robusta para a construção e treinamento de redes neurais profundas. A Keras, uma API de alto nível que roda sobre o TensorFlow, simplifica a criação de modelos de aprendizado profundo, fornecendo uma interface intuitiva para a construção de arquiteturas de rede e o ajuste de hiperparâmetros [21, 22].

Outras bibliotecas, como o scikit-learn, são amplamente utilizadas para aprendizado de máquina supervisionado e não supervisionado, com ferramentas que facilitam desde a seleção de algoritmos até a validação dos modelos. Já o NumPy é essencial para a manipulação e análise eficiente de grandes volumes de dados, permitindo operações matemáticas de alto desempenho que são fundamentais para o pré-processamento de dados em projetos de IA [23,24].

Portanto, a escolha de Python para o desenvolvimento deste projeto não apenas simplifica o processo de codificação, como também oferece uma vasta gama de ferramentas especializadas que são indispensáveis para a implementação de sistemas baseados em redes neurais e aprendizado de máquina.

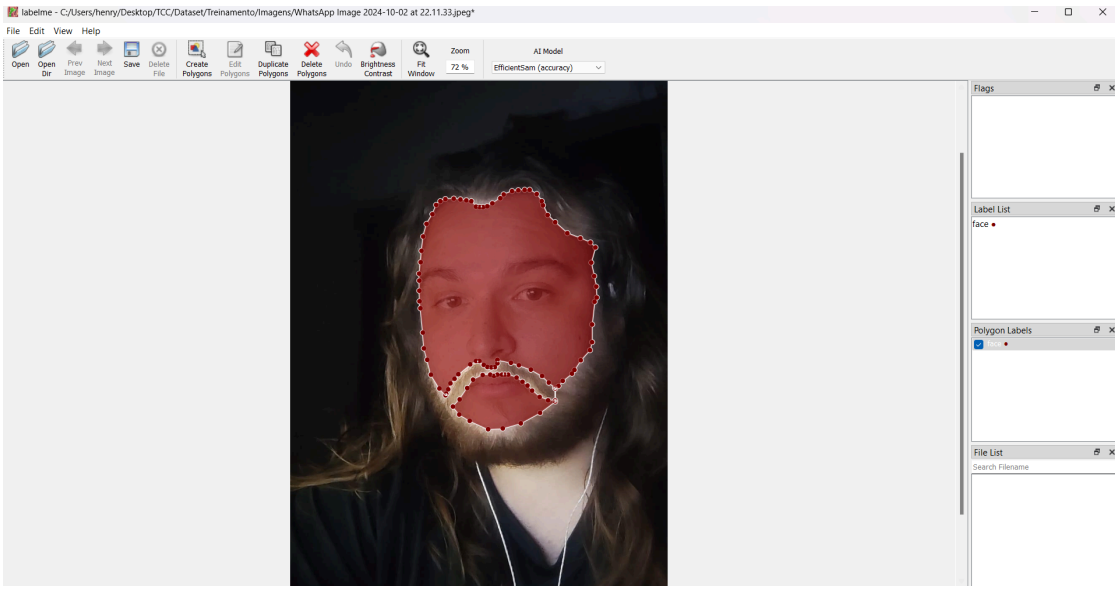
3.3 Rotulação de Dados para Segmentação

A etapa de rotulação dos dados foi fundamental para o treinamento das arquiteturas de segmentação avaliadas e comparadas para este trabalho. Para isso, foi escolhida a ferramenta

Labelme, que permite a rotulação manual das áreas de interesse nas imagens [25]. A rotulação foi realizada considerando uma única máscara por imagem, focada exclusivamente na segmentação da face dos usuários.

Nas imagens, a face foi rotulada a partir do alto da testa até o queixo, desconsiderando grandes áreas de pelos faciais, como barba, quando presentes. Lateralmente, a rotulação foi feita de orelha a orelha, exceto em casos onde o cabelo cobria essas regiões. Esse processo garantiu que apenas a face estivesse contornada de forma precisa, eliminando interferências de elementos como cabelo ou pelos faciais. O exemplo de como as imagens foram rotuladas no Labelme pode ser observado na Figura 14.

Figura 14 - Atribuição do rótulo “face” a imagem realizada no Labelme.

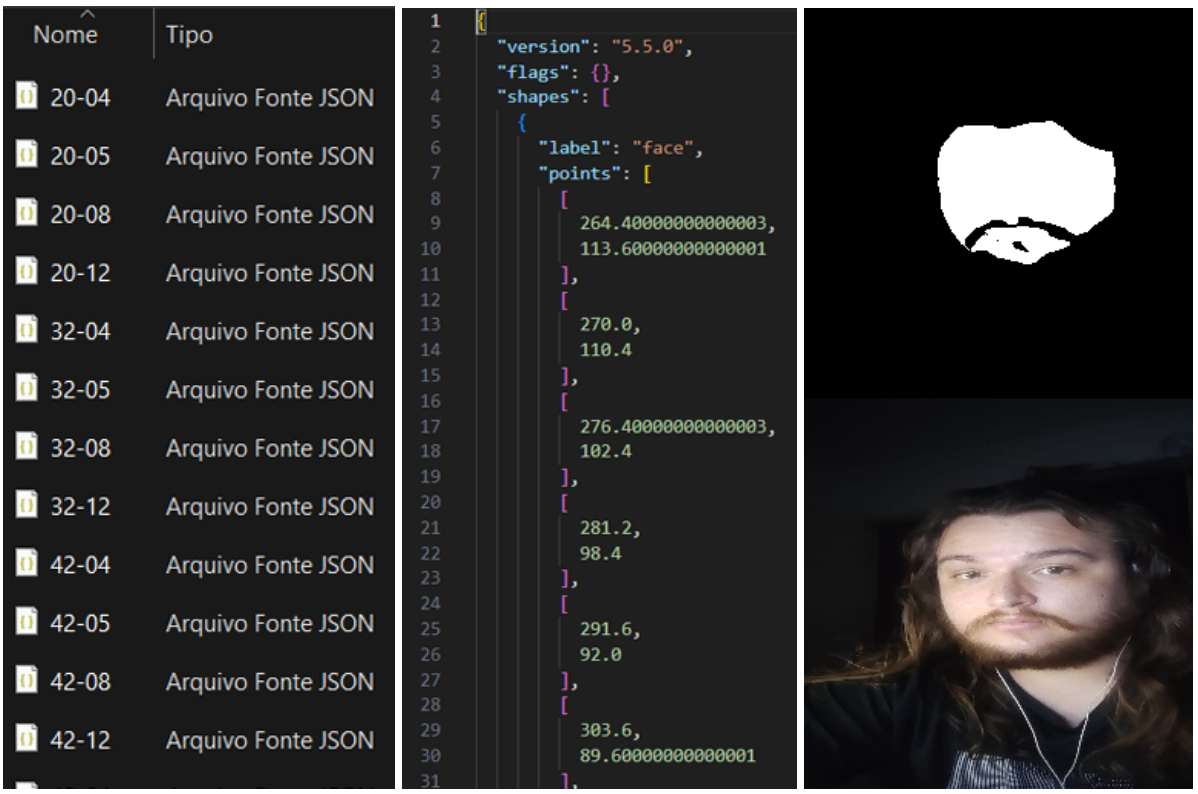


Fonte: Elaborada pelo autor.

A ferramenta Labelme gera arquivos *JavaScript Object Notation* (JSON) como resultado da rotulação. Esses arquivos contêm conjuntos com as coordenadas dos pontos que delimitam as faces nas imagens, formando o contorno exato da região segmentada. Após a criação desses arquivos JSON, eles foram convertidos para o formato *Portable Network*

Graphics (PNG), gerando máscaras binárias para cada imagem rotulada. As máscaras PNG mantiveram o mesmo nome das imagens originais, facilitando a correspondência entre as máscaras e as respectivas imagens de entrada. Para finalizar o processo, tanto as máscaras quanto as imagens originais foram redimensionadas para 256x256 pixels, o tamanho de entrada definido para os modelos de segmentação. Algumas etapas desse processo estão exemplificadas na Figura 15.

Figura 15 - a) Arquivos gerados pelo processo de rotulação b) Conteúdo dos arquivos c) Comparação entre máscara PNG e imagem original



a) b) c)

Fonte: Elaborada pelo autor.

A rotulação cuidadosa foi uma etapa crucial para garantir que as arquiteturas de segmentação pudessem ser treinadas com um conjunto de dados bem definido, permitindo

que os modelos fossem otimizados para a tarefa de reconhecimento facial de forma precisa e consistente.

3.4 Treinamento dos Modelos de Segmentação

Nesta seção, descrevemos o processo de treinamento e comparação dos modelos de segmentação escolhidos para o projeto: FCN, U-Net e DeepLabV3+. O objetivo principal desses modelos é realizar a detecção e segmentação das faces nas imagens, de modo que elas possam ser isoladas e posteriormente classificadas. Inicialmente, a arquitetura *Mask* R-CNN foi considerada, mas foi descartada devido ao seu maior tempo de treinamento e sua natureza voltada à detecção de múltiplas instâncias em uma mesma imagem, o que não era necessário neste contexto.

Para garantir uma comparação justa entre os resultados, todos os modelos foram treinados a partir dos mesmos parâmetros. O treinamento foi realizado utilizando as bibliotecas OpenCV, TensorFlow, Keras e NumPy, ferramentas amplamente utilizadas em projetos de visão computacional e aprendizado profundo. O OpenCV foi utilizado para manipulação e processamento das imagens, enquanto o TensorFlow e o Keras foram responsáveis pela construção, treinamento e avaliação dos modelos de redes neurais. O NumPy foi utilizado para a manipulação eficiente de conjuntos numéricos, fundamentais para a manipulação de dados de imagem [21, 22, 24, 26].

O treinamento foi feito com 140 imagens de faces rotuladas, que passaram por aumento de dados, uma técnica que aplica transformações nas imagens para criar novas variações. As transformações incluíram rotações de 20 graus, deslocamento vertical, lateral e ampliação de 20%, espelhamento horizontal, ajustes de cisalhamento de 20% e preenchimento de áreas vazias utilizando o método "mais próximo", garantindo que os modelos fossem expostos a uma maior diversidade de exemplos [22, 26].

A validação do modelo foi realizada a partir de 10% das imagens que foram retiradas do processo de treinamento após o aumento de dados.

O treinamento utilizou uma taxa de aprendizado de 1e-5, um hiperparâmetro que controla o tamanho dos ajustes feitos nos pesos da rede a cada atualização. Uma taxa de aprendizado muito alta pode fazer com que o modelo aprenda rapidamente, mas corra o risco de não encontrar a melhor solução, enquanto uma taxa muito baixa pode resultar em aprendizado mais lento, porém com maior precisão [22, 26].

Além disso, foram utilizados lotes de 4 imagens, o que significa que o modelo processa 4 imagens de cada vez antes de atualizar seus pesos. O uso de um tamanho de lote menor pode tornar o treinamento mais rápido em termos de memória, mas pode levar a atualizações menos estáveis, enquanto tamanhos maiores oferecem atualizações mais consistentes, mas exigem mais recursos computacionais [22, 26].

O treinamento foi realizado por 100 épocas, com o auxílio de um *callback* (função de retorno) para monitorar as perdas durante a validação, salvando automaticamente a melhor versão do modelo. As perdas indicam a discrepância entre as previsões do modelo e os valores reais, refletindo o quanto o modelo precisa ajustar seus parâmetros para melhorar a precisão. Quanto menor o valor da perda, mais próximo o modelo está do resultado ideal. Além disso, foi aplicada uma parada antecipada com paciência de 50 épocas, interrompendo o treinamento caso não houvesse melhorias significativas nesse período, o que previne o sobreajuste e otimiza o desempenho do modelo [22, 26].

Esse processo padronizado, apoiado por ferramentas robustas e técnicas de aprendizado máquina e aprendizado profundo, garantiu que a comparação entre os modelos fosse sólida e justa, permitindo uma análise detalhada sobre qual arquitetura apresentou o melhor desempenho na tarefa de segmentação facial.

3.5 Treinamento dos Modelos de Classificação

O treinamento dos modelos de classificação foi realizado com o objetivo de identificar de forma binária se uma imagem corresponde ao indivíduo que o modelo foi treinado para reconhecer, ou se a imagem pertence a outra pessoa. Esse processo foi projetado para permitir a automatização completa, eliminando a necessidade de rotulação manual e possibilitando a criação de modelos com apenas 4 imagens do usuário-alvo, tornando o sistema mais eficiente

e fácil de escalar. As arquiteturas comparadas, ResNet50, EfficientNetB3, MobileNet e VGG, foram treinadas utilizando um conjunto de 145 imagens segmentadas e recortadas pelo modelo de segmentação que se destacou na fase anterior. Dentre essas imagens, 4 pertenciam à classe do indivíduo-alvo, enquanto as outras 141 correspondiam a outras pessoas.

Para tratar o desbalanceamento entre as classes e evitar que o modelo ficasse tendencioso, foi utilizado aumento de dados, aplicando transformações apenas na classe com menos exemplos, ou seja, a do usuário-alvo. As transformações incluíram espelhamento horizontal, rotações de 5 graus e ajustes de brilho, variando entre 85% e 115%. Decidiu-se não utilizar transformações como zoom e deslocamento, pois as imagens já haviam passado por um processo de segmentação focado no rosto, e essas transformações poderiam cortar ou alterar características importantes da face do usuário, prejudicando o desempenho do modelo [22, 26].

Todos os modelos foram treinados utilizando uma taxa de aprendizado de 1e-5, um hiperparâmetro que controla o quanto os pesos da rede são ajustados a cada atualização. Além disso, as arquiteturas foram inicializadas com pesos pré-treinados no ImageNet, uma técnica que permite aproveitar o conhecimento prévio das redes sobre características gerais de imagens, como bordas e texturas, e adaptar esse conhecimento à tarefa específica de reconhecimento facial. Para isso, a camada final dos modelos foi substituída por camadas customizadas, específicas para a tarefa de classificação binária [22, 26].

O treinamento foi realizado em duas etapas. Na primeira etapa, durante 20 épocas, as camadas base do modelo foram congeladas, mantendo os pesos obtidos do treinamento no *ImageNet*. Nesse estágio, apenas as camadas criadas para a tarefa de classificação foram treinadas, permitindo que o modelo aprendesse a distinguir entre o indivíduo-alvo e outras pessoas sem alterar as camadas já treinadas. Na segunda etapa, conhecida como ajuste fino (*fine-tuning*), as camadas base foram descongeladas, permitindo que todo o modelo fosse treinado por mais 10 épocas. O ajuste fino adapta as camadas pré-treinadas para capturar melhor as características específicas das imagens do conjunto de dados, maximizando a precisão do modelo [22, 26].

Para automatizar o processo de rotulação, sem a necessidade de intervenção manual, o treinamento foi estruturado com base na organização de pastas. Duas pastas foram criadas:

uma contendo as 4 imagens do usuário e outra com as 141 imagens de diferentes pessoas. O código automaticamente rotulou as classes a partir dessa estrutura, garantindo que o processo fosse totalmente automatizado.

Durante ambas as etapas de treinamento, foi utilizado um *callback* para monitorar as perdas durante o processo de validação, salvando automaticamente a melhor versão do modelo. As perdas representam a discrepância entre as previsões do modelo e os valores reais, indicando o quanto o modelo ainda precisa ajustar seus parâmetros para melhorar sua precisão. Quanto menor o valor da perda, mais próximo o modelo está dos resultados desejados. Além disso, foi aplicada uma parada antecipada com paciência de 8 épocas, o que significa que o treinamento seria interrompido caso não houvesse melhorias significativas nesse intervalo de tempo, evitando o sobreajuste (*overfitting*) e otimizando o desempenho do modelo [22, 26].

Esse processo garantiu que o treinamento dos modelos de classificação fosse eficiente e automatizado, possibilitando o uso direto do modelo que apresentasse o melhor desempenho no projeto final, sem necessidade de ajustes manuais, e com uma boa capacidade de generalização, mesmo a partir de um pequeno conjunto de imagens do usuário.

3.6 Desenvolvimento do Sistema Principal

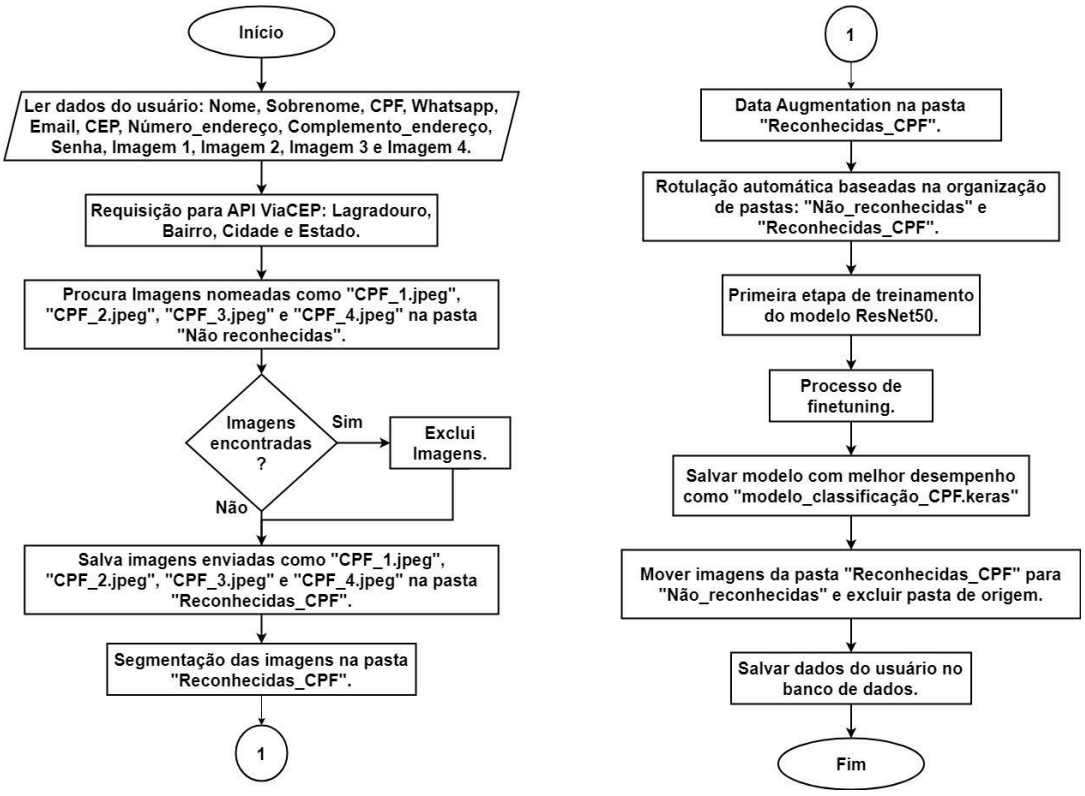
O desenvolvimento do sistema principal foi realizado utilizando o *framework* FastAPI e um banco de dados PostgreSQL, com o objetivo de garantir a eficiência do sistema em lidar com múltiplos usuários e realizar atividades em paralelo. O FastAPI foi escolhido por sua capacidade de fornecer um sistema altamente eficiente e performático, graças ao seu suporte nativo para comportamento assíncrono. O comportamento assíncrono permite que a API processe múltiplas solicitações ao mesmo tempo sem bloquear outras operações, o que é essencial em sistemas onde múltiplos usuários podem estar interagindo simultaneamente. Isso, combinado com a escalabilidade e robustez do PostgreSQL, garantiu um gerenciamento eficiente dos dados e operações do sistema [6, 27].

Inicialmente, o sistema foi projetado para funcionar por meio de um operador, que realiza a comunicação com as docas para testar o sistema. Esse operador recebe solicitações

de liberação de bicicletas em um endereço de e-mail, e a conexão automática do sistema com as docas será implementada após a validação completa do sistema. O sistema possui duas rotas principais, uma para o cadastro de usuários e outra para a liberação das bicicletas elétricas, além de rotas secundárias para o retreinamento do modelo do usuário, redefinição de senha ou troca de número de celular, e reporte de defeitos.

A rota de cadastro possui uma pasta própria para realizar seus processos na raiz do projeto e recebe informações do usuário, como nome, sobrenome, Cadastro de Pessoa Física (CPF), Código de Endereçamento Postal (CEP), número de endereço, número de telefone, complemento de endereço, e-mail, senha e quatro imagens do rosto do usuário. As demais informações sobre o endereço do usuário são coletadas através de uma requisição para a API ViaCEP com o CEP do mesmo. O processo é iniciado verificando se o CPF enviado já existe na pasta de rostos diversos localizada dentro da pasta reservada para o cadastro. Caso existam imagens com o mesmo CPF (por participação em testes anteriores), essas imagens são apagadas. Uma nova pasta é criada com o CPF do usuário, e as imagens enviadas são salvas com um nome baseado no CPF e um índice numérico. A seguir, o modelo de segmentação treinado é utilizado para identificar e recortar as faces nas imagens, e o modelo de classificação individual do usuário é treinado automaticamente baseado nos parâmetros explicados na seção 3.5 de treinamento dos modelos de classificação e utilizando a rotulação baseada na estrutura de pastas. Após o treinamento, as imagens do usuário são transferidas para a pasta de rostos diversos para aumentar o banco de imagens, e as informações do usuário são salvas na tabela “usuários” no banco de dados. O modelo de IA treinado é salvo em uma pasta dedicada aos modelos de classificação e ao modelo segmentação, nomeado com o CPF do usuário. O fluxograma com as etapas realizadas para cadastrar novos usuários está na Figura 16.

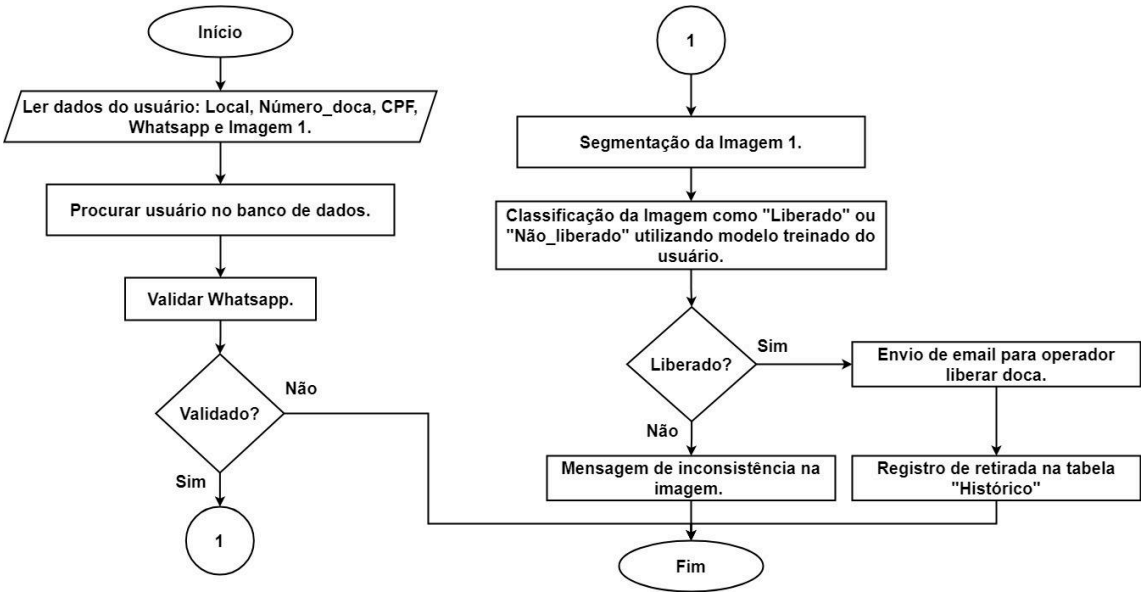
Figura 16 - Fluxograma da rota de cadastro de novos usuários.



Fonte: Elaborada pelo autor.

A rota de validação e liberação de bicicleta requer que o usuário informe o local onde está (BATLAB ou Eletroposto UFMS), o número da doca, seu CPF, número de telefone e envie uma imagem para validação. O processo começa com a busca das informações do usuário no banco de dados a partir do CPF informado. Caso o usuário seja encontrado, o próximo passo é a validação do número de telefone. Se ambas as informações forem confirmadas, a imagem enviada é segmentada pelo modelo de segmentação, e a face recortada é classificada pelo modelo de classificação do usuário, baseado no CPF. Se o usuário for identificado corretamente, é criado um registro no banco de dados na tabela “histórico”, contendo data, hora, ID do usuário, local e doca. Em seguida, um e-mail automático é enviado ao operador do sistema solicitando a liberação da doca. Caso o usuário não seja identificado, uma mensagem de erro é enviada. O fluxograma com as etapas realizadas para validar o usuário e liberar a doca está na Figura 17.

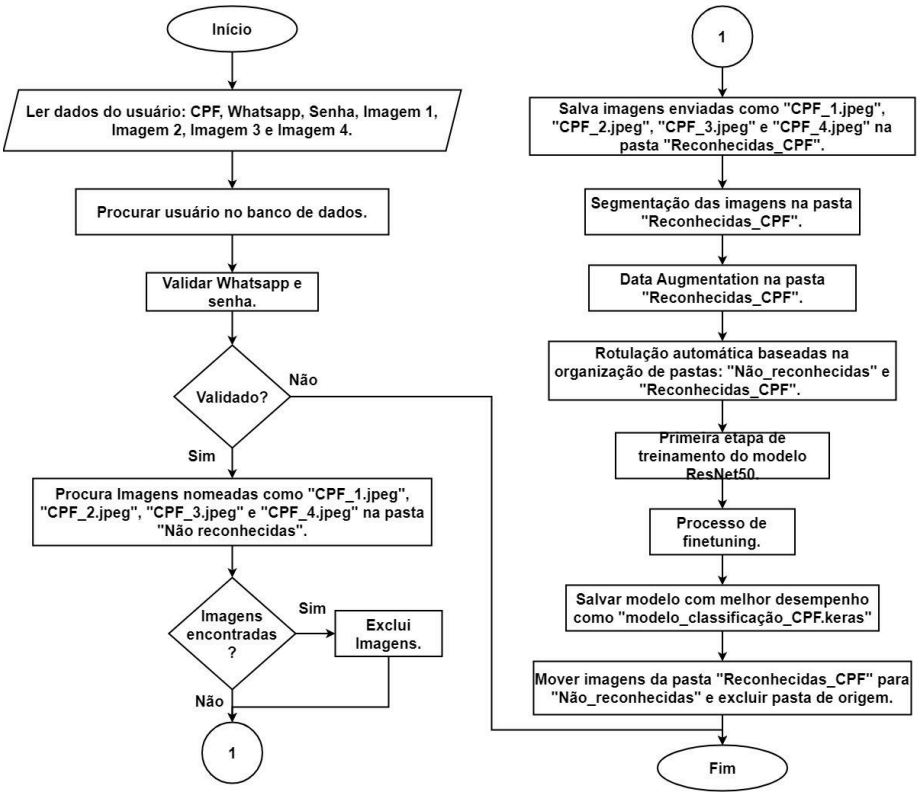
Figura 17 - Fluxograma da rota de validação e liberação.



Fonte: Elaborada pelo autor.

A rota de retreinamento permite que o usuário envie CPF, senha, número de telefone e quatro novas imagens para atualizar o modelo de classificação. O cadastro do usuário é localizado pelo CPF, e após a validação da senha e do número de telefone, as novas imagens são salvas e segmentadas, e o processo de treinamento é repetido, substituindo o modelo antigo por um novo, ajustado com as novas imagens. O fluxograma com as etapas realizadas para realizar o retreinamento do modelo de classificação do usuário está na Figura 18.

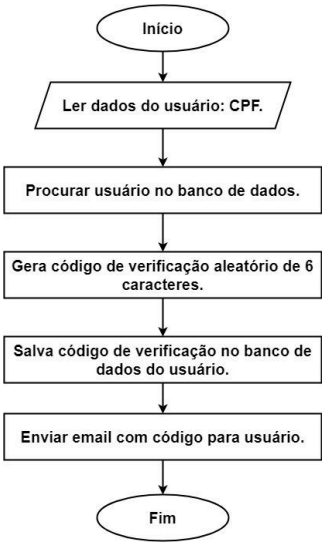
Figura 18 - Fluxograma da rota de retreinamento.



Fonte: Elaborada pelo autor.

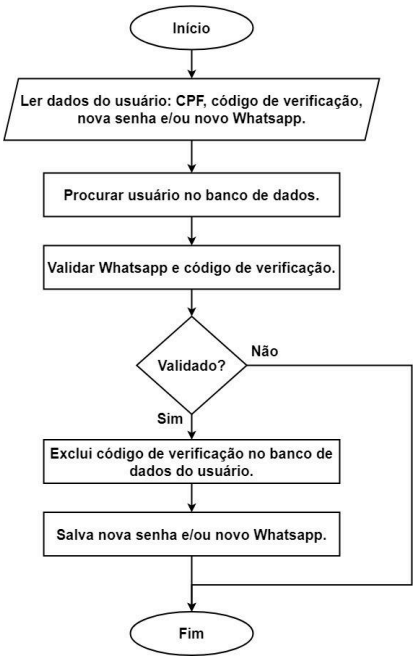
A rota de solicitação de redefinição de senha ou troca de número de telefone recebe apenas o CPF do usuário. Um código de verificação de seis caracteres é gerado, salvo no cadastro do usuário (no campo "código de verificação") e enviado para o e-mail cadastrado do usuário. A rota de redefinição permite que o usuário envie o CPF, o código de verificação recebido por e-mail, uma nova senha e um novo número de telefone. O sistema valida o código de verificação, limpa o campo de código e salva a nova senha e o novo número de telefone. O fluxograma de solicitação de redefinição de senha ou troca de número de telefone está na Figura 19, enquanto o fluxograma da rota de redefinição está na Figura 20.

Figura 19 - Fluxograma da rota de solicitação de redefinição de senha ou troca de número de telefone.



Fonte: Elaborada pelo autor.

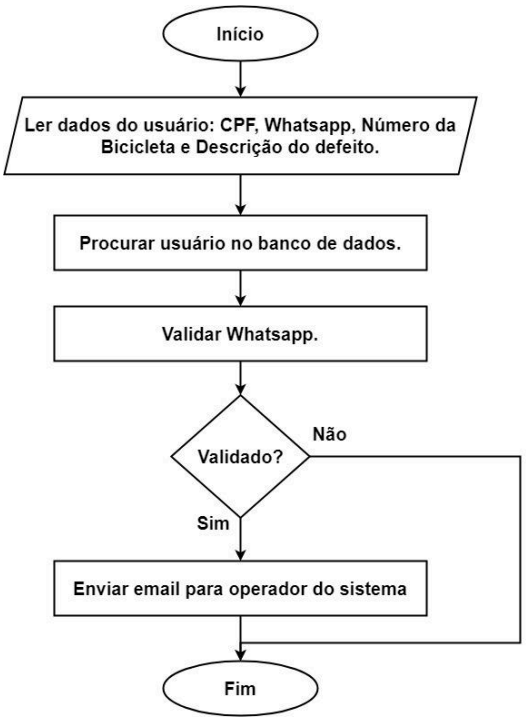
Figura 20 - Fluxograma da rota de redefinição.



Fonte: Elaborada pelo autor.

Por fim, a rota de reporte de defeitos exemplificada por meio do fluxograma presente na Figura 21, que permite que o usuário informe CPF, número de telefone, número da bicicleta e uma descrição do defeito. O sistema verifica o cadastro do usuário pelo CPF e valida o número de telefone. Se tudo estiver correto, um e-mail com as informações do defeito é enviado ao operador do sistema para que o problema seja registrado e solucionado.

Figura 21 - Fluxograma da rota de reporte de defeitos.



Fonte: Elaborada pelo autor.

Essas rotas foram desenvolvidas com o propósito de automatizar o processo de cadastro, validação e operação das bicicletas elétricas, garantindo que todas as interações do usuário com o sistema sejam eficientes, seguras e monitoradas de forma centralizada.

3.7 Desenvolvimento do Robô de WhatsApp

O desenvolvimento do robô de WhatsApp foi realizado em Python utilizando o framework FastAPI, devido ao seu suporte nativo para comportamento assíncrono, o que permite gerenciar múltiplos usuários simultaneamente de maneira eficiente. O robô recebe as mensagens dos usuários por meio de requisições *POST* e utiliza um dicionário para armazenar o estado atual de cada usuário, sendo que a chave do dicionário é o número de WhatsApp do usuário. Esse dicionário armazena tanto a variável de estado (indicando em qual fase do processo o usuário se encontra) quanto os dados necessários para fazer requisições ao sistema principal. Dessa forma, o consegue acompanhar o progresso de cada usuário e realizar ações apropriadas de acordo com o estágio atual de interação [6].

O funcionamento do pode ser comparado a um sistema de navegação por menus, semelhante a um jogo, onde o usuário faz escolhas entre opções apresentadas com base no estado atual em que ele se encontra. Essas escolhas determinam o próximo estado do usuário até que o processo seja concluído e uma requisição seja enviada à API do sistema principal. O principal objetivo do robô é evitar a necessidade de instalação de aplicativos adicionais, ao mesmo tempo em que coleta e organiza todos os dados necessários para que o usuário realize operações no sistema principal de maneira simples e eficiente.

Por exemplo, no processo de cadastro, o usuário começa no estado inicial, onde recebe uma mensagem de boas-vindas e uma lista de opções para continuar. Quando o usuário seleciona a opção de cadastro, o estado é alterado para "cadastro_nome", momento em que o robô solicita o nome do usuário. Após receber o nome, o estado é alterado para "cadastro_sobrenome", e continua pedindo as demais informações (sobrenome, CPF, CEP, etc.), mudando de estado a cada interação. Quando todos os dados necessários são coletados, o robô envia uma requisição para a rota de cadastro na API do sistema principal. Em seguida, o estado do usuário é redefinido para o estado inicial e os dados temporários armazenados no dicionário são apagados, garantindo que as informações não sejam retidas desnecessariamente.

Para o gerenciamento das mensagens do WhatsApp, o robô utiliza o serviço Twilio, uma plataforma de comunicação que permite o envio e recebimento de mensagens de maneira automatizada. O Twilio atua como intermediário, gerenciando as mensagens recebidas e

enviadas pelos usuários e fazendo as requisições à API do robô de WhatsApp, permitindo que a ferramenta interprete as mensagens e responda adequadamente. Além disso, o Twilio garante que o sistema funcione de forma escalável e confiável, mesmo com múltiplos usuários interagindo ao mesmo tempo [28].

A escolha por utilizar dicionários para armazenar as informações dos usuários temporariamente foi motivada pela simplicidade e eficiência desse método. Um dicionário, nesse contexto, é uma estrutura de dados que armazena pares de chave-valor, onde a chave é o número de WhatsApp do usuário e o valor é um conjunto de informações e o estado atual da interação. Essa abordagem foi escolhida para evitar o uso de um banco de dados, que seria desnecessário, uma vez que os dados do usuário são armazenados por curtos períodos de tempo e são descartados após o término do processo. Assim, o uso de dicionários torna o sistema mais leve e ágil, garantindo que o fluxo de informações seja dinâmico e eficiente.

Em resumo, o robô de WhatsApp foi desenvolvido para proporcionar uma experiência fluida e sem a necessidade de aplicativos adicionais, guiando o usuário por meio de interações simples e organizadas, coletando e processando os dados necessários para as operações no sistema principal de maneira eficiente e temporária.

4 Resultados e Discussões

A seção de Resultados e Discussões apresenta a análise detalhada dos experimentos realizados, comparando os modelos de segmentação e classificação de imagens, além de avaliar o desempenho geral do sistema principal e do robô de WhatsApp. Inicialmente, serão discutidos os resultados dos modelos de segmentação, como FCN, U-Net e DeepLabV3+, destacando os principais critérios de avaliação, como a precisão na detecção e segmentação das faces. A partir dessa comparação, será escolhido o modelo que apresentou o melhor desempenho para ser utilizado nas próximas etapas do projeto.

Em seguida, serão analisados os resultados dos modelos de classificação, incluindo as arquiteturas ResNet50, EfficientNetB3, MobileNet e VGG, treinadas para realizar a identificação binária dos usuários. Os resultados de cada modelo serão comparados com base na acurácia e eficiência, permitindo a seleção do modelo mais adequado para o reconhecimento facial no sistema proposto.

Por fim, será apresentado o desempenho do sistema principal, assim como o funcionamento do bot de WhatsApp, que foi projetado para guiar e coletar as informações dos usuários de forma interativa. Cada parte do sistema será avaliada com base em sua funcionalidade e eficiência, buscando validar a integração completa entre os componentes e a satisfação dos objetivos do projeto.

4.1 Modelos de Segmentação

Nesta seção, são apresentados os resultados e comparações das arquiteturas de segmentação de imagens utilizadas no projeto: FCN, U-Net e DeepLabV3+. Cada um desses modelos foi treinado através de um mesmo processador e a partir dos mesmos parâmetros, com variação apenas na arquitetura. A comparação dos modelos será realizada com a segmentação de 100 imagens de indivíduos que apresentam diferentes características, como tons de pele, condições de iluminação, tipos de cabelo, inclinações faciais, expressões e ambientes variados. A análise dos resultados será baseada em métricas como precisão e

perdas registradas durante o treinamento e a validação, além da avaliação qualitativa das segmentações geradas por cada modelo.

A análise inicia-se pelos valores de precisão e perdas registrados nos períodos de treinamento e validação de cada modelo. A precisão indica a taxa de acertos do modelo na identificação correta dos pixels faciais, enquanto as perdas medem a discrepância entre a previsão do modelo e as rotulações reais. Quanto menor o valor de perda, mais próximo o modelo está do resultado desejado, o que é essencial para minimizar erros e maximizar a acurácia na segmentação. Outro ponto importante a ser considerado é a quantidade de imagens em que os modelos falharam, ou seja, as segmentações onde o modelo não conseguiu identificar a face corretamente.

Entretanto, o critério mais relevante para a escolha do modelo final será a avaliação visual dos resultados. Essa análise permitirá identificar qual modelo segmenta as faces com maior precisão, excluindo objetos externos e mantendo o foco exclusivamente nas áreas de interesse, como a região do rosto. A avaliação visual é fundamental para garantir que o modelo escolhido forneça segmentações limpas e utilizáveis em aplicações futuras, como a classificação facial, que depende diretamente da qualidade das segmentações.

4.1.1 FCN (*Fully Convolutional Networks*)

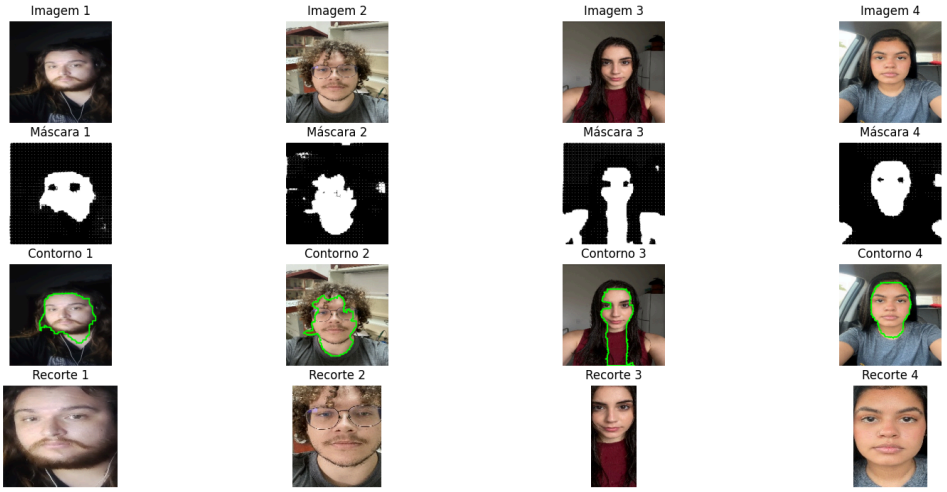
O modelo FCN apresentou resultados interessantes, especialmente considerando sua simplicidade e baixo custo computacional. Com um tempo médio de segmentação em torno de 62 milissegundos por imagem, o FCN demonstrou ser uma opção mais rápida e com menor exigência de processamento em comparação com modelos mais complexos [9].

No entanto, as segmentações realizadas pelo modelo mostraram limitações na precisão espacial. Em muitos casos, o FCN excluiu áreas importantes da face, como olhos e outras regiões essenciais, incluindo partes do corpo como braços que aparecem em algumas imagens. A previsão de precisão alcançada durante o treinamento foi de 84,54%, enquanto na validação o valor foi ligeiramente superior, atingindo 85,06%. Já as perdas previstas foram de 22,83% durante o treinamento e de 18,54% na validação. Esses valores de precisão indicam uma correspondência razoável entre os pixels rotulados manualmente como face e aqueles

previstos pelo modelo. No entanto, é importante ressaltar que essa métrica não garante um recorte preciso.

Como observado na Figura 22, o modelo frequentemente incluiu o pescoço como parte da face, indicando uma dificuldade do FCN em distinguir corretamente os limites faciais em algumas imagens. Essa limitação compromete a qualidade dos recortes finais, mesmo nos casos onde a precisão quantitativa parece aceitável. Quando testado em um conjunto de 100 imagens, o FCN apresentou erros significativos em 27 imagens, com problemas de definição nas demais, devido à inclusão do pescoço como parte do rosto.

Figura 22 - Mostra de imagens segmentadas através da arquitetura FCN.



Fonte: Elaborada pelo autor.

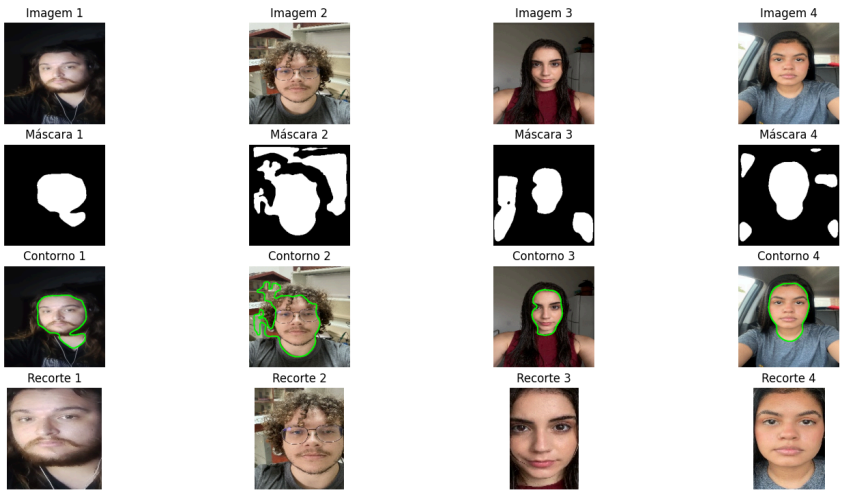
Embora o FCN ofereça vantagens em termos de velocidade e eficiência, os resultados demonstram que sua simplicidade estrutural impacta negativamente a precisão das segmentações, sugerindo que modelos mais complexos podem ser necessários para alcançar uma segmentação facial precisa.

4.1.2 U-Net

O modelo U-Net apresentou exigências maiores em termos de processamento, com uma média de 254 milissegundos por segmentação, devido à sua estrutura mais complexa e profundidade maior em relação ao FCN [9, 12]. Durante o treinamento, o U-Net alcançou uma precisão prevista de 92,47%, mas houve uma queda considerável para 80,95% na validação, sugerindo que o modelo pode ter dificuldades de generalização fora do conjunto de treino. A perda também apresentou um aumento expressivo, passando de 15,39% no treinamento para 30,94% na validação, indicando uma tendência ao sobreajuste.

Esse comportamento pode ser explicado pela própria arquitetura do U-Net, que se baseia em um formato de codificador-decodificador com saltos de conexões para preservar detalhes de alta resolução nas saídas. Essa estrutura faz com que o U-Net se concentre em características como contornos e tons de cor para definir regiões de interesse, o que é vantajoso em muitos casos, mas problemático em imagens onde o fundo apresenta tons semelhantes ao tom de pele ou em condições de alta luminosidade [12]. Como evidenciado na Figura 23, o modelo mostrou dificuldades ao tentar segmentar imagens com fundos de cor similar ao tom de pele, além de incluir o pescoço como parte da face, semelhante ao problema observado com o FCN.

Figura 23 - Mostra de imagens segmentadas através da arquitetura U-NET.



Fonte: Elaborada pelo autor.

Nos testes realizados com um conjunto de 100 imagens, a U-Net foi capaz de segmentar corretamente apenas 46 imagens, mas em todas elas o pescoço foi incluído como parte da face. Esse comportamento pode ser atribuído à característica da U-Net de capturar informações de contorno em larga escala, o que leva à inclusão de áreas adjacentes, como o pescoço, especialmente em imagens onde há uma continuidade de tons entre essas regiões [12].

Embora o U-Net ofereça maior precisão em alguns casos, a dependência da arquitetura em características visuais, como contornos e cores, limita sua capacidade de segmentação em situações complexas e reflete uma necessidade de ajuste adicional para segmentar faces de maneira mais precisa e sem a inclusão de áreas externas indesejadas.

4.1.3 DeepLabV3+

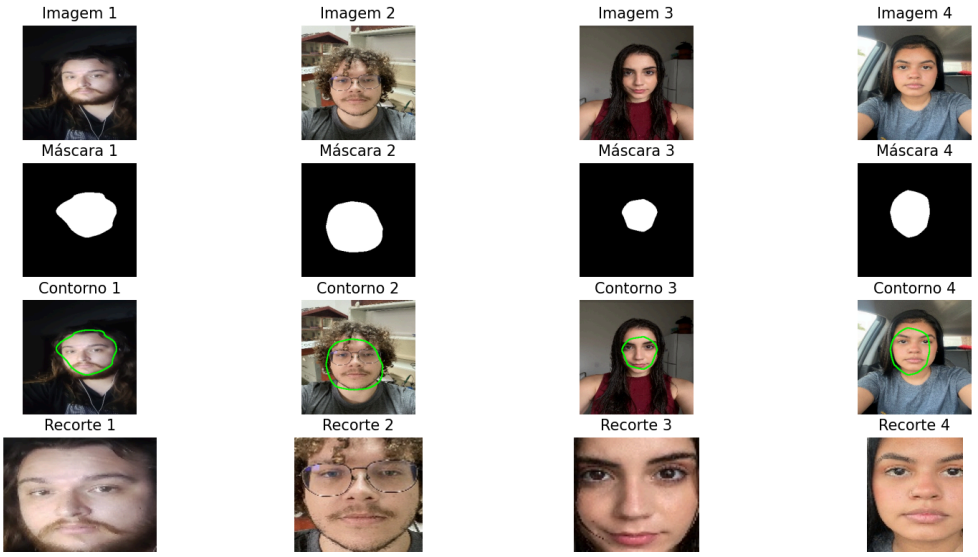
A arquitetura DeepLabV3+ apresentou um tempo médio de segmentação de 510 milissegundos por imagem, o que reflete sua alta demanda de poder de processamento devido à complexidade de sua estrutura. Durante o treinamento, o modelo obteve uma precisão prevista de 94,57%, mantendo um desempenho semelhante na validação com uma precisão de 94,65%. As perdas foram relativamente baixas, registrando 12,91% durante o treinamento e 12,19% na validação, o que indica uma boa capacidade de generalização sem sinais de sobreajuste significativo.

Esse desempenho é resultado da estrutura avançada do DeepLabV3+, que utiliza um módulo de Agrupamento Piramidal Espacial combinado com uma arquitetura codificador-decodificador. O Agrupamento Piramidal Espacial permite que o modelo capture informações em várias escalas por meio de diferentes taxas de dilatação, ampliando o campo de visão da rede sem perder detalhes nas áreas de interesse. Isso é essencial para garantir uma segmentação precisa, capturando detalhes finos, como os contornos da face, sem incluir áreas indesejadas como o pescoço ou elementos de fundo [13, 14].

A estrutura codificador-decodificador da DeepLabV3+ complementa o Agrupamento Piramidal Espacial, garantindo que o codificador capture informações semânticas em alta resolução, enquanto o decodificador refina os contornos e realça a precisão da segmentação.

Essa construção permite que o modelo exclua eficientemente áreas externas, mesmo em situações desafiadoras, como fundos de tons semelhantes à pele ou regiões de alta iluminação, como ilustrado na Figura 24 [13, 14].

Figura 24 - Mostra de imagens segmentadas através da arquitetura DeepLabV3+.



Fonte: Elaborada pelo autor.

Nos testes realizados em um conjunto de 100 imagens, o DeepLabV3+ falhou em apenas 7 imagens, demonstrando a capacidade do modelo de centralizar a face e eliminar regiões externas. Essa combinação de módulos permite um recorte bem focado, mantendo apenas as regiões da face, o que o torna altamente preciso para essa tarefa de segmentação facial.

4.1.5 Comparação entre Modelos de Segmentação

A comparação entre os três modelos de segmentação — FCN, U-Net e DeepLabV3+ — evidencia diferenças significativas em termos de precisão, perda e assertividade, refletindo diretamente no desempenho e na qualidade dos recortes faciais, essenciais para a etapa

subsequente de classificação binária. Conforme demonstrado na Tabela 1, o DeepLabV3+ apresentou a maior precisão e a menor perda tanto durante o treinamento quanto na validação, com valores de 94,57% e 94,65% de precisão, respectivamente, enquanto o FCN e o U-Net apresentaram variações maiores entre os estágios de treinamento e validação, indicando uma tendência ao sobreajuste e, consequentemente, uma capacidade limitada de generalização em dados de teste.

Tabela 1 - Comparação entre modelos de segmentação.

Comparação entre modelos de segmentação			
Arquitetura	FCN	U-NET	DeepLabV3+
Precisão durante treinamento	84,54%	92,47%	94,57%
Precisão durante validação	85,06%	80,95%	94,65%
Perdas durante treinamento	22,83%	15,39%	12,91%
Perdas durante validação	18,54%	30,94%	12,19%
Taxa de assertividade nos recorte	73,00%	45,00%	93,00%
Tempo médio de segmentação	62 milissegundos	254 milissegundos	510 milissegundos

Fonte: Elaborada pelo autor.

Além dos valores de precisão, a assertividade do recorte facial foi um fator determinante na escolha do modelo. O FCN, apesar de ser o mais rápido, apresentou falhas consistentes na segmentação, incluindo áreas do pescoço e excluindo frequentemente regiões faciais essenciais, como os olhos. Já o U-Net, embora mais preciso que o FCN, também apresentou dificuldades em segmentar imagens com fundos de cor semelhante ao tom de pele, além de incluir o pescoço em várias segmentações. Em contrapartida, o DeepLabV3+ mostrou-se capaz de focar exclusivamente na face, ignorando com precisão regiões externas e preservando contornos essenciais, como ilustrado na Figura 25, que apresenta uma comparação visual das segmentações geradas por cada modelo. Nos testes com 100 imagens, o DeepLabV3+ apresentou apenas 7 falhas, contra 27 do FCN e 54 do U-Net.

Figura 25 - Comparação entre modelos de segmentação.

Fonte: Elaborada pelo autor.

A escolha do DeepLabV3+ como modelo de segmentação para o projeto justifica-se pela importância de uma segmentação precisa e centralizada para a etapa de classificação binária das faces. Para que o classificador possa realizar uma identificação confiável, é essencial que a área de interesse — a face — seja isolada de forma consistente, sem incluir elementos externos que possam comprometer a acurácia do modelo de classificação. Como a etapa de classificação depende diretamente da qualidade dos recortes gerados pelo modelo de segmentação, o investimento em uma arquitetura mais complexa e exigente, como o DeepLabV3+, se justifica pela melhoria na precisão e na assertividade dos recortes.

Assim, ao garantir que a área segmentada corresponde com exatidão à face, o DeepLabV3+ contribui para um desempenho mais robusto e confiável na identificação dos usuários. Esse modelo, portanto, foi escolhido para a segmentação, assegurando que a etapa seguinte de classificação binária seja realizada com uma base de dados consistente e de alta qualidade, essencial para o sucesso do sistema como um todo.

4.2 Modelos de Classificação

Nesta seção, são apresentados os resultados e comparações das arquiteturas de classificação de imagens utilizadas no projeto: ResNet50, VGG, MobileNet e EfficientNetB3. Cada um desses modelos foi treinado através do mesmo processador para realizar a identificação binária de um usuário específico, diferenciando entre o usuário-alvo e outras pessoas. A análise dos resultados será realizada com base em métricas de precisão e perda durante o treinamento e validação, além do tempo de treinamento e de classificação de cada modelo.

A avaliação inclui também a assertividade dos modelos treinados em um conjunto de teste composto por 100 imagens segmentadas previamente pelo modelo DeepLabV3+. Esse conjunto foi projetado para testar a capacidade dos modelos em condições variadas, incluindo diferentes angulações de face e variações de iluminação. Das 100 imagens utilizadas para validação, apenas 6 imagens pertencem ao usuário-alvo, para o qual o modelo foi treinado, enquanto as demais contêm rostos de outras pessoas.

Essa comparação permitirá avaliar a eficiência dos modelos em situações reais, identificando a arquitetura mais precisa e robusta para a etapa de classificação binária do sistema, que depende da assertividade do modelo na distinção entre o usuário e outros indivíduos em condições variadas.

4.2.1 ResNet50 (Residual Networks)

A arquitetura ResNet50 apresentou excelente desempenho em termos de precisão e tempo de resposta. O tempo médio de treinamento foi de 7 minutos, e durante o processo de classificação o modelo demonstrou rapidez, com um tempo médio de 55 milissegundos por imagem, atendendo aos requisitos de resposta rápida e precisa.

Nas métricas de desempenho, a ResNet50 obteve uma precisão final de 96,91% e uma perda final de 20,87% durante o treinamento, indicando que o modelo aprendeu com eficácia as características da classe alvo. Durante a validação, a precisão foi de 100,00%, com uma

perda mínima de 0,66%, o que sugere uma excelente capacidade de generalização, sem sinais de sobreajuste.

Nos testes com um conjunto de 100 imagens, contendo apenas 6 pertencentes ao usuário-alvo, o modelo manteve os 100,00% de assertividade, identificando corretamente as imagens do usuário e classificando de forma precisa as imagens de outras pessoas. Esse resultado demonstra que a ResNet50 é robusta para o reconhecimento facial, mantendo alta precisão mesmo em condições variadas de iluminação e angulação, além de operar com uma velocidade que viabiliza seu uso em tempo real.

4.2.2 VGG (*Visual Geometry Group Network*)

A arquitetura VGG apresentou um bom desempenho no treinamento e na classificação, ainda que com um tempo de processamento mais elevado em comparação a outros modelos. O tempo médio de treinamento foi de 9 minutos e o tempo médio de classificação de cada imagem ficou em torno de 110 milissegundos. Apesar de ser um pouco mais lento, o modelo atendeu aos requisitos de precisão e assertividade necessários para a tarefa.

Durante o treinamento, a VGG alcançou uma precisão final de 95,88% e uma perda de 11,22%, indicando que o modelo foi eficaz em capturar as características da classe alvo. Na validação, a precisão aumentou para 97,73%, com uma perda mínima de 3,02%, sugerindo que o modelo conseguiu generalizar bem o aprendizado, com baixo risco de sobreajuste.

Nos testes com um conjunto de 100 imagens, contendo apenas 6 pertencentes ao usuário-alvo, o modelo manteve uma assertividade de 100,00%, identificando corretamente as imagens do usuário e classificando com precisão as imagens de outras pessoas. Esses resultados indicam que a arquitetura VGG é robusta para o reconhecimento facial, oferecendo alta precisão e mantendo sua eficácia em condições diversas de iluminação e angulação. Mesmo com um tempo de resposta um pouco mais alto, o desempenho em termos de precisão e assertividade torna a VGG uma escolha confiável para aplicações onde a robustez do reconhecimento facial é essencial.

4.2.3 MobileNet

A arquitetura MobileNet apresentou um excelente desempenho em termos de velocidade, tanto no treinamento quanto na classificação. Com um tempo médio de treinamento de 2 minutos e um tempo médio para classificar cada imagem de 31 milissegundos, a MobileNet mostrou-se altamente eficiente e de baixo custo computacional, atendendo a cenários que exigem rápida resposta.

Durante o treinamento, o modelo alcançou uma precisão final de 97,94% e uma perda de 11,99%, demonstrando uma forte capacidade de aprendizado para a classe alvo. Na validação, a precisão foi de 97,73%, com uma perda de 7,46%, o que indica um bom desempenho na generalização dos dados, embora a perda mais elevada na validação sugira uma leve tendência a erros em imagens desconhecidas.

Nos testes realizados com 100 imagens, das quais apenas 6 pertenciam ao usuário-alvo, a MobileNet obteve uma assertividade de 91,00%. Embora tenha classificado corretamente a maioria das imagens, o modelo apresentou uma taxa de erro superior em relação a outras arquiteturas, especialmente em imagens com variações de iluminação e angulação. Esse resultado aponta que, apesar de ser altamente rápida e eficiente, a MobileNet pode ter limitações em contextos que exigem máxima precisão no reconhecimento facial, mostrando-se mais vulnerável a condições variáveis nos dados de entrada.

4.2.4 EfficientNetB3

A arquitetura EfficientNetB3 apresentou um equilíbrio entre precisão e velocidade, com um tempo médio de treinamento de 6 minutos e uma média de 65 milissegundos para classificar cada imagem. Esses valores indicam que o modelo é relativamente rápido, mantendo uma boa relação entre custo computacional e desempenho.

Durante o treinamento, a EfficientNetB3 alcançou uma precisão final de 88,66% e uma perda de 54,16%, valores que demonstram a capacidade do modelo de aprender as características da classe alvo. Na validação, a precisão aumentou para 97,73%, com uma

perda final de 6,95%, sugerindo uma boa generalização dos dados e um baixo risco de sobreajuste.

Nos testes realizados com um conjunto de 100 imagens, contendo apenas 6 pertencentes ao usuário-alvo, a EfficientNetB3 atingiu uma assertividade de 93,00%. Embora o modelo tenha sido bem-sucedido na maioria das classificações, ele apresentou uma leve taxa de erro em relação a outras arquiteturas. Esses erros ocorreram principalmente em imagens com variações de iluminação e angulação.

Os resultados indicam que a EfficientNetB3 é uma opção eficiente para o reconhecimento facial, oferecendo uma combinação sólida de precisão e rapidez, mas com algumas limitações em condições desafiadoras.

4.2.5 Comparação entre Modelos de Classificação

A comparação entre as quatro arquiteturas de classificação — ResNet50, VGG, MobileNet e EfficientNetB3 — evidencia diferenças claras em termos de precisão, perdas, e tempos de treinamento e classificação, conforme ilustrado na Tabela 2. Esses dados permitem avaliar o desempenho de cada modelo, considerando tanto sua eficácia em identificar corretamente o usuário-alvo quanto sua velocidade e custo computacional.

Tabela 2 - Comparação entre modelos de classificação.

Comparação entre modelos de classificação				
Arquitetura	ResNet50	VGG	MobileNet	EfficientNetB3
Tempo médio de treinamento	7 minutos	9 minutos	2 minutos	6 minutos
Tempo médio de classificação	55 milissegundos	110 milissegundos	31 milissegundos	65 milissegundos
Precisão durante treinamento	96,91%	95,88%	97,94%	88,66%
Precisão durante validação	100,00%	97,73%	97,73%	97,73%
Perdas durante treinamento	20,87%	11,22%	11,99%	54,16%
Perdas durante validação	00,66%	03,02%	07,46%	06,95%
Teste de assertividade	100,00%	100,00%	91,00%	93,00%

Fonte: Elaborada pelo autor.

A ResNet50 apresentou uma precisão consistente e uma assertividade de 100,00% no teste final com 100 imagens, destacando-se pela capacidade de identificar corretamente o usuário-alvo em diversas condições, incluindo variações de iluminação e angulação. Com um tempo médio de treinamento de 7 minutos e 55 milissegundos por classificação, o modelo demonstrou um excelente equilíbrio entre precisão e tempo de processamento. A VGG também obteve um desempenho de 100,00% de assertividade, mas com um tempo de classificação mais elevado (110 milissegundos) e uma perda de validação de 3,02%, o que representa um custo de processamento superior.

A MobileNet, por outro lado, destacou-se como o modelo mais rápido, com 2 minutos de treinamento e 31 milissegundos para classificar cada imagem, porém, sua assertividade foi de 91,00% no teste com 100 imagens, mostrando limitações em termos de precisão em

situações variadas. A EfficientNetB3 obteve uma precisão sólida de 97,73% na validação e 93,00% de assertividade no teste final, apresentando um tempo médio de classificação de 65 milissegundos e um tempo de treinamento de 6 minutos, o que a coloca entre as arquiteturas mais equilibradas em velocidade e precisão, embora com uma ligeira perda de assertividade em condições de teste.

Considerando esses resultados, o modelo escolhido para a classificação final foi a ResNet50, que, apesar de apresentar um tempo de treinamento e classificação maior do que alguns outros modelos, mostrou o melhor equilíbrio entre precisão e exigência de processamento. A assertividade de 100,00% no teste final confirma sua robustez e precisão para a tarefa de reconhecimento facial, mesmo em condições variadas. Após essa análise comparativa, o modelo ResNet50 foi submetido a mais 19 treinamentos, cada um com um usuário-alvo distinto, e os resultados obtidos replicaram o desempenho anterior, com pequenas variações nos tempos de treinamento e classificação, mas alcançando no mínimo 98% de assertividade no teste com 100 imagens. Esses resultados comprovam a confiabilidade e a consistência da ResNet50 para a aplicação proposta.

4.3 Sistema Principal

Após a seleção dos modelos DeepLabV3+ para segmentação e ResNet50 para classificação, ambos foram integrados ao sistema principal, que foi implementado em um servidor para disponibilizar suas rotas e funcionalidades para testes completos. Esses testes foram realizados utilizando o Postman, uma ferramenta que facilita o envio de requisições HTTP (*GET*, *POST*, *PUT*, *DELETE*) para APIs, permitindo simular e verificar o comportamento das rotas de um sistema em desenvolvimento. Através do Postman, cada rota do sistema principal pôde ser testada e analisada, assegurando seu correto funcionamento e resposta dentro dos parâmetros esperados.

A rota de cadastro foi um dos processos principais testados. Esse processo utiliza o modelo pré-treinado de segmentação para recortar as imagens faciais enviadas pelo usuário e treina o modelo de classificação individual. Devido à diferença entre o poder de processamento do ambiente de testes e do servidor final, o tempo médio para concluir o

cadastro de um usuário foi de aproximadamente 17 minutos (média obtida a partir do cadastro de 10 usuários). Durante o cadastro, a integração com o banco de dados, que armazena informações do usuário, mostrou-se eficaz e funcional, com todas as informações sendo registradas corretamente.

A rota de validação, responsável pela verificação de identidade do usuário e pela solicitação de liberação de doca, também se mostrou plenamente funcional. O tempo médio para todo o processo — desde a validação do usuário até o envio do e-mail para o operador — foi de cerca de 10 segundos. Esse intervalo inclui o uso do modelo de segmentação e do modelo de classificação, adaptados para rodar em um ambiente com poder de processamento menor, o que justifica a duração do processo.

As demais rotas, incluindo redefinição de senha, troca de número de celular e reporte de defeitos, foram totalmente funcionais, com tempos médios de resposta de 2 segundos para cada operação. A única exceção foi a rota de retreinamento, que apresenta um tempo semelhante ao da rota de cadastro, pois envolve o treinamento de um novo modelo de classificação para o usuário.

Com os testes realizados e a integração bem-sucedida entre o modelo de segmentação DeepLabV3+, o modelo de classificação ResNet50 e o sistema principal, conclui-se que o sistema atingiu o objetivo proposto e está completamente funcional. Ele permite um fluxo eficiente e seguro para o gerenciamento do cadastro, validação e operação de liberação de bicicletas elétricas, garantindo precisão e agilidade em todos os processos.

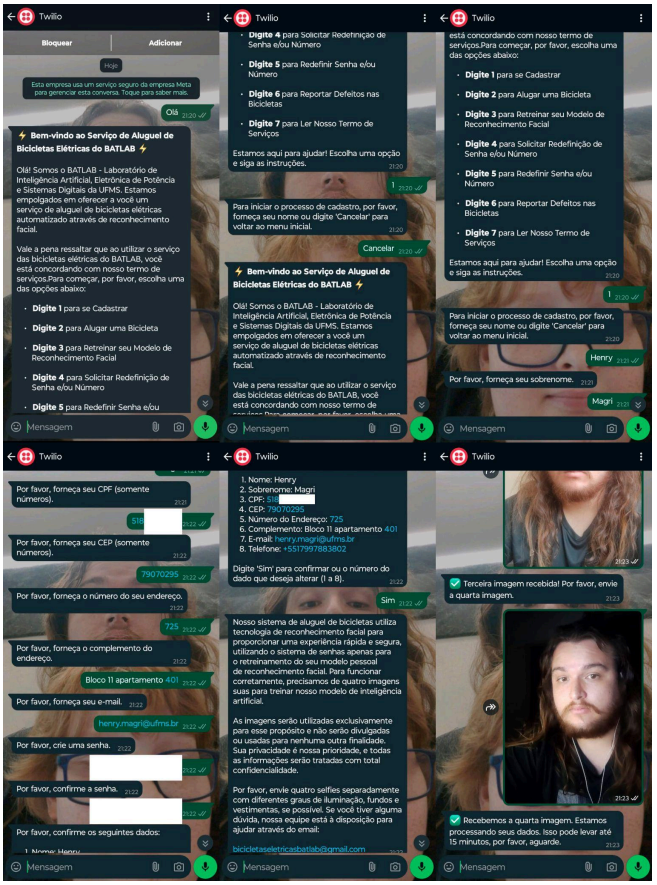
4.4 Bot de WhatsApp

Após a alocação do robô de WhatsApp em um servidor e a vinculação com o serviço de comunicação do Twilio, foram realizados testes para validar seu funcionamento e avaliar sua eficácia como ferramenta auxiliar ao sistema principal. Durante esses testes, o robô demonstrou-se totalmente funcional, cumprindo o objetivo de guiar o usuário de forma intuitiva e organizada por meio de estágios, com a finalidade de facilitar a realização dos processos, como cadastro, validação e solicitação de serviços, oferecidos pelo sistema principal.

O design do robô foi pensado para simplificar o uso e tornar os processos mais acessíveis, guiando o usuário em cada etapa e garantindo que todas as informações necessárias sejam coletadas e encaminhadas ao sistema de forma correta. Essa estrutura em estágios permite que o usuário avance no fluxo de interação a partir das opções apresentadas, com o robô ajustando o diálogo de acordo com o contexto, o que proporciona uma experiência de fácil compreensão e evita a necessidade de um aplicativo adicional.

Na Figura 26, apresentamos um exemplo de interação, ilustrando o processo de cadastro de um novo usuário, desde a saudação inicial até a coleta das informações e confirmação final, o que demonstra o funcionamento prático do robô. Esse exemplo evidencia a eficácia do robô em guiar o usuário por todas as etapas do cadastro, garantindo que todos os dados necessários sejam capturados para o sistema principal.

Figura 26 - Funcionamento do robô de Whatsapp integrado com o sistema.



Fonte: Elaborada pelo autor.

5 Conclusão

Neste capítulo, apresentam-se as Considerações Finais, que sintetizam os principais resultados e contribuições obtidos ao longo do desenvolvimento e validação do sistema de empréstimo de bicicletas elétricas, que integra Inteligência Artificial e Internet das Coisas para reconhecimento e segmentação facial. Em seguida, na seção de Trabalhos Futuros, são discutidas oportunidades de aprimoramento e novas possibilidades de pesquisa e desenvolvimento, visando expandir a funcionalidade e a aplicabilidade do sistema em diferentes cenários e acompanhar as inovações tecnológicas emergentes.

5.1 Considerações Finais

Com a finalização do sistema de empréstimo de bicicletas elétricas, foi possível verificar, durante os testes de desenvolvimento e integração, que ele atende aos requisitos de segurança, eficiência e interação estabelecidos no projeto. Através do uso de tecnologias de Inteligência Artificial e Internet das Coisas, o sistema mostrou-se uma ferramenta excelente para o gerenciamento automatizado de serviços de aluguel e empréstimo, além de oferecer uma experiência de usuário interativa e inovadora, diferenciando-se das soluções tradicionais no mercado.

Inicialmente, o sistema será testado com um grupo de aproximadamente 50 estudantes, com o objetivo de identificar possíveis falhas ou brechas que possam ser ajustadas antes da integração completa com as docas de bicicletas. Esse primeiro ciclo de testes, realizado através de um operador, serve como uma etapa fundamental de validação e aperfeiçoamento do sistema. Após essa fase, o sistema estará pronto para ser disponibilizado para um público mais amplo, incluindo alunos e pessoas sem envolvimento direto com o projeto, expandindo sua aplicação e permitindo que uma base maior de usuários aproveite os benefícios dessa tecnologia.

Além de ser uma solução segura, o sistema destaca-se pela interatividade e pelo apelo tecnológico, integrando recursos de IA e IoT, que têm mostrado crescimento contínuo e amplo potencial de aplicação em diversos setores. Com uma interação fluida e intuitiva com o

usuário, o sistema não só automatiza o processo de empréstimo, mas também envolve o usuário de maneira acessível e conectada, criando uma experiência diferenciada.

O sistema também apresenta grande potencial de escalabilidade. As funcionalidades desenvolvidas podem ser facilmente adaptadas para diferentes contextos de empréstimo e aluguel, ou ainda como uma ferramenta de controle inteligente de acesso para outras aplicações. Essa flexibilidade amplia as possibilidades de uso e viabiliza a expansão do sistema para novos mercados e áreas de atuação.

Por fim, o sistema é projetado para um desenvolvimento contínuo. À medida que a base de usuários cresce, o banco de imagens e dados também se expande, tornando os modelos de IA mais robustos e precisos. Esse crescimento orgânico permitirá que os modelos de reconhecimento facial se tornem cada vez mais consistentes, facilitando uma adaptação contínua às características e necessidades dos novos usuários, o que assegura uma solução ainda mais eficaz e completa ao longo do tempo. Testes mais intensivos realizados com os modelos selecionados para o sistema — DeepLabv3+ para segmentação e ResNet50 para classificação — evidenciaram que o desempenho de cada um está intrinsecamente ligado a diferentes fatores. Para o modelo de segmentação, a assertividade aumenta à medida que o conjunto de treinamento inclui mais imagens brutas com fundos variados, mesmo que as fotografias sejam das mesmas pessoas. Por outro lado, o modelo de classificação demonstrou maior dependência de um conjunto diversificado de imagens de rostos, com diferentes características faciais, para aprimorar sua precisão. Esses resultados reforçam a importância de um conjunto de dados balanceado e diversificado para atender às necessidades específicas de cada modelo, assegurando a robustez e a eficácia das soluções de Inteligência Artificial implementadas no sistema.

5.2 Trabalhos Futuros

Nos Trabalhos Futuros, destacam-se as próximas etapas e possibilidades de expansão do sistema, que permitirão aprimorar ainda mais sua funcionalidade e aplicabilidade. Após os testes iniciais, será necessária a integração completa do sistema com as docas, possibilitando a automação completa do processo de empréstimo e devolução de bicicletas. Esse avanço

permitirá a coleta e o monitoramento em tempo real de dados essenciais para a gestão do sistema, além de abrir novas possibilidades para o desenvolvimento de funcionalidades adicionais.

Com a expansão do banco de dados e a integração direta com as docas, o sistema poderá monitorar a disponibilidade das bicicletas em cada local, além de outras informações relevantes, como o nível de carga das baterias. Esse monitoramento permitirá a criação de um histórico completo de empréstimos, incluindo dados como o horário de retirada e devolução das bicicletas, o tempo total de uso e o consumo médio de bateria. Essas informações possibilitarão uma gestão mais detalhada e eficiente dos recursos.

Outro aprimoramento proposto é a implementação de um Sistema de Previsão de Demanda. Utilizando algoritmos de IA, o sistema poderia analisar os dados históricos de uso das bicicletas para prever a demanda em horários específicos e diferentes locais. Essa previsão permitiria uma alocação de bicicletas mais estratégica, otimizando o atendimento nos horários de pico e reduzindo a probabilidade de falta de bicicletas disponíveis.

Além disso, seria vantajoso implementar um sistema de monitoramento e manutenção preventiva. Sensores nas bicicletas, integrados ao sistema de IoT, poderiam monitorar o estado das bicicletas em tempo real e alertar sobre a necessidade de manutenção antes que problemas maiores ocorram. Essa funcionalidade reduziria o tempo de inatividade das bicicletas e garantiria mais segurança e confiabilidade para os usuários.

Uma funcionalidade adicional a ser considerada é a inclusão de um sistema de feedback dos usuários, que permitiria coletar informações sobre a experiência de uso, sugestões de melhoria e notificações de problemas. Esses dados poderiam ser integrados ao sistema para otimizar continuamente o serviço e aumentar a satisfação dos usuários.

A integração com aplicativos de mapas e roteamento também é uma possibilidade interessante. Através de APIs, o sistema poderia fornecer informações em tempo real sobre a localização das docas e indicar as rotas mais eficientes até a doca mais próxima ou menos ocupada. Esse recurso melhoraria a experiência do usuário e incentivaria o uso contínuo do serviço.

Além dessas inovações, um dos pontos críticos para manter a qualidade do sistema será a atualização periódica do modelo de segmentação de faces. De tempos em tempos, será necessário um retreinamento manual do modelo de segmentação, utilizando novos dados para aprimorar sua capacidade de detecção e excluir imagens menos relevantes. A nova versão do modelo deverá substituir a anterior, garantindo que ele se mantenha atualizado e alinhado com as condições reais de uso. O gerenciamento cuidadoso do banco de imagens também será fundamental para evitar o sobreajuste, de modo que o modelo de segmentação e os modelos de classificação se mantenham robustos e confiáveis.

Essas sugestões para trabalhos futuros ampliam as perspectivas do sistema, que poderá evoluir de uma solução específica de empréstimo de bicicletas para uma plataforma robusta de gestão e monitoramento de mobilidade elétrica.

6 Referências Bibliográficas

1. Kavre, Mahesh, Gadekar, Aditya, & Gadhade, Yash. (2019). Internet of Things (IoT): A Survey. In 2019 IEEE Pune Section International Conference (PuneCon) (pp. 1-6). doi:10.1109/PuneCon46936.2019.9105831.
2. Ramaditiya, Achmad, Rahmatia, Suci, Munawar, Aris, & Samijayani, Octarina Nur. (2021). Implementation Chatbot Whatsapp using Python Programming for Broadcast and Reply Message Automatically. In 2021 International Symposium on Electronics and Smart Devices (ISESD) (pp. 1-4). doi:10.1109/ISESD53023.2021.9501523.
3. Maia, Deise Santana. (2014). Detecção e Reconhecimento de Face Utilizando o Matlab (Monografia). UESB - Universidade Estadual do Sudoeste da Bahia. Orientador: Prof. Dr. Roque Mendes Prado Trindade.
4. Shah, Sajjad Hussain, & Yaqoob, Ilyas. (2016). A survey: Internet of Things (IoT) technologies, applications and challenges. In 2016 IEEE Smart Energy Grid Engineering (SEGE) (pp. 381-385). doi:10.1109/SEGE.2016.7589556.
5. Abdul-Qawy, A. S., Pramod, P. J., Magesh, E., & Srinivasulu, T. (2015). The Internet of Things (IoT): An Overview. International Journal of Engineering Research and Applications, 5(12), 71-82. ISSN: 2248-9622.
6. Ramirez, S. (2018). FastAPI Documentation. Acesso em 26 de outubro de 2024. https://fastapi.tiangolo.com.
7. Nagila, S., & Myna, A. N. (2017). Automatic face image annotation using machine learning techniques. International Journal of Current Research, 9(10), 59193-59198.
8. Domingos, P. (2012). A few useful things to know about machine learning. Communications of the ACM, 55(10), 78-87. https://doi.org/10.1145/2347736.2347755.
9. Long, J., Shelhamer, E., & Darrell, T. (2014). Fully convolutional networks for semantic segmentation. CoRR, abs/1411.4038. http://arxiv.org/abs/1411.4038.
10. He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In 2017 IEEE International Conference on Computer Vision (ICCV) (pp. 2980-2988). doi:10.1109/ICCV.2017.322.
11. Hmidani, O., & Ismaili Alaoui, M. (2023). Developing Mask R-CNN framework for real-time object detection. In 2023 6th International Conference on Advanced

Communication Technologies and Networking (CommNet) (pp. 1-8). doi:10.1109/CommNet60167.2023.10365298.

12. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. CoRR, abs/1505.04597. http://arxiv.org/abs/1505.04597.
13. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. CoRR, abs/1802.02611. http://arxiv.org/abs/1802.02611.
14. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(4), 834-848. https://doi.org/10.1109/TPAMI.2017.2699184.
15. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. CoRR, abs/1512.03385. http://arxiv.org/abs/1512.03385.
16. Wan, J., Li, B., Wang, K., Teng, X., Wang, T., & Mao, B. (2024). An Improved ResNet50 for Environment Image Classification. Procedia Computer Science, 242, 1000-1007. https://doi.org/10.1016/j.procs.2024.08.246.
17. Astawa, I. N. G. & Radhitya, M., Ardana, I. W. & Dwiyanto, F. (2021). Face Images Classification using VGG-CNN. Knowledge Engineering and Data Science, 4, 49. doi:10.17977/um018v4i12021p49-54.
18. Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L.-C., Tan, M., Chu, G., Vasudevan, V., Zhu, Y., Pang, R., Adam, H., & Le, Q. (2019). Searching for MobileNetV3. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 1314-1324). doi:10.1109/ICCV.2019.00140.
19. Sinha, D. & El-Sharkawy, M. (2019). Thin MobileNet: An Enhanced MobileNet Architecture. In 2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) (pp. 0280-0285). doi:10.1109/UEMCON47517.2019.8993089.
20. Utami, P., Hartanto, R., & Soesanti, I. (2022). The EfficientNet Performance for Facial Expressions Recognition. In 2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) (pp. 756-762). doi:10.1109/ISRITI56927.2022.10053007.

21. TensorFlow Documentation. Acesso em 26 de outubro de 2024. https://www.tensorflow.org/api_docs.
22. Keras Documentation. Acesso em 26 de outubro de 2024. https://keras.io/api/.
23. Scikit-learn Documentation. Acesso em 26 de outubro de 2024. https://scikit-learn.org/stable/documentation.html.
24. NumPy Documentation. Acesso em 26 de outubro de 2024. https://numpy.org/doc/stable/.
25. LabelMe Documentation. Acesso em 26 de outubro de 2024. https://github.com/wkentaro/labelme/blob/master/README.md.
26. OpenCV Documentation. Acesso em 26 de outubro de 2024. https://docs.opencv.org/master/.
27. PostgreSQL Documentation. Acesso em 26 de outubro de 2024. https://www.postgresql.org/docs/.
28. Twilio Documentation. Acesso em 26 de outubro de 2024. https://www.twilio.com/docs.