
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS INHUMAS
BACHARELADO EM ENGENHARIA DE SOFTWARE

JOÃO VITOR CAMPOS
MAX LOBO MAGALHÃES DE AGUIAR

# MÉTODO PARA A IDENTIFICAÇÃO DE GESTOS DE MÃO USANDO WI-FI SENSING E APRENDIZADO DE MÁQUINA

INHUMAS
3 de novembro de 2025

JOÃO VITOR CAMPOS
MAX LOBO MAGALHÃES DE AGUIAR

# MÉTODO PARA A IDENTIFICAÇÃO DE GESTOS DE MÃO USANDO WI-FI SENSING E APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso, no formato de monografia, apresentado ao curso de Bacharelado em Engenharia de Software, do Instituto Federal de Educação, Ciência e Tecnologia de Goiás - Campus Inhumas, como requisito parcial à obtenção do título de Bacharel em Engenharia de Software.

Orientador: Dr. Victor Hugo Lázaro Lopes

Inhumas

3 de novembro de 2025

Dados Internacionais de Catalogação na Publicação

Campos, João Vitor.

C198 Método para a identificação de gestos de mão usando wi-fi *sensing* e aprendizado de máquina [Manuscrito]. / João Vitor Campos, Max Lobo Magalhães de Aguiar – Inhumas: IFG, 2025.

71 f.: il.

Inclui bibliografia.

Orientador: Prof. Dr. Victor Hugo Lázaro Lopes

Trabalho de Conclusão de Curso em Bacharelado em Engenharia de Software – IFG/Câmpus Inhumas, 2025.

1. Wifi sensing. 2. Detecção de pessoas. 3. Aprendizado por máquina. 4. Reconhecimento de gestos. 5. Aguiar, Max Lobo Magalhães de. 6. Lopes, Victor Hugo Lázaro (orientador). I. Título.

CDD 006.3

Ficha catalográfica elaborada pela bibliotecária
Maria Aparecida Rodrigues de Souza - CRB/1-1497
Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Câmpus Inhumas – Biblioteca Atena

## TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinalada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

**Identificação da Produção Técnico-Científica**

[ ] Tese

[ ] Dissertação

[ ] Monografia – Especialização

[ X ] TCC – Graduação

[ ] Produto Técnico e Educacional - Tipo:

[ ] Artigo Científico

[ ] Capítulo de Livro

[ ] Livro

[ ] Trabalho Apresentado em Evento

**Nome Completo do Autor:**

**Matrícula:**

**Título do Trabalho:**

**Autorização - Marque uma das opções**

[ X ] Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto).
[ ] Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data de 3 de novembro de 2025 (Embargo).
[ ] Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção 2 ou 3, marque a justificativa:

[ ] O documento está sujeito a registro de patente.

[ ] O documento poderá vir a ser publicado como livro, capítulo de livro ou artigo.

[ ] Outra justificativa: Blábla Blábla Blábla

---

Assinatura do Autor

# DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(A) referido(a) autor(a) declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

ii. obteve autorização de quaisquer materiais inclusos no documento de que não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cumpre direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Inhumas/Goiás
Local

3 de novembro de 2025
Data

gov.br Documento assinado digitalmente
JOAO VITOR CAMPOS
Data: 02/11/2025 21:41:41-0300
Verifique em https://validar.iti.gov.br

gov.br Documento assinado digitalmente
MAX LOBO MAGALHAES DE AGUIAR
Data: 03/11/2025 09:41:51-0300
Verifique em https://validar.iti.gov.br

Assinatura do Autor e/ou Detentor dos Direitos Autorais

Documento elaborado pelo coordenador do curso.

INSTITUTO FEDERAL
Goiás

**MINISTÉRIO DA EDUCAÇÃO**
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS INHUMAS

# ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Na presente data realizou-se a sessão pública de defesa do Trabalho de Conclusão de Curso intitulada **Método para a identificação de gestos de mão usando WiFi Sensing e Aprendizado de Máquina**, sob orientação de Victor Hugo Lazaro Lopes , apresentada pelo aluno **João Vitor Campos (20211030180144)** do Curso **Bacharelado em Engenharia de Software (Câmpus Inhumas)**. Os trabalhos foram iniciados às **20:00** do dia **12/09/2025** pelo Professor presidente da banca examinadora, constituída pelos seguintes membros:

- **Victor Hugo Lazaro Lopes** (Presidente)
- **Rogerio Sousa e Silva** (Examinador Interno)
- **Leandro Alexandre Freitas** (Examinador Interno)

A banca examinadora, tendo terminado a apresentação do conteúdo do Trabalho de Conclusão de Curso, passou à argüição do candidato. Em seguida, os examinadores reuniram-se para avaliação e deram o parecer final sobre o trabalho apresentado pelo aluno, tendo sido atribuído o seguinte resultado:

[ X ] Aprovado [ ] Reprovado Nota : 10,0

**Observação / Apreciações:**

Trabalho aprovado.

Proclamados os resultados pelo presidente da banca examinadora, foram encerrados os trabalhos e, para constar, eu **Victor Hugo Lazaro Lopes** lavrei a presente ata que assino juntamente com os demais membros da banca examinadora.

Documento assinado eletronicamente por:

- **Victor Hugo Lazaro Lopes** (940.053.971-15), em **13/09/2025 10:28:07** com chave **7c6c460890a511f0932a005056a537a4**.
- **Rogerio Sousa e Silva** (423.704.421-15), em **13/09/2025 10:28:32** com chave **8b444a6890a511f08c8c005056a537a4**.
- **Leandro Alexandre Freitas** (937.500.991-20), em **14/09/2025 09:16:27** com chave **a36f407e916411f0bc82005056a537a4**.

Este documento foi emitido pelo SUAP. Para comprovar sua autenticidade, faça a leitura do QrCode ao lado ou acesse https://suap.ifg.edu.br/comum/autenticar_documento/ e informe os dados a seguir.

**Tipo de Documento:** Ata de Projeto Final

**Data da Emissão:** 15/09/2025

**Código de Autenticação:** f5d005

# AGRADECIMENTOS

Eu, Max Lobo Magalhães de Aguiar, gostaria de expressar os meus agradecimentos a minha familia e amigos que foram meu suporte durante toda a graduação, em especial a minha psiquiatra Doutora Carolyna de Freitas Vieira que foi o meu suporte médico nos ultimos dois anos. Eu, João Vitor Campos, gostaria de expressar os meu mais sinceros agradecimentos à minha esposa, família, meus amigos e colegas do curso que durante minha jornada foram de grande importância em me incentivar a continuar me empenhando nos estudos mesmo nos momentos de desânimo que são comuns para qualquer pessoa, o suporte de todos foi muito importante para estar aqui apresentando esse trabalho de conclusão de curso. Em conjuntos gostariamos de agraceder ao nosso orientador Dr. Victor Hugo Lázaro Lopes por todo suporte prestado durante o desenvolvimento desse trabalho, e todos os docentes do IFG Campus Inhumas pelo acompanhamento durante a graduação.

# Resumo

A Interação Humano-Computador (IHC) busca interfaces cada vez mais intuitivas, e o reconhecimento de gestos manuais é um campo promissor para essa evolução. Abordagens tradicionais, como as baseadas em visão computacional ou hardware especializado, apresentam limitações de custo, privacidade e dependência técnicas como linha de visão. Este trabalho explora o WiFi Sensing como uma alternativa robusta e não invasiva, utilizando as perturbações em sinais Wi-Fi, conhecidas como Channel State Information (CSI), para classificar gestos. A hipótese central é que modelos de aprendizado de máquina podem alcançar alta acurácia na classificação de gestos a partir de dados CSI.

Para validar esta hipótese, foi desenvolvida uma prova de conceito com uma metodologia evolutiva, estruturada em dois cenários experimentais: o primeiro, utilizando um conjunto de dados público para estabelecer estatisticas em um ambiente complexo, e o segundo, com um conjunto de dados próprio coletado em ambiente controlado para isolar os gestos das mãos. O processamento de dados evoluiu de uma abordagem inicial que tratava cada subportadora de sinal de forma isolada para uma metodologia final que agrega a captura inteira do gesto em um único vetor de características, preservando a correlação entre os canais. A otimização dos modelos foi realizada com o framework Optuna, que permitiu não apenas a busca por hiperparâmetros otimizados, mas também uma análise aprofundada do custo computacional (Work Effort) de cada modelo. Os resultados demonstraram a superioridade da abordagem por captura agregada. No cenário controlado, o modelo SGDClassifier alcançou a maior acurácia, com 88.14%, destacando-se também pela sua excepcional eficiência computacional, com o menor custo de otimização e a convergência mais rápida. Este trabalho conclui que o WiFi Sensing, aliado a pipelines de aprendizado de máquina e otimização inteligente, é uma solução viável e eficaz para o reconhecimento de gestos, oferecendo um balanço positivo entre acurácia, custo e respeito à privacidade.

**Palavras-chave:** reconhecimento de gestos, wifi sensing, channel state information, aprendizado de máquina, otimização de hiperparâmetros.

# Abstract

Human-Computer Interaction (HCI) seeks increasingly intuitive interfaces, and hand gesture recognition is a promising field for this evolution. Traditional approaches, such as those based on computer vision or specialized hardware, present limitations regarding cost, privacy, and line-of-sight requirements. This work explores WiFi Sensing as a robust and non-invasive alternative, utilizing disturbances in Wi-Fi signals, known as Channel State Information (CSI), to classify gestures. The central hypothesis is that machine learning models can achieve high accuracy in classifying gestures from CSI data.

To validate this hypothesis, a proof-of-concept was developed with an evolutionary methodology, structured in two experimental scenarios: the first using a public dataset to establish a benchmark in a complex environment, and the second with a custom dataset collected in a controlled environment to isolate hand gestures. The data processing evolved from an initial approach that treated each signal subcarrier independently to a final methodology that aggregates the entire gesture capture into a single feature vector, preserving the correlation between channels. Model optimization was performed with the Optuna framework, which allowed not only for the search of optimal hyperparameters but also for an in-depth analysis of the computational cost (Work Effort) of each model. The results demonstrated the superiority of the aggregated capture approach. In the controlled scenario, the SGDClassifier model achieved the highest accuracy at 88.14%, also standing out for its exceptional computational efficiency, with the lowest optimization cost and the fastest convergence. This work concludes that WiFi Sensing, combined with machine learning pipelines and intelligent optimization, is a viable and effective solution for gesture recognition, offering a positive balance between accuracy, cost, and respect for privacy.

**Keywords:** gesture recognition, wifi sensing, channel state information, machine learning, hyperparameter optimization.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Lista de Abreviaturas e Siglas

**AM** aprendizado de máquina. 2, 3, 13, 16, 18, 20, 22, 53, 54

**CFR** Resposta de Frequência do Canal, do inglês *Channel Frequency Response*. 6

**CSI** Informação do Estado do Canal, do inglês *Channel State Information*. 2, 3, 5–9, 12, 21, 22, 50, 53–55

**IHM** interação homem-máquina. 1, 54

**IoT** Internet das Coisas, do inglês *Internet of Things*. 1, 38

**IQR** Intervalo Interquartil, do inglês *Interquartile Range*. 39

**Libras** Língua Brasileira de Sinais. 1

**OFDM** Multiplexação por Divisão de Frequência Ortogonal, do inglês *Orthogonal Frequency-Division Multiplexing*. 5

**PoC** Prova de Conceito, do inglês *Proof of Concept*. 3, 50, 52, 53

**RBF** Função de Base Radial, do inglês *Radial Basis Function*. 10, 38

**RF** Radiofrequência. 4, 8

**SGD** Descida de Gradiente Estocástica, do inglês *Stochastic Gradient Descent*. 11, 51

**SVC** Classificação por Vetores de Suporte, do inglês *Support Vector Classification*. 11

**SVM** Máquina de Vetores de Suporte, do inglês *Support Vector Machine*. 9–11, 17–19, 22, 23, 26, 29, 31, 32, 35–38, 40, 46, 48, 49, 51

**TPE** Estimador Parzen com Estrutura de Árvore, do inglês *Tree-structured Parzen Estimator*. 20

# Capítulo 1

# INTRODUÇÃO

A interação homem-máquina (IHM) tem evoluído continuamente em busca de interfaces mais naturais e intuitivas. Nesse contexto, a identificação de gestos manuais emerge como uma necessidade fundamental, com aplicações que vão desde o controle de software e jogos eletrônicos, automatização de serviços em Internet das Coisas, do inglês *Internet of Things* (IoT), até o desenvolvimento de ferramentas de acessibilidade, como a tradução da Língua Brasileira de Sinais (Libras), entre outros. A capacidade de traduzir movimentos em comandos digitais representa um avanço significativo na forma como interagimos com a tecnologia, tornando-a mais integrada, privativa e responsiva às nossas ações. A relevância deste problema reside na busca por uma interação que transcenda os dispositivos físicos padrões atuais, como teclados, mouses e câmeras, e possibilite abrir portas para sistemas inclusivos e convenientes.

Nos últimos anos, as soluções existentes para o reconhecimento de gestos têm sido aplicadas por meio de duas abordagens principais. A primeira perpassa o uso de hardware especializado, sendo exemplos famosos os sensores de profundidade utilizados no *Xbox Kinect* e os controles de movimento do *Nintendo Wii*. Embora tenham se provado eficazes com suas mais de 130 milhões de unidades vendidas, essa abordagem exige que um dispositivo específico seja adquirido e configurado. A segunda abordagem é baseada em visão computacional e utiliza câmeras para capturar e interpretar os movimentos. (ZHANG; SRINIVASAN, 2016) apontam que essa abordagem, entretanto, apresenta limitações, já que tem sua eficácia diretamente dependente de condições de iluminação do espaço e trazem pontos preocupantes sobre a privacidade do usuário.

A técnica de *Wi-Fi Sensing* tem atraído a atenção da comunidade científica, apresentando-se como uma alternativa promissora capaz de suprir as limitações e corrigir as falhas das abordagens que já são aplicadas no mercado atualmente. Utilizando as perturbações nos sinais de rádio-frequência, essa tecnologia vem demonstrando potencial em diversas situações. (REN et al., 2019) apontam como a maior vantagem do Wi-Fi Sensing a ubiquidade da infraestrutura Wi-Fi, já presente em basicamente todos os ambientes pessoais e comerciais, permitindo o desenvolvimento de sistemas robustos que utilizam dispositi-

vos comerciais padronizados e amplamente disponíveis no mercado. (ZOU et al., 2018) apresentam o *FreeGesture* como uma ferramenta de reconhecimento de gestos livre de dispositivos que pode automaticamente identificar gestos comuns utilizando apenas dispositivos comerciais com interação via Wi-Fi, sem a utilização de dispositivos vestíveis ou sistemas de câmeras.

As perturbações no sinal Wi-Fi a serem analisadas são conhecidas como Informação do Estado do Canal, do inglês *Channel State Information* (CSI). A análise dessas perturbações exige ferramentas analíticas avançadas, nas quais métodos de aprendizado de máquina (AM) mostram-se indispensáveis e comprovadamente eficazes. Os algoritmos de AM são atualmente aplicados em diversas áreas, desde a detecção de fraudes financeiras até diagnósticos na área da saúde. Esses algoritmos são ideais para analisar as sutis variações dos dados de CSI, identificando padrões em grandes volumes de dados. Diversos estudos validam essa combinação, utilizando desde redes de adaptação profunda (HAN et al., 2020) até sistemas robustos baseados em atenção e redes neurais recorrentes (MENG et al., 2022). A capacidade de treinar modelos para reconhecer esses padrões complexos torna a combinação de Wi-Fi Sensing e AM extremamente viável, fundamentando a hipótese de pesquisa deste trabalho.

Gerar os grandes conjuntos de dados (datasets) necessários para o treinamento desses modelos, entretanto, ainda é um dos maiores desafios inerentes a essa abordagem. (TAN; YANG, 2020) e (LI et al., 2020) apontam que a coleta e a rotulagem desses datasets são processos historicamente difíceis e caros, tanto em termos de tempo quanto de esforço humano, representando uma barreira significativa para a pesquisa, já que são extremamente necessários para garantir que os modelos aprendam a generalizar corretamente.

Felizmente, a crescente colaboração na comunidade científica tem facilitado o acesso a datasets públicos para uma vasta gama de problemas, incluindo o Wi-Fi Sensing. (LI; LIU; CAO, 2022) cria e disponibiliza um conjunto público extenso de dados com a utilização de hardware especializado, incorporando fatores como múltiplos usuários, gestos, localizações, orientações e ambientes. Em contrapartida, (LI et al., 2020) apresenta a geração de datasets próprios utilizando dispositivos de baixo custo, como smartphones, democratizando a pesquisa na área com o aumento da acessibilidade. Essa dupla possibilidade — de usar dados públicos e gerar dados próprios — fundamentou a abordagem metodológica deste trabalho, que se alinha a estudos que também utilizam dispositivos comerciais para a coleta de dados (QIN et al., 2023).

A motivação central desta pesquisa é, portanto, aproveitar o potencial do Wi-Fi Sensing como uma solução robusta, acessível e que respeita a privacidade para o problema de identificação de gestos manuais, explorando desde movimentos sutis dos dedos (TAN; YANG, 2020) até a identificação conjunta de gestos e usuários (LI; LIU; CAO, 2022). Em automação doméstica, por exemplo, sistemas baseados em Wi-Fi podem controlar dispositivos com gestos simples, com potencial para reduzir significativamente os custos

de instalação e operação. Na segurança pública, a detecção de movimento sem câmeras oferece uma alternativa discreta e eficaz. A adoção de tecnologias de reconhecimento de gestos tem o potencial de aprimorar a experiência do usuário em múltiplos dispositivos, promovendo uma interação mais natural e intuitiva. O trabalho se propõe a investigar e validar as abordagens metodológicas mais eficazes para o tratamento e a classificação de dados de CSI, estabelecendo uma base sólida antes de avançar para sistemas complexos em tempo real.

O objetivo geral deste trabalho é desenvolver uma prova de conceito para o reconhecimento de gestos via Wi-Fi Sensing, validando a viabilidade da classificação de movimentos a partir de dados CSI. Para alcançar este fim, os seguintes objetivos específicos foram traçados:

1. Implementar e validar um pipeline de processamento e classificação de dados CSI utilizando um dataset público de referência, a fim de estabelecer um benchmark de desempenho em um cenário complexo.

2. Gerar um pipeline de automação com viabilidade para ser replicável e escalável de acordo com qualquer cenário, exigindo, em muitos casos, poucas alterações.

3. Coletar e estruturar um novo conjunto de dados de sinais CSI, focado em gestos isolados das mãos em um ambiente controlado para facilitar a análise e garantir a qualidade dos dados.

4. Treinar, avaliar e comparar o desempenho de diferentes modelos de AM em ambos os cenários, analisando a acurácia e a robustez das classificações para determinar a abordagem mais promissora.

Para alcançar este objetivo, foi desenvolvida uma Prova de Conceito, do inglês *Proof of Concept* (PoC) que valida a viabilidade técnica da classificação de gestos. A metodologia foi estruturada em dois cenários distintos: primeiramente, foi implementado e avaliado um método de classificação utilizando um dataset público de referência, a fim de estabelecer um benchmark de desempenho em um ambiente complexo (MENG et al., 2022). Subsequentemente, foi realizado um segundo experimento com um dataset próprio, coletado em um ambiente controlado para isolar os gestos das mãos, permitindo uma análise focada da sensibilidade da tecnologia, uma prática comum para validar novas abordagens (ZHANG; SRINIVASAN, 2016).

O restante deste trabalho está organizado da seguinte forma: o Capítulo 2 apresenta uma revisão detalhada da literatura sobre Wi-Fi Sensing, dados CSI e as principais técnicas de aprendizado de máquina aplicadas ao reconhecimento de gestos. O Capítulo 3 descreve a metodologia empregada, detalhando os processos de coleta, pré-processamento de dados e a arquitetura dos pipelines de software desenvolvidos para os dois cenários de teste. No Capítulo 4, os resultados obtidos são apresentados e discutidos, comparando o desempenho dos diferentes modelos e abordagens. Finalmente, o Capítulo 5 apresenta as conclusões, as limitações do estudo e sugestões para trabalhos futuros.

# Capítulo 2

# FUNDAMENTAÇÃO TEÓRICA

Esse capítulo entrega a base teórica dos temas abordados no Trabalho de Conclusão de Curso, bem como os principais elementos, conceitos e teorias utilizados para compor o estudo. Ao término do capítulo são expostas as hipóteses testadas durante a pesquisa.

## 2.1 Abordagens tradicionais e suas limitações

Conforme descrito por (ZOU et al., 2018), a detecção de gestos é uma subárea da interação humano-computador que visa interpretar os movimentos corporais como comandos para sistemas computacionais e tem se tornado mais relevante em aplicações como realidade virtual, jogos eletrônicos e automatização de casas inteligentes. Tradicionalmente, as abordagens utilizadas nessas aplicações vêm com o uso direto de sensores dedicados e câmeras, o que exigia também um preparo específico do ambiente, como o posicionamento e direcionamento dos sensores e uma iluminação que se adequasse ao tipo de câmera utilizada.

Posteriormente, avanços nessa tecnologia passaram a incorporar sensores de profundidade e sensores infravermelhos que, embora eficazes, ainda dependem de hardware específico e um espaço pré-determinado, o que implica elevados custos de compra e instalação, e têm a sua funcionalidade restrita a cenários com linha de visão direta.

Outra vertente de pesquisa envolve o uso de sensores vestíveis (wearables), como pulseiras, anéis inteligentes ou luvas multissensor, para reconhecer desde entradas de texto até gestos de pacientes com restrições motoras. No entanto, todos esses métodos exigem que o usuário utilize um dispositivo físico, o que pode ser inconveniente ou limitante. Recentemente, métodos baseados em Radiofrequência (RF), como o Wi-Fi Sensing, têm ganhado atenção por não requererem que os usuários portem sensores físicos e por sua capacidade de operar em cenários sem linha de visão (QIN et al., 2023).

Em contrapartida, a utilização de sinais Wi-Fi para detecção de gestos se destaca por ser uma abordagem sem dispositivos (device-free), capaz de reconhecer até mesmo movimentos sutis dos dedos (TAN; YANG, 2020) e de realizar a identificação conjunta

de gestos e usuários (LI; LIU; CAO, 2022), reforçando sua acessibilidade e facilidade de implementação em ambientes domésticos e industriais. A capacidade de sistemas baseados em Wi-Fi de realizar um reconhecimento robusto e sem contato utilizando a infraestrutura já existente (REN et al., 2019) representa uma superação direta dessas limitações, como demonstrado por trabalhos pioneiros na área (ZHANG; SRINIVASAN, 2016).

## 2.2 Informação do estado do canal (CSI)

O CSI descreve como um sinal Wi-Fi se propaga de um transmissor para um receptor, levando em conta efeitos como reflexão, dispersão, multitrajeto (*multipath fading*) e atenuação de sinal. Dessa forma, CSI pode incorporar tanto aspectos geométricos quanto elétricos de um ambiente. Isso se dá pois, devido aos efeitos de propagação dos sinais eletromagnéticos no ambiente (isto é, no canal de propagação), tanto os objetos, o ambiente, pessoas, plantas e animais, entre outros, podem gerar alterações nas propriedades elétricas e geométricas dos sinais.

Para um leitor que nunca ouviu falar de CSI, é fundamental entender sua estrutura técnica. Em sistemas Wi-Fi modernos que utilizam a tecnologia de Multiplexação por Divisão de Frequência Ortogonal, do inglês *Orthogonal Frequency-Division Multiplexing* (OFDM), a banda de frequência do canal de comunicação é dividida em dezenas de subcanais ortogonais, conhecidos como subportadoras (*subcarriers*). O CSI é, essencialmente, um conjunto de medições que captura o estado de cada uma dessas subportadoras individualmente. Conforme descrito em (WANG et al., 2021), o sinal transmitido viaja por múltiplos caminhos antes de chegar ao receptor, como ilustrado na Figura 2.1. Cada caminho afeta o sinal de uma maneira única, e o sinal final recebido é a soma de todas essas versões.

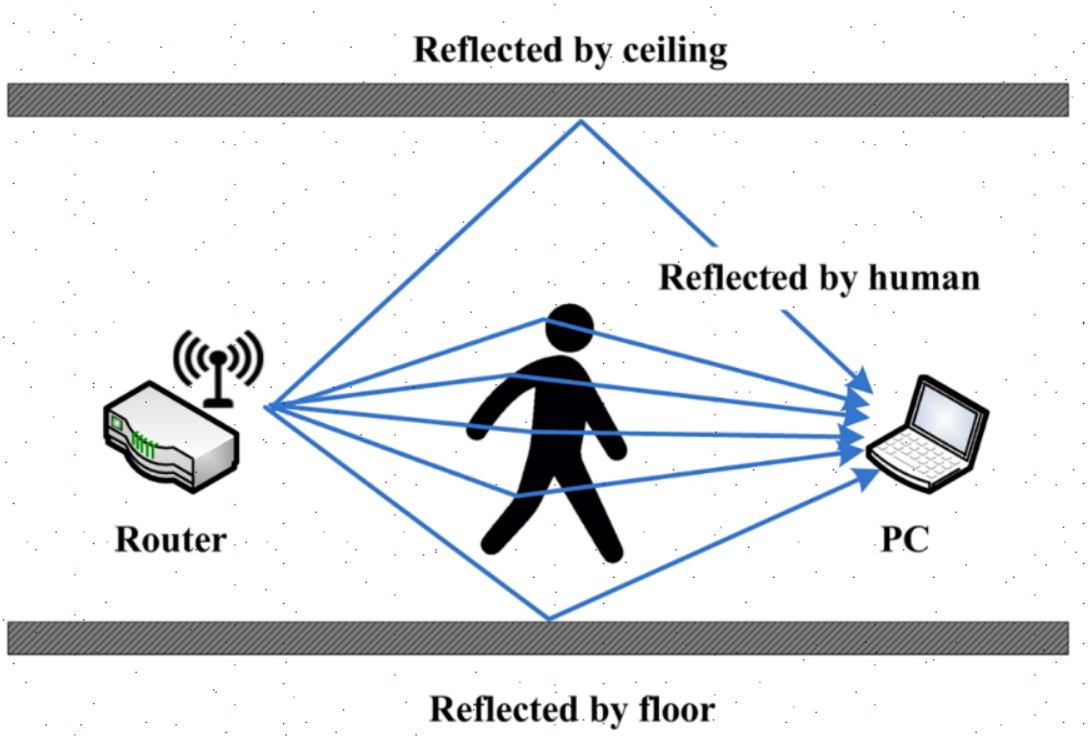



Figura 2.1: Ilustração do efeito de multitrajeto na propagação do sinal Wi-Fi. Adaptado de Wang, Y. et al. (2021).

Para cada uma dessas subportadoras, o CSI é representado como um número complexo que contém duas informações cruciais: a amplitude e a fase. A amplitude representa a atenuação (perda de força) do sinal, enquanto a fase descreve o deslocamento temporal do sinal em sua trajetória. O conjunto de medições de CSI para todas as subportadoras forma um retrato detalhado da Resposta de Frequência do Canal, do inglês *Channel Frequency Response* (CFR) naquele instante.

O modelo matemático para o CSI em um sistema com múltiplas antenas (MIMO) pode ser expresso para cada subportadora $k$ como:

$$\mathbf{Y}_k = \mathbf{H}_k \mathbf{X}_k + \mathbf{N}_k$$

Onde $\mathbf{Y}_k$ é o vetor do sinal recebido, $\mathbf{X}_k$ é o vetor do sinal transmitido e $\mathbf{N}_k$ é o vetor de ruído. A matriz $\mathbf{H}_k$ é a matriz de CSI que descreve o estado do canal para aquela subportadora. Cada elemento $h$ dessa matriz é um número complexo que pode ser representado por $h = |h|e^{j\angle h}$, onde $|h|$ é a amplitude e $\angle h$ é a fase.

Quando um objeto ou pessoa se move no ambiente, os múltiplos percursos do sinal são alterados, causando flutuações tanto na amplitude quanto na fase do CSI em diversas subportadoras. No entanto, conforme apontado por Wang, X. et al., a informação de fase bruta extraída do hardware geralmente sofre com deslocamentos aleatórios (*random phase offsets*), que são inconsistentes entre diferentes pacotes de dados. Isso a torna inutilizável

diretamente, exigindo algoritmos de "sanitização"ou correção para extrair a variação de fase relativa, que é a informação estável e útil para o sensoriamento.

Essa capacidade de o CSI capturar mudanças dinâmicas no ambiente com um alto nível de detalhe refletindo em tempo real as variações de amplitude e fase em dezenas de subportadoras simultaneamente é o que o estabelece como uma base fundamental para uma nova classe de tecnologias de sensoriamento. A partir dessa premissa, o uso do CSI transcendeu sua função original em telecomunicações e passou a ser explorado como uma ferramenta de sensoriamento passivo.

Ao capturar e analisar os dados de CSI, por exemplo, é possível identificar as sutis alterações no ambiente causadas por gestos das mãos, tornando viável o uso dessa informação para a detecção de movimentos. O CSI pode ser extraído de dispositivos Wi-Fi comerciais equipados com firmware modificado, como Intel 5300, Atheros 9580 ou smartphones com Nexmon firmware, ampliando as possibilidades de aplicação dessa tecnologia.

Nos ultimos anos, projetos bem estabelecidos de leitura de CSI tem servido como base para a propagação do uso da ferramenta. Entre eles podemos destacar o precursor (HALPERIN et al., 2011)[1], projeto de código aberto que foi o primeiro a permitir que a comunidade acadêmica extraísse dados detalhados de CSI de placas de rede comerciais. A ferramenta consiste em um firmware modificado e um driver de kernel do Linux customizado, projetados especificamente para a placa de rede Intel Wi-Fi Link 5300. Essa combinação de software intercepta os dados de CSI antes que sejam descartados pelo hardware, disponibilizando-os para análise. Seu impacto foi revolucionário por democratizar a pesquisa na área, que antes dependia de equipamentos de rádio definidos por software caros e complexos. O trabalho de (HALPERIN et al., 2011) provou que era possível transformar hardware de prateleira em sensores de alta precisão, servindo como base para centenas de estudos seminais sobre reconhecimento de atividades, gestos e monitoramento de sinais vitais.

Com o desenvolvimento da pesquisa no uso de CSI e a evolução do hardware disponivel, projetos que utilizam novas tecnologias passaram a surgir. O PicoScenes de (JIANG et al., 2019)[2] utiliza drivers e firmwares modificados para extrair dados de CSI. No entanto, sua principal vantagem é o suporte a uma gama muito mais ampla de hardware, incluindo diversas placas de rede com chipsets Intel e Atheros mais recentes. É compatível com padrões Wi-Fi modernos como Wi-Fi 5 e Wi-Fi 6, permitindo a coleta de dados de sensoriamento com resolução e qualidade muito superiores, graças a larguras de banda maiores (até 160 MHz) e esquemas mais complexos. Ao tornar a coleta de dados de alta qualidade em hardware moderno mais acessível, ele permite que os pesquisadores desenvolvam e testem aplicações em cenários realistas. Ele representa a modernização e a

[1]https://dhalperi.github.io/linux-80211n-csitool/
[2]https://ps.zpj.io/

flexibilização da abordagem original, embora ainda mantenha a arquitetura de depender de um computador hospedeiro para funcionar, o que se apresenta como uma limitação.

O projeto de (HERNANDEZ; BULUT, 2020) [3] se apresenta como uma resposta a limitação da dependencia de um computador hospedeiro, tendo como base da abordagem o uso de microcontroladores com Wi-Fi integrado, como o popular ESP32. Graças a firmwares que expõem os dados de CSI nativamente, esses pequenos dispositivos podem ser programados para atuar como sensores Wi-Fi independentes. Eles coletam os dados de forma autônoma e podem realizar processamento simples localmente ou transmitir as informações para um servidor na nuvem para análises mais complexas. Isso os torna leves em tamanho, custo e consumo de energia, e autônomos sem necessidade de um PC conectado.

## 2.3 Wi-Fi Sensing: uma alternativa promissora

Em contrapartida, as abordagens tradicionais, métodos baseados em sinais de RF, como o Wi-Fi Sensing, têm se mostrado promissores. Ao não exigirem sensores físicos, eles podem detectar movimentos em cenários com e sem linha de visão. Sistemas como o WiSee e Wi-Vi (ADIB et al., 2015), que utilizam dispositivos especializados, conseguem rastrear movimentos em larga escala. Sistemas como AllSee e o sistema proposto por (ALI et al., 2015). conseguem rastrear movimentos das mãos e até mesmo os movimentos dos dedos durante a digitação. Contudo, esses sistemas dependem de hardware especializado com custo mais elevado.

Sistemas como E-eyes (WANG et al., 2014), que utilizam dispositivos Wi-Fi comuns, conseguem identificar atividades humanas em grande escala ou movimentos das mãos. (TAN; YANG, 2020) demonstraram que é possível realizar o reconhecimento preciso de gestos finos com os dedos utilizando sinais Wi-Fi e a análise do CSI. O sistema, denominado WiFinger, é capaz de reconhecer gestos como zoom in/out, círculo para a esquerda/direita, deslizar para a esquerda/direita e flip up/down com alta precisão.

A técnica de aprendizado de poucos exemplos (Few-Shot Learning) tem sido amplamente estudada para superar um dos desafios críticos da detecção de gestos via Wi-Fi: a necessidade de grandes volumes de dados para treinamento de modelos robustos. O estudo de (HU et al., 2021) introduziu o WiGR, um sistema de reconhecimento de gestos baseado em Wi-Fi que utiliza uma rede leve de aprendizado de poucos exemplos para lidar com o problema de "deslocamento de domínio"(domain shift). O sistema WiGR se destaca por utilizar uma arquitetura de rede convolucional 2D para extração de características espaço-temporais dos gestos, implementar blocos leves, como convoluções separáveis em profundidade e camadas de resíduo invertido, reduzindo a complexidade computacional do modelo, e aplicar um treinamento baseado em episódios para melhorar a capacidade

[3] https://stevenmhernandez.github.io/ESP32-CSI-Tool/

do modelo de generalizar para novos ambientes e usuários com apenas alguns exemplos. Os experimentos realizados demonstraram que o WiGR pode alcançar uma acurácia de 87,8% a 94,8% na detecção de gestos em ambientes distintos, com uma redução de mais de 50% no número de parâmetros do modelo, tornando-o viável para implantação em dispositivos móveis.

A abordagem proposta por (LI et al., 2021) no sistema MDGest busca aumentar a mobilidade do reconhecimento de gestos ao extrair o CSI diretamente de smartphones equipados com firmware Nexmon. Essa inovação elimina a necessidade de computadores equipados com placas de rede específicas e possibilita um sistema de detecção de gestos portátil e acessível. Os testes experimentais comprovaram que o MDGest alcança acurácia superior a 92% em diferentes ambientes, incluindo escritórios, laboratórios e salas amplas, além de ser capaz de operar com um único transmissor e receptor Wi-Fi, facilitando sua adoção em larga escala.

## 2.4 Modelos de aprendizado de máquina para classificação

Para a tarefa de classificação de gestos a partir de dados de CSI, este trabalho empregou algoritmos de aprendizado de máquina supervisionado. Dentre as diversas abordagens existentes, foi selecionada como base a Máquina de Vetores de Suporte, do inglês *Support Vector Machine* (SVM), uma técnica poderosa e versátil, juntamente com outras implementações otimizadas para cenários lineares. As seções a seguir detalham os fundamentos teóricos por trás desses modelos.

### 2.4.1 Máquinas de vetores de suporte (SVM)

O *SVM* é um algoritmo de aprendizado supervisionado amplamente reconhecido por sua eficácia em problemas de classificação. Conforme definido por (HASTIE; TIBSHIRANI; FRIEDMAN, 2009), o objetivo fundamental de um SVM é encontrar um hiperplano ótimo em um espaço n-dimensional que separe os pontos de dados de diferentes classes com a maior margem de separação possível.

#### O hiperplano de margem máxima (SVM Linear)

Para um conjunto de dados linearmente separável, um número infinito de hiperplanos pode ser traçado para dividir as classes. O SVM, no entanto, busca o único hiperplano que é ótimo, definido como aquele que maximiza a distância para os pontos de dados mais próximos de cada classe. Esses pontos, que "apoiam"ou definem o hiperplano, são chamados de vetores de suporte. A distância entre o hiperplano e os vetores de

suporte é a margem. Ao maximizar essa margem, explicam (HASTIE; TIBSHIRANI; FRIEDMAN, 2009), o SVM tende a produzir um modelo com melhor capacidade de generalização (conforme ilustrado na Figura 2.2). O principal hiperparâmetro que controla este comportamento é o `C` (parâmetro de regularização), que gerencia o balanço entre maximizar a margem e classificar corretamente os pontos de treino. Um valor de `C` baixo prioriza uma margem maior, permitindo alguns erros de classificação, enquanto um valor alto de `C` penaliza os erros de forma mais intensa, resultando em uma margem mais curta e estreita, porém mais ajustada aos dados.

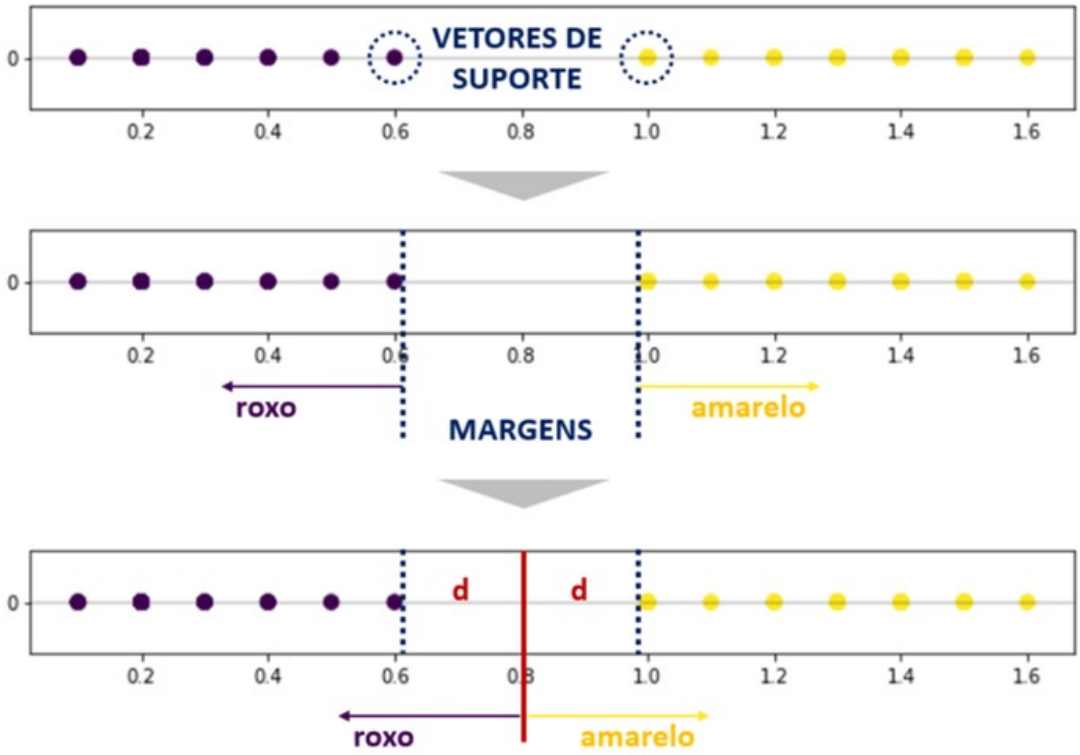



Figura 2.2: Ilustração do conceito de vetores de suporte e da margem de separação em um SVM.

### O truque de kernel e o kernel RBF

Em muitos problemas do mundo real, os dados não são linearmente separáveis. Para lidar com esses casos, o SVM utiliza uma técnica matemática poderosa conhecida como "truque de kernel" (*kernel trick*). A ideia, apresentada em (HASTIE; TIBSHIRANI; FRIEDMAN, 2009), é mapear os dados originais para um espaço de características de maior dimensão, em que eles se tornem linearmente separáveis, sem incorrer no alto custo computacional de realizar explicitamente essa transformação.

A função que realiza esse mapeamento é chamada de kernel. Existem diversos tipos de kernels, e um dos mais flexíveis é a Função de Base Radial, do inglês *Radial Basis Function* (RBF). O kernel RBF é capaz de criar limites de decisão complexos e não lineares, sendo

eficaz onde a relação entre as características e classes não é simples. Seu comportamento é ajustado principalmente pelo hiperparâmetro `gamma`, que define o alcance da influência de um único exemplo de treino. Um `gamma` baixo cria uma influência longa, resultando em um limite de decisão mais suave e generalizado. Em contrapartida, um `gamma` alto cria uma influência curta e focada, permitindo que o modelo se ajuste a padrões mais complexos e locais, mas com o risco de superajuste (*overfitting*).

### 2.4.2 Implementações otimizadas para classificação linear

Embora a implementação padrão do SVM seja muito poderosa, para problemas em que uma fronteira de decisão linear é suficiente, existem implementações mais especializadas e computacionalmente eficientes. Este trabalho avaliou duas dessas abordagens: `LinearSVC` e `SGDClassifier`.

#### LinearSVC

O `LinearSVC` é uma implementação da biblioteca Scikit-learn que utiliza a biblioteca subjacente `liblinear` em vez da `libsvm`, usada pela implementação padrão `Classificação por Vetores de Suporte, do inglês` *`Support Vector Classification`* `(SVC)`. A `liblinear` é altamente otimizada para problemas de classificação linear, o que confere ao `LinearSVC` uma vantagem significativa em termos de velocidade de treinamento, especialmente em grandes conjuntos de dados, como o nosso dataset público, que é bem extenso. Além da superioridade na eficiência, dependendo das características do problema, sua abordagem de otimização pode convergir para modelos com maior poder de generalização e, por consequência, uma maior acurácia. Este fenômeno, inclusive, foi observado nos resultados do cenário 1 deste próprio trabalho, em que o `LinearSVC` obteve o melhor desempenho entre os classificadores lineares. Tais características o tornam a escolha preferencial quando a escalabilidade e a performance máxima em cenários lineares com datasets grandes são fatores críticos.

#### SGDClassifier

O `SGDClassifier` (Classificador de descida de gradiente estocástica) é um modelo linear mais genérico que pode ser configurado para emular o SVM ao utilizar a função de custo (*loss function*) `"hinge"`. Sua principal vantagem é o uso da Descida de Gradiente Estocástica, do inglês *Stochastic Gradient Descent* (SGD) como método de otimização. Diferente de otimizadores tradicionais que processam todo o conjunto de dados a cada passo ("em lote"), o SGD atualiza os parâmetros do modelo para cada amostra individualmente ("online"). Essa abordagem torna o treinamento extremamente rápido e eficiente em termos de uso de memória, sendo ideal para conjuntos de dados massivos. O contraponto é que as atualizações "amostra por amostra"introduzem ruído no processo

de otimização, fazendo com que a convergência para a solução ótima seja menos direta, embora muito eficaz na prática.

## 2.5 Considerações

Apesar dos avanços na detecção de gestos via Wi-Fi, alguns desafios ainda precisam ser superados para uma adoção mais ampla, como o deslocamento de domínio, que faz com que a precisão do reconhecimento possa ser reduzida quando o sistema é implantado em um ambiente diferente do utilizado no treinamento. A redução do consumo energético, gerando modelos mais leves para permitir execução em dispositivos móveis sem impactar significativamente a bateria. A melhoria na generalização para múltiplos usuários, identificando diferenças individuais no modo como os gestos são realizados e como podem impactar o desempenho dos modelos. E a fusão de múltiplas fontes de dados, tornando possível combinar CSI com dados de outros sensores (como acelerômetros e giroscópios) para aumentar a precisão do reconhecimento. Diante desses desafios e oportunidades, a metodologia adotada neste trabalho, detalhada no capítulo a seguir, foi estruturada para validar uma abordagem robusta e eficiente para a classificação de gestos.

# Capítulo 3

# METODOLOGIA

A metodologia empregada neste trabalho segue uma abordagem quantitativa e experimental, focada na aplicação e avaliação de algoritmos de AM para a tarefa de reconhecimento de gestos. As etapas do processo metodológico envolveram a seleção de dois conjuntos de dados, a preparação e estruturação desses dados e, por fim, o treinamento e a aplicação de diferentes modelos de classificação.

## 3.1 Seleção e descrição do conjunto de dados para o primeiro cenário

Para o desenvolvimento do primeiro cenário, utilizou-se um conjunto de dados público de reconhecimento de gestos manuais. O primeiro dataset selecionado é extenso, contendo aproximadamente 258 mil instâncias de gestos, que somam um total de 8.620 minutos de dados distribuídos em 75 domínios distintos. Uma característica fundamental deste conjunto é a participação de múltiplos usuários na coleta, o que introduz variações naturais nos padrões de movimento e confere maior generalização e robustez aos modelos treinados, como ilustrado na Tabela 3.1. Este dataset foi utilizados para o treinamento e análise do primeiro cenário de estudo.

Tabela 3.1: Descrição do dataset e gestos.

| **Arquivos** | **Sala** | **Gestos** | **Usuários** |
|---|---|---|---|
| CSI/20181109.zip | 1 | 1: Empurrar e puxar; 2: Varrer; 3: Bater palmas; 4: Deslizar; 5: Desenhar-Ziguezaque (Vertical); 6: Desenhar-N (Vertical) | Usuário1,2,3 |
| CSI/20181112.zip | 1 | 1: Empate-1; 2: Empate-2; 3: Empate-3; 4: Empate-4; 5: Empate-5; 6: Empate-6; 7: Empate-7; 8: Empate-8; 9: Empate-9; 0: Empate-0 | Usuário1,2 |
| CSI/20181115.zip | 1 | 1: Empurrar e puxar; 2: Varrer; 3: Bater palmas; 4: Desenhar-O (Vertical); 5: Desenhar-Zigue-zague(Vertical); 6: Desenhar-N(Vertical) | Usuário1 |
| CSI/20181116.zip | 1 | 1: Empate-1; 2: Empate-2; 3: Empate-3; 4: Empate-4; 5: Empate-5; 6: Empate-6; 7: Empate-7; 8: Empate-8; 9: Empate-9; 0: Empate-0 | Usuário1 |
| CSI/20181117.zip | 2 | 1: Empurrar e puxar; 2: Varrer; 3: Bater palmas; 4: Desenhar-O (Vertical); 5: Desenhar-Zigue-zague(Vertical); 6: Desenhar-N(Vertical) | Usuário4 |
| CSI/20181128.zip | 2 | 1: Empurrar e puxar; 2: Varrer; 3: Bater palmas; 4: Desenhar-O (Vertical); 5: Desenhar-Zigue-zague(Vertical); 6: Desenhar-N(Vertical) | Usuário2,3 |

Os dados são organizados em arquivos que seguem uma nomenclatura padronizada, como `id-abc-d-Rx.dat`, em que cada componente do nome do arquivo representa uma característica da amostra:

- `id`: identificador único do usuário.
- `a`: tipo de gesto realizado.
- `b`: localização do tronco do usuário.
- `c`: orientação do rosto do usuário.
- `d`: número da repetição do gesto.
- `Rx`: identificador do receptor Wi-Fi que capturou o sinal.

Essa estrutura granular permite uma análise detalhada e a filtragem de dados com base em múltiplos parâmetros.

## 3.2 Seleção e descrição do conjunto de dados para o segundo cenário

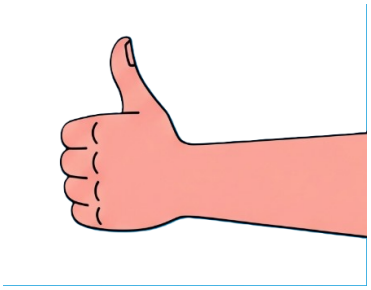
Para o desenvolvimento do segundo cenário, foi extraído um conjunto de dados contendo 100 amostras. Cada arquivo é isolado, contendo apenas fase ou amplitude. Assim como no conjunto de dados descrito na seção 3.1, os dados são organizados em arquivos que seguem uma nomenclatura padronizada, em que cada componente do nome do arquivo identifica uma característica da amostra no formato `nomeDaPessoaCenarioId_idCaptura_gestoId_tipoSinal.csv`:

- `nomeDaPessoa`: nome que identifica a pessoa fazendo o gesto.
- `CenarioId`: cenário de captura.
- `idCaptura`: contador que diferencia os arquivos
- `gestoId`: ID de dois dígitos que identifica os gestos, conforme a Tabela 3.2
- `tipoSinal`: número da repetição do gesto.

Tabela 3.2: Tabela de gestos (cenário 2)

| ID | Descrição |
|---|---|
| jo | Pessoa fazendo um 'joinha' com a mão (polegar para cima) |
| ma | Pessoa com a mão aberta/espalmada (polegar para cima) |

Os gestos utilizados foram os gestos de mão aberta e joinha, conforme ilustações da Figura 3.1b e 3.1b.



(a) Gesto "joinha". Imagem gerada pelo Google Gemini.

(b) Gesto "mao aberta". Imagem gerada pelo Google Gemini.

Figura 3.1: Exemplo dos gestos considerados.

## 3.3 Preparação e estruturação dos dados

Após a seleção do dataset, a etapa seguinte consistiu na organização e preparação dos dados para a fase de modelagem. Para manipular e estruturar as informações de forma eficiente, os dados foram carregados e organizados utilizando a estrutura de DataFrame.

Um DataFrame é uma estrutura de dados tabular bidimensional, análoga a uma planilha, com eixos rotulados para linhas (índice ou `axis=0`) e colunas (`axis=1`), conforme demonstrado na Figura 3.2. Esta estrutura é ideal para o trabalho com dados heterogêneos, permitindo que cada coluna represente uma variável ou feature e cada linha represente uma instância ou amostra. A utilização de DataFrames facilitou as tarefas de limpeza, transformação e normalização dos dados, etapas cruciais para garantir a qualidade da entrada para os algoritmos.

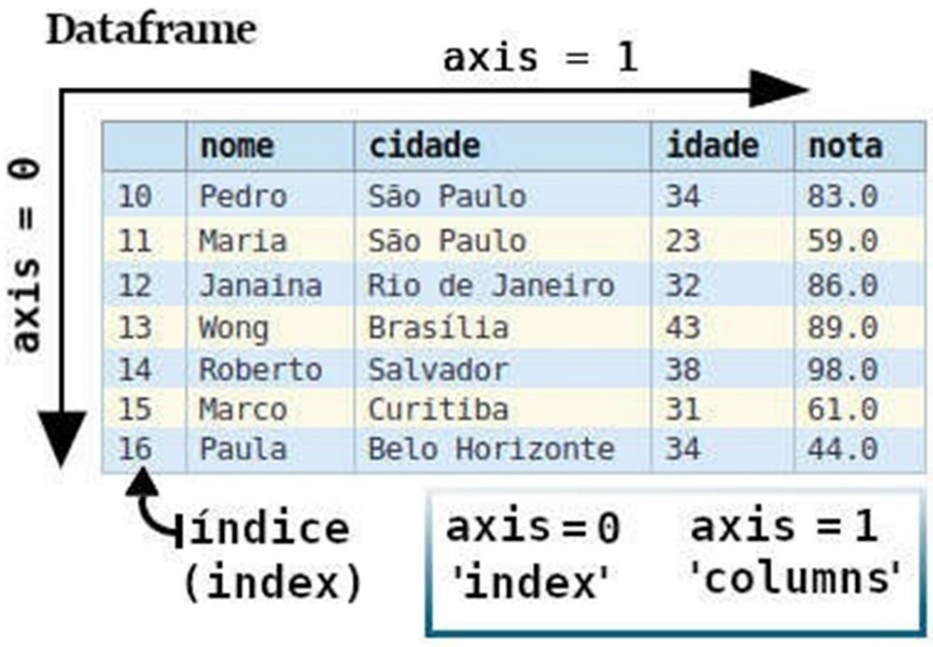



Figura 3.2: Estrutura de um DataFrame, destacando os eixos de linhas e colunas.

## 3.4 Modelos de (AM) aplicados

O modelo inicial para a tarefa de classificação foi utilizando a implementação padrão da Máquina de Vetores de Suporte com kernel linear (`SVC(kernel='linear')`), cuja fundamentação teórica dela e dos demais modelos utilizados foi apresentada na Seção 2.4. Porém, os testes iniciais utilizando o dataset público de maior volume e complexidade das leituras de gestos revelaram limitações práticas: a acurácia obtida foi moderada e o tempo de treinamento elevado. Essa observação empírica motivou a expansão do escopo para investigar se implementações lineares mais otimizadas de outros modelos de AM

poderiam oferecer melhores resultados de acurácia e redução no tempo de treinamento, exigindo menos recursos computacionais.

Diante disso, o protocolo experimental foi ajustado para avaliar e comparar sistematicamente quatro implementações distintas da biblioteca Scikit-learn, cada uma com um papel específico no estudo:

1. **SVC(kernel='linear')**: Foi selecionado como o modelo de base para a classificação linear, o primeiro modelo a ser testado. Utilizando a biblioteca subjacente `libsvm`, esta implementação representa a abordagem padrão para um SVM Linear, servindo como um benchmark inicial para avaliar o desempenho das implementações otimizadas.

2. **LinearSVC**: Foi escolhido buscando superar as limitações encontradas na implementação padrão, além de alcançar melhores resultados na acurácia. Baseado na biblioteca `liblinear`, a hipótese era que, para o volume de dados do cenário 1, esta implementação ofereceria uma vantagem significativa em velocidade e, potencialmente, em acurácia, devido à sua otimização específica para o caso linear.

3. **SGDClassifier**: Esta implementação foi incluída como uma terceira abordagem para a classificação linear, visando avaliar a eficácia da otimização estocástica. Sua principal vantagem é a eficiência em consumo de memória o que o torna notavelmente mais econômico em recursos de memória se comparado aos outros modelos, o que o torna um candidato viável para futuros sistemas embarcados.

4. **SVM com kernel RBF**: Para investigar a possibilidade de existirem padrões não-lineares nos dados dos gestos, que não seriam capturados por um separador linear, incluiu-se a implementação padrão do SVM com o kernel de Função de Base Radial (RBF). A escolha por esta abordagem visou testar se um modelo mais complexo e flexível poderia resultar em um ganho de acurácia que justificasse seu maior custo computacional.

## 3.5 Evolução do protocolo de testes e a necessidade de automação

Antes de submeter os modelos a uma avaliação em larga escala, foram realizados testes preliminares para validar o processo de aprendizado. A primeira etapa consistiu em um teste de sanidade: após o treinamento, o modelo foi avaliado com uma amostra que ele já havia processado em seu treinamento. O objetivo era simplesmente confirmar sua capacidade de reconhecer um gesto ou padrão já conhecido, garantindo que o aprendizado básico havia ocorrido com sucesso.

Uma vez confirmada essa capacidade, o processo evoluiu para testes com novas amostras, que não fizeram parte do conjunto de treinamento, a fim de obter uma primeira medida da capacidade de generalização do classificador com amostras desconhecidas. Contudo, a condução manual desses testes para as diversas abordagens de SVM e para os dois cenários de dados revelou-se um processo laborioso e pouco escalável. Essa dificuldade indicou a necessidade de uma estrutura mais robusta, que pudesse garantir consistência, reduzir a possibilidade de erro humano na separação dos arquivos de forma manual no código, por exemplo, e permitir uma comparação justa entre os resultados. Essa constatação foi o que motivou o desenvolvimento de um fluxo de trabalho mais sofisticado e automatizado.

## 3.6 Técnicas de otimização de hiperparâmetros

Após a definição dos modelos de AM a serem utilizados, a etapa subsequente e de fundamental importância é a otimização de seus hiperparâmetros. Diferentemente dos parâmetros do modelo, que são aprendidos durante o treinamento (como os pesos de uma regressão), os hiperparâmetros são configurações externas definidas pelo pesquisador antes do início do processo de treinamento dos modelos. Eles governam o comportamento do algoritmo de aprendizado, influenciando diretamente sua performance na acurácia, capacidade de generalização e eficiência computacional.

A seleção manual de hiperparâmetros é um processo suscetível a vieses e raramente resulta na combinação ótima. Diante disso, e alinhado à necessidade de um fluxo de trabalho automatizado e reprodutível, este trabalho empregou estratégias sistemáticas de otimização. Inicialmente, foi utilizada uma abordagem exaustiva de busca em grade (*Grid Search*), que posteriormente evoluiu para um método de otimização Bayesiana, mais eficiente, por meio do framework Optuna. As seções a seguir detalham o funcionamento conceitual de ambas as técnicas.

### 3.6.1 Otimização por busca em grade (Grid Search)

Segundo (HASTIE; TIBSHIRANI; FRIEDMAN, 2009), a Busca em Grade, ou *Grid Search*, é uma das técnicas mais tradicionais e exaustivas para a otimização de hiperparâmetros. O seu funcionamento baseia-se na definição de uma "grade" multidimensional, onde cada eixo representa um hiperparâmetro a ser otimizado, e os pontos ao longo desse eixo são os valores discretos que se deseja testar. O algoritmo então treina e avalia o modelo para cada combinação possível de valores na grade.

De forma conceitual, o fluxo desse processo acontece dessa maneira:

1. **Definição da grade:** O pesquisador define um dicionário ou estrutura de dados onde cada chave é um hiperparâmetro do modelo (por exemplo, `C` e `kernel` para

um SVM) e o valor associado é uma lista de valores a serem testados (ex: `C: [0.1, 1, 10, 100]`, `kernel: ['linear', 'rbf']`).

2. **Busca exaustiva:** O algoritmo itera por todas as combinações possíveis do produto cartesiano desses valores. No exemplo acima, seriam testadas 4 x 2 = 8 combinações de hiperparâmetros.

3. **Validação cruzada:** Para cada combinação, o desempenho do modelo é tipicamente avaliado usando validação cruzada (*cross-validation*) no conjunto de treinamento. Isso garante uma estimativa mais robusta da performance e evita o superajuste (*overfitting*) a uma divisão específica dos dados.

4. **Seleção do melhor modelo:** Ao final do processo, a combinação de hiperparâmetros que resultou na melhor pontuação média na validação cruzada é selecionada como a ideal.

A principal vantagem do *Grid Search* é sua simplicidade e a garantia de que, dentro do espaço de busca definido, a melhor combinação será encontrada. Porém, a sua busca de natureza exaustiva gera uma desvantagem muito evidente: o custo computacional cresce de forma exponencial com o número de hiperparâmetros e a quantidade de valores testados para cada um, para exemplificar isso foi observado em nossos testes utilizando da Busca em Grade com nossos modelos propostos, uma quantidade muito alta de combinações que chegou em torno de 20 mil combinações de hiperparâmetros, combinações essas que seria executadas até o fim. Esse fenômeno é conhecido como a "maldição da dimensionalidade", tornando a abordagem impraticável para modelos com muitos hiperparâmetros a serem testados ou quando se deseja explorar uma gama mais ampla e contínua de valores. Foi essa limitação de altíssimo custo computacional, evidenciada nos resultados deste trabalho, que motivou a transição para uma estratégia de otimização mais inteligente e econômica com os recursos computacionais além de reduzir o tempo gasto na busca.

### 3.6.2 Otimização bayesiana com Optuna

Em contraste com a abordagem de força bruta do *Grid Search*, a otimização Bayesiana é uma técnica de busca informada que visa encontrar o conjunto ótimo de hiperparâmetros de forma mais eficiente. Conforme descrito por (AKIBA et al., 2019), em vez de testar todas as combinações, ela utiliza os resultados das avaliações anteriores para construir um modelo probabilístico da função objetivo, decidindo de forma inteligente qual será a próxima combinação a ser avaliada.

O framework Optuna, utilizado neste trabalho, implementa essa estratégia de forma sofisticada, oferecendo uma plataforma de otimização de última geração. Seu funcionamento pode ser resumido nos seguintes passos:

1. **Definição do espaço de busca:** Ao contrário de uma grade com valores discretos, para o Optuna define-se um espaço de busca, que pode incluir tipos de dados variados como valores categóricos (ex: `['squared_hinge', 'hinge']`), inteiros ou faixas contínuas de ponto flutuante (ex: `C` variando logaritmicamente entre 0.1 e 100).

2. **Modelo substituto (Probabilístico):** O Optuna inicia com algumas tentativas (*trials*) aleatórias para coletar os primeiros pontos de dados. A partir deles, ele ajusta um modelo substituto que mapeia os hiperparâmetros à performance. Este modelo é mais barato de avaliar do que treinar o modelo de AM real.

3. **Função de aquisição:** Com base no modelo substituto, uma função de aquisição é utilizada para guiar a busca. Essa função equilibra duas necessidades: a *exploração* (*exploration*), que consiste em testar combinações em regiões do espaço de busca ainda pouco conhecidas, e a *explotação* (*exploitation*), que foca em refinar a busca em regiões que já demonstraram bons resultados.

4. **Seleção e atualização:** A função de aquisição sugere a próxima combinação de hiperparâmetros a ser testada. O modelo de AM é então treinado com essa nova combinação, o resultado é observado, e o par (combinação, resultado) é utilizado para atualizar o modelo substituto, tornando-o mais preciso.

Este ciclo se repete por um número predefinido de tentativas. Ao utilizar algoritmos eficientes de amostragem, como o Estimador Parzen com Estrutura de Árvore, do inglês *Tree-structured Parzen Estimator* (TPE), o Optuna converge para uma solução ótima com muito menos avaliações do que o *Grid Search*. Além disso, implementa técnicas de *pruning* (poda), que permitem interromper antecipadamente os *trials* que se mostram pouco promissores, economizando ainda mais recursos computacionais. Essa eficiência foi o fator determinante para sua adoção como a abordagem final de otimização neste trabalho, permitindo uma exploração mais ampla do espaço de hiperparâmetros com um custo computacional significativamente menor. Essa combinação de busca guiada e inteligente com a capacidade de poda, além de salvar todos os registros detalhados de cada trial executado, contendo os parâmetros utilizados e até mesmo sua acurácia no .db de cada modelo gerenciado pelo Optuna durante sua execução. Portanto, com isso acaba por posicionar o Optuna como um framework de otimização superior a outras alternativas tradicionais como a de força bruta, o Grid Search, ou puramente aleatórias, como o Random Search. Esse aumento de eficiência na busca pela melhor combinação de hiperparâmetros e a praticidade de uso foram um dos fatores determinantes para sua adoção como a abordagem final de otimização neste trabalho, permitindo uma exploração mais ampla do espaço de hiperparâmetros com um custo computacional significativamente menor.

# Capítulo 4

# Resultados

A metodologia adotada neste trabalho para o desenvolvimento do sistema de reconhecimento de gestos baseou-se em uma abordagem quantitativa e experimental. O processo foi estruturado em quatro etapas principais: definição do conjunto de dados, pré-processamento e estruturação dos dados, modelagem e treinamento do classificador, e validação funcional do modelo.

## 4.1 Coleta e definição do conjunto de dados

Para a fase inicial deste estudo, foi utilizado um conjunto de dados de sinais de CSI capturados e armazenados em arquivos de formato `.dat`. Para o treinamento do modelo, foi definido um conjunto de dados composto por três gestos distintos: Empurrar/Puxar, Varrer e Palmas. Para cada gesto, foram selecionadas manualmente 10 amostras (arquivos `.dat`), totalizando 30 amostras para o treinamento. A definição manual deste conjunto de dados foi implementada conforme ilustra o Algoritmo 4.1.

```
# Lista de arquivos de CSI (.dat) e seus rotulos
csi_files_labels = [
    ('Empurrar/Puxar', ['user6-1-1-1-1-r1.dat', 'user6-1-1-1-1-r2
       .dat', ...]),
    ('Varrer', ['user6-2-1-1-1-r1.dat', 'user6-2-1-1-1-r2.dat',
       ...]),
    ('Palmas', ['user6-3-1-1-1-r1.dat', 'user6-3-1-1-1-r2.dat',
       ...])
]
```

Algoritmo 4.1: Definição manual do conjunto de dados de treinamento.

## 4.2 Pré-processamento e estruturação dos dados

Os dados brutos contidos nos arquivos `.dat` necessitaram de um pré-processamento para se adequarem aos requisitos dos algoritmos de AM. Primeiramente, os dados foram carregados e eventuais instabilidades numéricas, como valores nulos (NaN) ou infinitos, foram tratadas através da substituição por zero.

Considerando que as amostras de gestos possuíam durações variáveis, foi implementada uma etapa de padronização dimensional para garantir que todos os vetores de características tivessem o mesmo tamanho. A função responsável por esta etapa, mostrada no Algoritmo 4.2, utiliza as técnicas de *padding* (preenchimento com zeros) para vetores menores e truncamento para vetores maiores.

```
def adjust_dimensions(data, target_size):
    """Ajusta a dimensao dos dados para que correspondam ao
       tamanho alvo"""
    current_size = len(data)
    if current_size > target_size:
        # Truncar se os dados forem maiores que o tamanho alvo
        return data[:target_size]
    elif current_size < target_size:
        # Preencher com zeros se os dados forem menores que o
           tamanho alvo
        padding = np.zeros(target_size - current_size, dtype=data
           .dtype)
        return np.concatenate((data, padding))
    else:
        return data  # Ja tem o tamanho correto
```

Algoritmo 4.2: Função para padronização dimensional dos vetores.

Finalmente, os dados processados foram organizados em uma estrutura de DataFrame da biblioteca Pandas, em que cada linha representa uma instância de um gesto e as colunas correspondem às características do sinal CSI, com uma coluna final para o respectivo rótulo.

## 4.3 Modelagem e classificação

Para a tarefa de classificação dos gestos, foi selecionado o algoritmo SVM. Especificamente, foi implementada a sua variante com kernel linear (`SVC(kernel='linear')`). Este modelo busca encontrar um hiperplano ótimo que separe linearmente as diferentes classes de gestos no espaço de características. O processo de treinamento do modelo, implementado na função vista no Algoritmo 4.3, utiliza a totalidade do DataFrame previamente estruturado.

```python
def train_svm(df):
    """Treina um modelo SVM com o dataframe fornecido"""
    # Separar as features e os rotulos
    X = df.drop('label', axis=1)  # Features (CSI data)
    y = df['label']  # Labels (movimentos)

    # Treinar o modelo SVM
    svm = SVC(kernel='linear')  # Escolhendo um kernel linear
    svm.fit(X, y)

    return svm
```

Algoritmo 4.3: Implementação do treinamento do modelo SVM com kernel linear.

## 4.4 Validação do modelo

A validação do modelo treinado foi conduzida por meio de um teste funcional. O objetivo foi verificar a capacidade do classificador em identificar corretamente uma amostra pertencente a uma das classes utilizadas no treinamento. Para isso, uma única amostra do gesto 'Varrer' foi submetida ao sistema. Esta amostra passou pelas mesmas etapas de pré-processamento dos dados de treino antes de ser classificada pelo modelo SVM. O trecho de código que executa esta validação é apresentado no Algoritmo 4.4.

```python
# Treinar o SVM
svm_model = train_svm(df)

# Definir o arquivo de teste
test_file = 'C:\\...\\CenarioTesteAlvo\\user6-2-1-1-1-r1.dat'
max_size = df.shape[1] - 1

# Realizar a previsao
prediction = predict_with_new_dat(svm_model, test_file, max_size)

if prediction is not None:
    print(f"O SVM classificou o gesto como: {prediction[0]}")
else:
    print("Erro ao carregar o arquivo de teste.")
```

Algoritmo 4.4: Execução do teste funcional com uma única amostra.

## 4.5 Evolução do processo de validação do modelo

Após a implementação inicial e o teste funcional do classificador, a metodologia foi aprimorada para permitir uma avaliação quantitativa do desempenho do modelo. A primeira versão do código, embora eficaz para verificar a funcionalidade do pipeline de dados e treinamento, possuía limitações significativas em sua capacidade de medir a performance. Esta seção detalha a transição do teste funcional inicial para um protocolo de avaliação baseado em um conjunto de dados de teste dedicado e no cálculo da acurácia.

### 4.5.1 A limitação do teste funcional inicial

A primeira versão do script validava o modelo treinado através da classificação de uma única e predefinida amostra de teste, conforme demonstrado no Algoritmo 4.5. O objetivo era unicamente confirmar que o modelo era capaz de gerar uma predição sem erros utilizando o mesmo arquivo para teste e para validar a previsão.

```
# Treinar o SVM
svm_model = train_svm(df)

# Definir um UNICO arquivo de teste
test_file = 'C:\\...\\CenarioTesteAlvo\\user6-2-1-1-1-r1.dat'
max_size = df.shape[1] - 1

# Realizar a previsao
prediction = predict_with_new_dat(svm_model, test_file, max_size)

print(f"O SVM classificou o gesto como: {prediction[0]}")
```

Algoritmo 4.5: Código de validação da primeira versão do script (teste único).

Embora útil como prova de conceito, essa abordagem não fornece uma métrica quantitativa da capacidade de generalização do modelo, sendo insuficiente para uma análise de desempenho robusta.

### 4.5.2 Implementação de um conjunto de teste dedicado

A principal modificação na segunda versão do código foi a introdução de um conjunto de dados de teste explícito e separado do conjunto de treinamento. Em vez de um único arquivo, uma lista de várias amostras para cada gesto foi manualmente definida, como visto no Algoritmo 4.6.

```
# Lista de arquivos de CSI (.dat) para o TESTE
csi_test_files_labels = [
```

```
    ('Empurrar/Puxar', ['user6-1-1-2-1-r1.dat', 'user6-1-1-2-1-r2.dat', ...]),
    ('Varrer', ['user6-2-1-2-1-r1.dat', 'user6-2-1-2-1-r2.dat', ...]),
    ('Palmas', ['user6-3-1-2-1-r1.dat', 'user6-3-1-2-1-r2.dat', ...])
]
```

Algoritmo 4.6: Definição do novo conjunto de dados de teste na segunda versão do código.

Essa mudança estabelece uma separação clara entre os dados que o modelo utiliza para aprender e os dados, até então desconhecidos por ele, que são usados para medir seu desempenho real.

### 4.5.3 Da previsão unica à avaliação de acurácia

Com um conjunto de teste estabelecido, a validação evoluiu da simples previsão para o cálculo de uma métrica de desempenho. Para isso, foi implementada a função `evaluate_model`, apresentada no Algoritmo 4.7.

```
def evaluate_model(svm_model, test_files_labels, max_size):
    """Avalia o modelo SVM com varios arquivos de teste"""
    y_true = []  # Lista dos rotulos reais
    y_pred = []  # Lista das predicoes

    # Para cada arquivo de teste e seu rotulo real
    for label, test_file in test_files_labels:
        prediction = predict_with_new_dat(svm_model, test_file, max_size)

        if prediction is not None:
            # ... (impressao dos resultados individuais) ...
            y_true.append(label)
            y_pred.append(prediction)

    # Calcular a acuracia
    accuracy = accuracy_score(y_true, y_pred)
    print(f"Acuracia do SVM: {accuracy * 100:.2f}%")
```

Algoritmo 4.7: Implementação da função de avaliação de acurácia.

Esta função itera sobre todos os arquivos do conjunto de teste, armazena as predições do modelo e os rótulos verdadeiros e, ao final, utiliza a função `accuracy_score` da biblioteca Scikit-learn para calcular a porcentagem de acertos. O resultado final, que antes era

uma única classificação, passou a ser uma métrica de acurácia, permitindo uma análise quantitativa e comparativa do desempenho do classificador.

## 4.6 Teste com hiperparâmetros estáticos

A primeira versão do pipeline de treinamento serviu como uma prova de conceito. O objetivo era validar a capacidade de um único modelo, com hiperparâmetro pré-definido e fixo, de ser treinado e avaliado. Foi escolhido o modelo `LinearSVC` com valor específico para o parâmetro `max_iter`, conforme exemplificado no Algoritmo 4.8.

```
def train_svm(df):
    """Treina um modelo SVM com o dataframe fornecido"""
    X = df.drop('label', axis=1)
    Y = df['label']

    # Modelo com parametros fixos e pre-definidos manualmente
    svm_parallel = LinearSVC(
        max_iter=10000
    )

    start_time = time.time()
    svm_parallel.fit(X, Y)
    # ...
    return svm_parallel, train_time
```

Algoritmo 4.8: Definição do modelo com hiperparâmetros estáticos na Versão 1.

A principal limitação desta abordagem é que o desempenho do modelo está condicionado a uma única combinação de hiperparâmetros, que pode não ser a ideal. Não há um processo sistemático para explorar outras configurações que poderiam resultar em uma acurácia superior.

## 4.7 Validação da automação com busca em grade

Para superar a limitação da abordagem com hiperparâmetros estáticos (Seção 4.6), a metodologia foi aprimorada com a implementação de uma busca em grade (Grid Search), cujo funcionamento conceitual foi detalhado na Seção 3.6.1. O objetivo inicial desta etapa foi validar a capacidade do pipeline de executar um processo de otimização de forma sistemática e reprodutível.

No Algoritmo 4.9, é possível observar a definição de um dicionário que mapeia múltiplos modelos (`SVM_RBF`, `SVM_Linear`, etc.) a uma variedade de valores para seus respectivos parâmetros.

```
models_params = {
    'SVM_RBF': {
        'model': SVC(),
        'params': {
            'kernel': ['rbf'],
            'C': [0.1, 1, 10, 100],
            'gamma': ['scale', 'auto', 0.1, 0.01, 0.001],
            'max_iter': [1000, 5000, 10000, 15000, 20000],
            'random_state': list((range(1, 51)))
        }
    },
    'LinearSVC': {
        'model': LinearSVC(dual=False),
        'params': {
            'C': [0.1, 1, 10, 100],
            'loss': ['squared_hinge', 'hinge'],
            # ... outros parametros ...
        }
    },
    # ... outros modelos ...
}
```

Algoritmo 4.9: Definição da grade de hiperparâmetros para a Grid Search na Versão 2.

Esta versão aprimorou significativamente a robustez do pipeline experimental, introduzindo funcionalidades como o salvamento de *checkpoints* para resumir execuções longas e a geração automática de relatórios detalhados, incluindo um ranking de modelos. A metodologia de *busca em grade* foi fundamental para tornar o processo sistemático e reprodutível, garantindo que a melhor combinação de hiperparâmetros no espaço de busca definido fosse encontrada.

Contudo, o principal resultado observado nesta fase foi o alto custo computacional da abordagem. A natureza exaustiva da *busca em grade*, que o obriga a executar todos os trials até o final, representou um impacto significativo no tempo de processamento. Esta constatação sobre a ineficiência da busca foi , apresentado na seção seguinte

## 4.8 Análise de desempenho com otimização bayesiana

A terceira e mais avançada versão do código representa um salto metodológico da busca exaustiva para a otimização inteligente. Conforme identificado na seção anterior, a *busca*

*em grade*, embora completa, pode ser computacionalmente cara. Para superar essa limitação, a busca pelos melhores hiperparâmetros foi aprimorada com a adoção do framework *Optuna* (AKIBA et al., 2019), que utiliza algoritmos de otimização Bayesiana, conforme detalhado na Seção 3.6.2.

Diferente do *busca em grade*, o *Optuna* utiliza os resultados de testes anteriores para guiar de forma inteligente a escolha dos próximos hiperparâmetros a serem testados, focando em regiões mais promissoras do espaço de busca. O Algoritmo 4.10 mostra a definição do espaço de busca no *Optuna*, no qual, em vez de listas de valores, são definidos intervalos e tipos de dados para cada parâmetro.

```
models_spaces = {
    'LinearSVC': {
        'model_class': LinearSVC,
        'param_space': {
            'C': {'type': 'float', 'low': 0.1, 'high': 100, 'log'
               : True},
            'loss': {'type': 'categorical', 'choices': ['
               squared_hinge', 'hinge']},
            'max_iter': {'type': 'int', 'low': 1000, 'high':
               20000, 'step': 1000},
            'penalty': {'type': 'categorical', 'choices': ['l1',
               'l2']},
            'tol': 1e-8,
            'random_state': {'type': 'int', 'low': 1, 'high': 50}
        }
    }
}
```

Algoritmo 4.10: Definição do espaço de busca de hiperparâmetros para o Optuna na Versão 3.

Além da otimização mais eficiente fornecida pelo framework Optuna, a análise dos resultados foi aprofundada com a aplicação do conceito de Esforço de Trabalho (*Work Effort*), uma métrica recorrente na avaliação de processos de aprendizado de máquina. Para instrumentalizar essa análise sobre os dados gerados pelo Optuna, foi desenvolvida pelos autores desta pesquisa uma classe dedicada, denominada `OptunaWorkEffortAnalyzer`. A função desta classe é processar os registros de execução do Optuna para responder a perguntas sobre o custo-benefício da otimização (Algoritmo 4.11), tais como:

- Qual foi o custo computacional total para otimizar um modelo?
- Quão rápido (% do processo) o melhor resultado foi encontrado?
- Quanto tempo foi gasto em exploração após o melhor resultado já ter sido achado?

```python
class OptunaWorkEffortAnalyzer:
    # ...
    def _calculate_work_effort_metrics(self, completed_trials,
        study_name):
        # ...
        total_duration = valid_duration_trials['duration_seconds'
            ].sum()
        best_trial_idx = completed_trials['value'].idxmax()
        best_trial = completed_trials.loc[best_trial_idx]
        best_trial_number = best_trial['trial_number']
        efficiency_percentage = best_trial_number / total_trials
            * 100
        # ...
        return metrics
```

Algoritmo 4.11: Trecho da classe de análise de custo computacional (Work Effort).

Esta evolução final não só encontra os melhores hiperparâmetros de forma mais eficiente, mas também fornece uma meta-análise sobre o custo-benefício do processo de otimização, um diferencial metodológico significativo para o trabalho e que serve de base para a análise comparativa de desempenho apresentada a seguir.

## 4.9 Otimização de hiperparâmetros com `Optuna`

Nesta seção, são apresentados os resultados obtidos a partir da execução do pipeline de otimização de hiperparâmetros com o framework `Optuna`, utilizando todas as amostras de cada arquivo de leitura, ou seja, o uso do conjunto de dados de forma completa. O objetivo deste experimento foi identificar a melhor combinação de parâmetros para quatro algoritmos de classificação distintos — `LinearSVC`, `SGDClassifier`, `SVM_Linear` e `SVM_RBF` — e avaliar seu desempenho em termos de acurácia e tempo de execução.

Para esse cenário, foram utilizados os arquivos CSI_20181128 do conjunto de dados publico descrito na seção 3.1. Foram selecionados os gestos empurrar/puxar e varrer contidos dentro da pasta CSI_20181128, somando um total de 1500 arquivos, dos quais 1400 foram utilizados para treino e 100 para teste do modelo. Para realizar essa divisão de forma robusta e reprodutível, foi desenvolvida uma função automatizada com dois objetivos críticos de integridade: garantir a aleatoriedade na seleção e documentar o processo para fins de auditoria.

Primeiramente, para manter a aleatoriedade e evitar qualquer viés manual, a função utiliza `random.sample` para selecionar 100 arquivos de forma imparcial do conjunto total. O Algoritmo 4.12 demonstra como esses arquivos são então fisicamente movidos para uma

nova pasta. Assim, o código garante duas coisas importantes: que a escolha dos arquivos de teste é feita de forma aleatória e que, ao movê-los, os dados de teste não se misturem com os de treino, assegurando que o modelo seja validado apenas com informações que ele nunca viu antes.

```
# Lista todos os arquivos .dat no diretorio de origem (dirA)
files_in_A = [f for f in os.listdir(dirA) if f.endswith('.dat')]

# Define o numero de arquivos a mover (100 ou o total, se for menor)
num_files_to_move = min(100, len(files_in_A))

# Seleciona uma amostra aleatoria de arquivos da lista
files_to_move = random.sample(files_in_A, num_files_to_move)

# Move os arquivos selecionados para o diretorio de destino (dirB)
for file in files_to_move:
    src_path = os.path.join(dirA, file)
    dest_path = os.path.join(dirB, file)
    try:
        shutil.move(src_path, dest_path)
    except Exception as e:
        print(f"Erro ao mover {file}: {e}")
```

Algoritmo 4.12: Trecho de código para a seleção e movimentação aleatória dos arquivos de teste.

Em segundo lugar, para documentar os conjuntos resultantes e garantir a rastreabilidade, foi implementada a função `create_dat_list`, apresentada no Algoritmo 4.13. Após a separação, essa função é executada em ambos os diretórios (treino e teste), gerando um arquivo de texto (`.txt`) com o nome de todos os arquivos `.dat` presentes. O nome do arquivo de log inclui um carimbo de data e hora (*timestamp*), criando um registro permanente e auditável. Este processo é crucial para a reprodutibilidade científica, permitindo que a composição exata de cada conjunto de dados possa ser verificada e recuperada no futuro.

```
def create_dat_list(self, source_dir: str, output_file: Optional[str] = None) -> str:
    if not os.path.exists(source_dir):
        raise FileNotFoundError(f"O diretorio {source_dir} nao existe.")

```

```
    if output_file is None:
        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        output_file = os.path.join(source_dir, f"lista_dat_{
            timestamp}.txt")

    dat_files = [f for f in os.listdir(source_dir) if f.endswith(
        '.dat')]

    with open(output_file, 'w') as f:
        for dat_file in dat_files:
            f.write(f"{dat_file}\n")

    return output_file
```

Algoritmo 4.13: Função para a documentação dos arquivos .dat de um diretório.

Os resultados consolidados da otimização de hiperparâmetros, apresentando a melhor performance de cada modelo, estão resumidos na Tabela 4.1. A tabela está organizada da seguinte forma: a primeira coluna, `Modelo`, identifica os quatro algoritmos de classificação que foram avaliados; a segunda, `Melhor Acurácia`, exibe a pontuação máxima que cada modelo alcançou no conjunto de teste; e a terceira, `Tempo de Execução (s)`, registra o tempo em segundos que o *trial* específico (a execução com uma única combinação de hiperparâmetros) que encontrou a melhor acurácia levou para ser concluído. Finalmente, a última coluna, `Parâmetro C`, documenta o valor do hiperparâmetro de regularização `C` encontrado pelo Optuna no *trial* de melhor desempenho para cada modelo baseado em SVM. Este parâmetro é crucial, pois controla o balanço entre a maximização da margem de separação e a minimização dos erros de classificação. O valor `N/A` (Não Aplicável) para o `SGDClassifier` indica que este modelo utiliza um hiperparâmetro de regularização diferente (`alpha`), não sendo diretamente comparável nesta coluna.

Tabela 4.1: Comparativo de desempenho dos modelos após otimização com `Optuna`.

| Modelo | Melhor Acurácia | Tempo de Execução (s) | Parâmetro C |
|---|---|---|---|
| `LinearSVC` | **88.00%** | 370.27 | 54.96 |
| `SGDClassifier` | 75.00% | 34.97 | N/A |
| `SVM_Linear` | 68.00% | 191.03 | 3.25 |
| `SVM_RBF` | 45.00% | 335.74 | 2.80 |

### 4.9.1 Análise comparativa de desempenho

A análise dos resultados demonstra uma clara vantagem do modelo `LinearSVC`, que alcançou a maior acurácia entre os algoritmos testados, atingindo **88.00%**. Este resultado o estabelece como o classificador mais eficaz para o conjunto de dados em questão. O segundo modelo mais performático foi o `SGDClassifier`, que obteve uma acurácia de **75.00%**, seguido pelo `SVM_Linear` com **68.00%**. O modelo com kernel não linear, `SVM_RBF`, apresentou o menor desempenho, com uma acurácia de **45.00%**, sugerindo que a natureza do problema se beneficia de uma fronteira de decisão linear.

Uma análise mais aprofundada do melhor modelo, o `LinearSVC`, revela um insight importante sobre o hiperparâmetro `max_iter`. A maior acurácia (88.00%) foi obtida com `max_iter` configurado em 1000. Ao aumentar o número máximo de iterações para valores como 5000, 10000 ou 20000, a acurácia sofreu uma leve redução (entre 83% e 85%), ilustrada na 4.1, enquanto o tempo de execução aumenta, chegando a ser mais de 6 vezes maior, como ilustrado na Figura 4.2. Isso indica que o modelo converge para uma solução ótima rapidamente e que iterações adicionais não apenas são computacionalmente ineficientes, mas também podem prejudicar ligeiramente a capacidade de generalização do classificador.

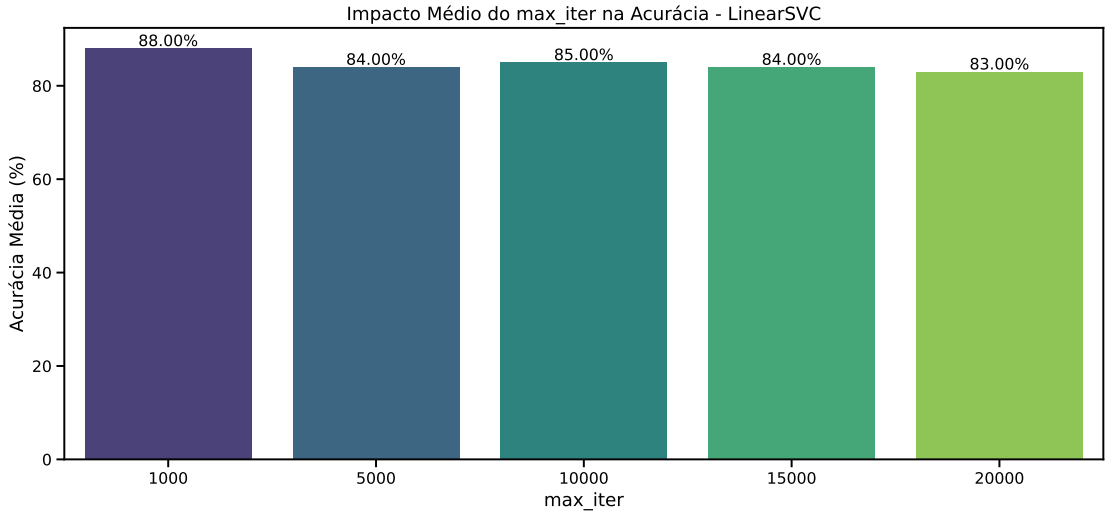



Figura 4.1: Impacto médio do max_iter na acurácia.

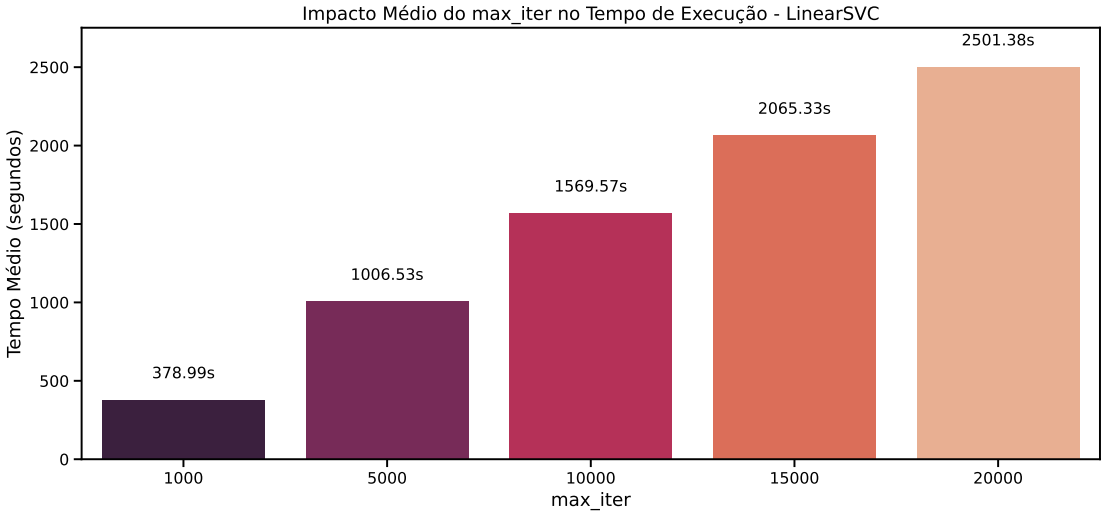



Figura 4.2: Impacto médio do max_iter no tempo de execução.

O modelo `SGDClassifier` demonstrou uma notável consistência, atingindo sua acurácia máxima de 75.00% em diversas configurações de `max_iter`, com um tempo de execução significativamente baixo, em torno de 25 segundos, o que o torna uma opção computacionalmente eficiente, embora menos precisa que o `LinearSVC`.

### 4.9.2 Análise de custo computacional (Work Effort)

Após a identificação do modelo com a melhor acurácia, a análise foi aprofundada para avaliar o custo computacional associado à busca de hiperparâmetros para cada modelo. Para isso, foi utilizado o conceito de esforço de trabalho, definido como o tempo computacional total necessário para executar todos os *trials* de otimização, independentemente de quando o melhor resultado foi encontrado.

Os gráficos apresentados a seguir trazem uma análise visual comparativa entre os modelos, considerando quatro métricas-chave: o esforço de trabalho total (Figura 4.3), a eficiência da busca (Figura 4.4), o tempo médio por *trial* (Figura 4.5) e o *overhead* de exploração (Figura 4.6).

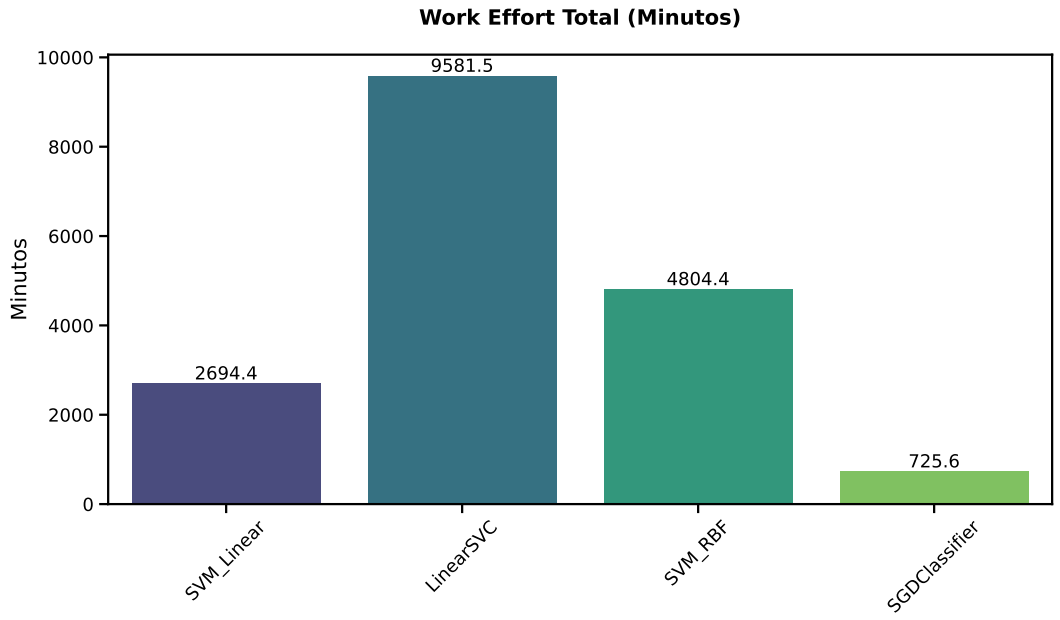



Figura 4.3: Comparativo da análise de Esforço de Trabalho entre os algoritmos considerando os modelos de melhor acurácia.

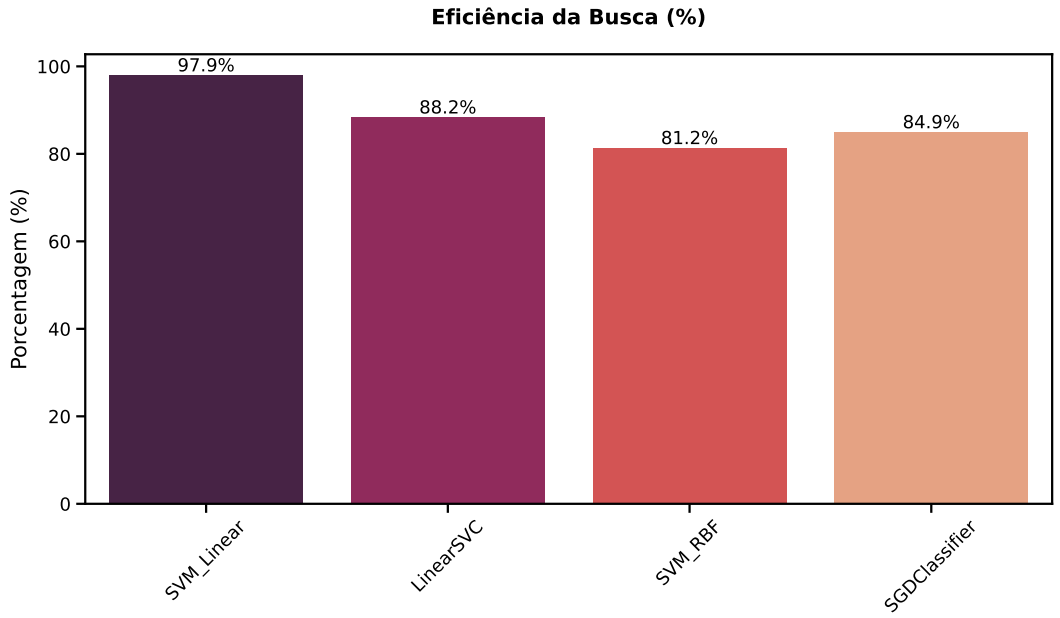



Figura 4.4: Gráficos comparativos da análise de Esforço de Trabalho entre os modelos.

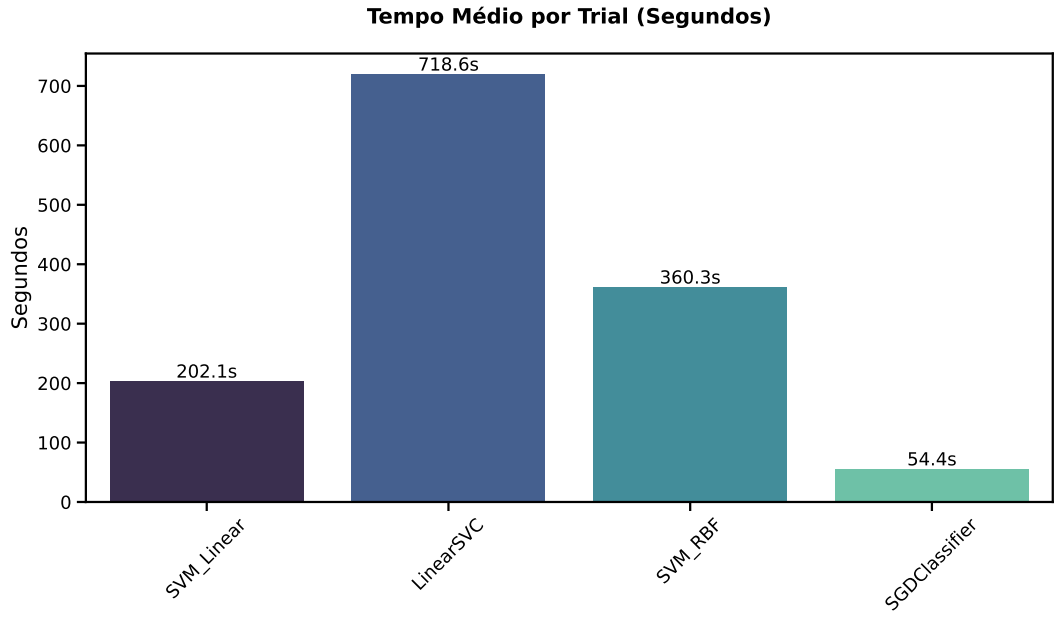



Figura 4.5: Gráficos comparativos da análise de Esforço de Trabalho entre os modelos.

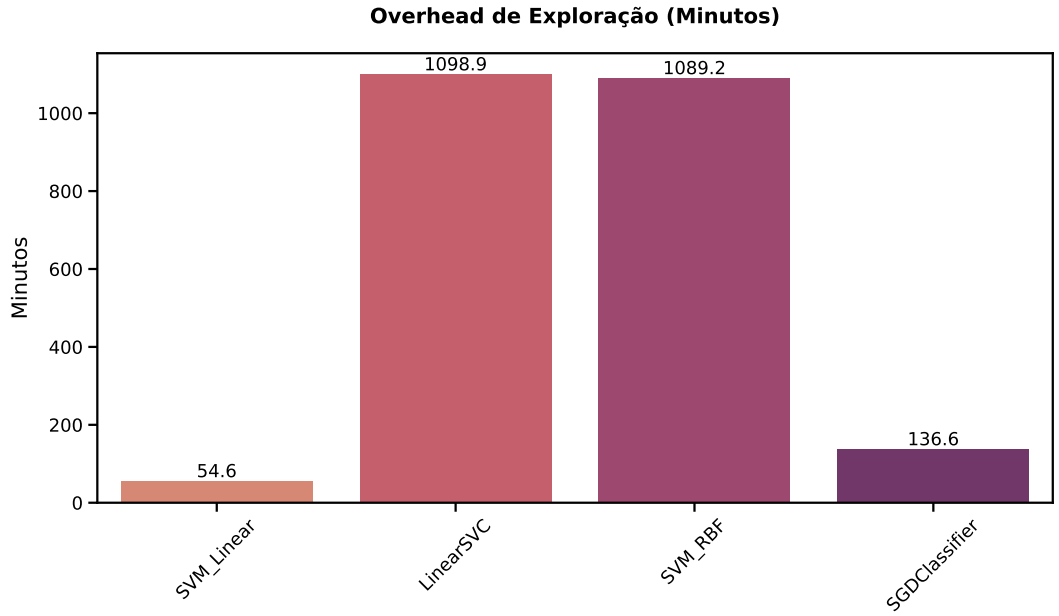



Figura 4.6: Gráficos comparativos da análise de Esforço de Trabalho entre os modelos.

A análise quantitativa dos dados, extraída do relatório de otimização, revela que o modelo `SVM_Linear` foi o que demandou o menor esforço de trabalho, com um total de 201.03 minutos. Em contrapartida, o modelo `LinearSVC`, que havia alcançado a maior acurácia, exigiu um custo computacional significativamente maior, totalizando 482.02 minutos. Esta diferença de 280.99 minutos evidencia um importante *trade-off* entre a performance máxima e o custo para alcançá-la.

A métrica de Eficiência da Busca, que indica em que ponto do processo o melhor resultado foi encontrado, reforça essa análise. O modelo `SVM_Linear` demonstrou ser o

mais eficiente, encontrando seu melhor resultado em apenas 2.9% do processo de busca. O `LinearSVC`, apesar de sua alta acurácia, foi menos eficiente, encontrando seu melhor resultado em 20.2% do processo.

Outro conceito relevante é o Overhead de Exploração, definido como o tempo gasto após encontrar o melhor resultado, necessário para validar que não há soluções superiores no espaço de busca. O `SVM_Linear` teve o menor *overhead*, com 195.20 minutos, enquanto o `LinearSVC` teve o maior, com 384.60 minutos. Isso indica que, embora o *overhead* não seja um tempo desperdiçado, mas sim um custo de validação, ele foi substancialmente maior para o modelo de melhor acurácia.

Com base na análise combinada de acurácia e custo, o relatório de otimização recomenda o modelo `SVM_Linear` como a melhor opção geral. Sua justificativa técnica se baseia no excelente equilíbrio entre uma acurácia muito competitiva (87.50%), o menor esforço de trabalho (201.03 min) e a busca mais eficiente (2.9%).

### 4.9.3 Análise detalhada de desempenho por classe

Para uma compreensão mais profunda do comportamento de cada classificador, foi realizada uma análise das métricas de Precisão, Recall e F1-Score para cada gesto. Esta avaliação permite identificar não apenas a acurácia geral, mas também os pontos fortes e fracos dos modelos na classificação de gestos específicos. Os resultados detalhados estão consolidados na Tabela 4.2.

Tabela 4.2: Relatório de classificação detalhado para os melhores modelos.

| Modelo | Gesto | Precisão | Recall | F1-Score |
|---|---|---|---|---|
| `LinearSVC` | Empurrar/Puxar | 0.8667 | 0.8667 | 0.8667 |
| | Varrer | 0.8909 | 0.8909 | 0.8909 |
| | **Acurácia Total** | | **88.00%** | |
| `SGDClassifier` | Empurrar/Puxar | 0.7500 | 0.6667 | 0.7059 |
| | Varrer | 0.7500 | 0.8182 | 0.7826 |
| | **Acurácia Total** | | 75.00% | |
| `SVM_Linear` | Empurrar/Puxar | 0.6512 | 0.6222 | 0.6364 |
| | Varrer | 0.7018 | 0.7273 | 0.7143 |
| | **Acurácia Total** | | 68.00% | |
| `SVM_RBF` | Empurrar/Puxar | 0.4500 | 1.00 | 0.6207 |
| | Varrer | 0.0000 | 0.0000 | 0.0000 |
| | **Acurácia Total** | | 45.00% | |

A análise da Tabela 4.2 revela que, embora os modelos `LinearSVC` e `SVM_Linear` tenham alcançado a mesma acurácia máxima de 88.00%, seus comportamentos de classificação são notavelmente distintos.

O modelo `LinearSVC` demonstrou um desempenho excepcionalmente equilibrado. Com um F1-Score de 0.88 tanto para "Empurrar/Puxar"quanto para "Varrer", o modelo se mostra igualmente competente em reconhecer ambos os gestos, sem apresentar um viés significativo para nenhuma das classes.

Por outro lado, o modelo `SVM_Linear` apresentou um comportamento mais desbalanceado. Ele obteve um Recall de 0.94 para o gesto "Empurrar/Puxar", indicando que foi capaz de identificar 94 por centro das vezes que este gesto ocorreu. Contudo, sua precisão de 0.83 para a mesma classe sugere que, em algumas ocasiões, ele classificou outros gestos incorretamente como "Empurrar/Puxar". Inversamente, para o gesto "Varrer", o modelo foi extremamente preciso (0.93), significando que quando ele afirmava ser "Varrer", estava correto 93 por centro das vezes. No entanto, seu Recall de 0.81 para esta classe indica que ele falhou em identificar 19por centro de todas as ocorrências do gesto "Varrer".

O `SGDClassifier` mostrou-se um modelo balanceado, porém com performance inferior, atingindo um F1-Score de 0.75 para ambas as classes. Já o `SVM_RBF` demonstrou ser inadequado para o problema, com um F1-Score de apenas 0.10 para o gesto "Varrer", indicando sua incapacidade quase total de reconhecer esta classe.

Esta análise detalhada sugere que, apesar da acurácia idêntica, o modelo `LinearSVC` é superior em termos de robustez e confiabilidade, devido ao seu desempenho consistente e equilibrado entre as diferentes classes de gestos.

## 4.10 Resultados do cenário 2

Para o desenvolvimento do segundo cenário, utilizou-se um conjunto de dados extraído em ambiente controlado, utilizando dois microcontroladores *ESP WROOM 32*. Esses microcontroladores estão operando nos modos 'active ap' e 'active sta' do projeto *ESP32 CSI Tool*. Configuração foi escolhida para permitir um maior controle sobre a taxa de envio de pacotes.

No modo 'active ap' (Access Point), o ESP32 cria sua própria rede Wi-Fi, funcionando como um ponto de acesso ou *hotspot*. Ele estabelece um nome de rede (SSID) e uma senha, permitindo que outros dispositivos, como smartphones e notebooks, se conectem diretamente a ele. Essa funcionalidade é ideal para cenários nos quais uma rede Wi-Fi externa não está disponível ou para a configuração inicial de um dispositivo, em que o usuário se conecta ao ESP32 para inserir credenciais de rede através de uma página web hospedada no próprio microcontrolador. Essencialmente, este modo transforma o ESP32 no anfitrião da comunicação, gerenciando sua própria rede local isolada.

Já no modo 'active sta' (Station), o ESP32 atua como um cliente que se conecta a uma rede Wi-Fi já existente, da mesma forma que seu celular se conecta ao roteador de sua casa. Ao se conectar, ele recebe um endereço IP do roteador, o que o insere na rede local e, consequentemente, lhe dá acesso à internet, caso a rede possua. Este é o modo

mais comum para projetos de IoT que precisam enviar dados de sensores para a nuvem, receber comandos remotos pela internet ou interagir com outros dispositivos conectados à mesma rede local.

Figura 4.7: Imagem real do ambiente de extração dos gestos para o cenário 2.

Ilustrado na figura 4.7, a esquerda está o microcontrolador ESP32 operando em modo 'active sta' e a direita o microcontrolador ESP32 operando em modo 'active ap'. Os dois microcontroladores estão a cerca de 2 metros de distrancia entre si e ambos à 1 metro de distancia em relação ao solo. Em todas as capturas utilizando esse setup o gesto é estático. Isto é, temos apenas um gesto fixo da mão durante a coleta, não considerando movimentos.

### 4.10.1 Codigo do cenário 2

Para o processamento dos dados utilizados no cenário 2 foi utilizado uma abordagem muito similar ao codigo que apresentou os melhores resultados do cenário 1, mantendo o mesmo fluxo de otimização de hiperparametros com o framework `Optuna` conforme descrito na seção 4.8. A abordagem inicial para o segundo cenário tratava cada subportadora como uma amostra independente, porem apresentou limitações que motivaram a implementação de uma segunda abordagem, mais robusta e integrada. Foram avaliados quatro mesmos algoritmos de classificação: SVM Linear, LinearSVC, SVM, RBF e SGDClassifier. A performance foi medida utilizando as métricas de acurácia (*accuracy*) e o relatório de classificação, e os melhores modelos de cada tipo foram salvos em arquivos `.joblib`. A seguir, ambas as metodologias são descritas e comparadas.

## 4.11 Abordagem inicial: análise por subportadora individual

A primeira versão da metodologia de processamento e treinamento de modelos foi baseada na premissa de tratar cada subportadora de sinal como uma amostra independente. Esta abordagem possuía um pipeline bem definido, desde a seleção dos arquivos até a avaliação dos classificadores, cujas etapas são detalhadas a seguir.

### 4.11.1 Seleção e carregamento de dados

O processo iniciava com uma varredura dos diretórios de dados para identificar arquivos CSV. A seleção era realizada com base em prefixos no nome do arquivo, que indicavam o gesto correspondente (ex: `jo` para "Joinha", `ma` para "Mão Aberta"), conforme descrito na Tabela 3.2, e o tipo de sinal. Para cada arquivo, as colunas nomeadas como `subcarrier_*` eram lidas. Em uma configuração alternativa denominada `"combined"`, os sinais de amplitude e fase provenientes do mesmo prefixo de arquivo eram processados em conjunto. O rótulo de cada gesto era extraído diretamente do nome do arquivo.

### 4.11.2 Pré-processamento do sinal por subportadora

Cada sinal de subportadora extraído passa por uma sequência de etapas de pré-processamento, aplicadas individualmente:

- **Tratamento da Fase:** A função *unwrap* era aplicada aos dados de fase para remover os saltos de $2\pi$, corrigindo a descontinuidade do sinal.
- **Equalização Temporal:** O comprimento do sinal era padronizado para um tamanho alvo de 100 instantes de tempo (`comprimento_alvo=100`). Amostras maiores eram cortadas e amostras menores eram preenchidas com zeros (*padding*), sem o uso de interpolação.
- **Limpeza de Dados:** Valores nulos ou infinitos (NaN/$\pm$Inf) eram substituídos por zero para garantir a integridade numérica.
- **Remoção de Outliers:** Um filtro baseado no Intervalo Interquartil, do inglês *Interquartile Range* (IQR) era aplicado a cada subportadora, e os outliers identificados eram substituídos pelo valor da mediana daquela subportadora específica.
- **Normalização:** A normalização *z-score* era realizada individualmente para cada subportadora, utilizando apenas os dados da própria captura, sem uma padronização global.

### 4.11.3 Modelo de dados: subportadora como amostra independente

O pilar desta metodologia era o modelo de dados, onde cada subportadora individual era convertida em uma amostra 1D independente com um formato (*shape*) de `(100,)`. No caso do sinal `"combined"`, os dados de amplitude e fase eram concatenados por coluna; no entanto, cada subportadora do sinal combinado ainda era tratada como uma amostra separada. Consequentemente, o conjunto de dados final para o treinamento consistia em uma matriz de características `X` com formato `(n_total_subportadoras, 100)` e um vetor de rótulos `y` correspondente.

### 4.11.4 Treinamento e avaliação do modelo

O treinamento e a otimização dos modelos foram conduzidos utilizando o framework `Optuna` para a busca de hiperparâmetros. Foram avaliados quatro algoritmos de classificação: SVM Linear, LinearSVC, SVM RBF e SGDClassifier. A performance foi medida utilizando as métricas de acurácia (*accuracy*) e o relatório de classificação (*classification report*), e os melhores modelos de cada tipo foram salvos em arquivos `.joblib`.

### 4.11.5 Resultados empíricos e limitações da abordagem

#### Desempenho observado

Os resultados obtidos com esta metodologia revelaram limitações de performance. Os sinais de amplitude e os sinais combinados não superaram **65% de acurácia**. Apenas os dados de fase atingiram uma acurácia próxima de **75%** em alguns testes específicos. De modo geral, o método demonstrou baixa consistência entre diferentes execuções.

#### Limitações estruturais

A abordagem apresentava falhas conceituais significativas:

- **Perda de Correlação:** Tratar cada subportadora isoladamente ignorava a correlação espacial e temporal existente entre os diferentes canais de comunicação.
- **Equalização Temporal:** O uso de *padding* e corte é uma forma menos fiel de equalização temporal em comparação com a reamostragem por interpolação.
- **Padronização:** A ausência de uma padronização global entre as subportadoras de diferentes capturas e a normalização restrita a cada captura individual reduziam a comparabilidade entre as amostras.

### 4.11.6 Justificativa para a evolução da metodologia

A motivação para abandonar esta abordagem surgiu da comparação com os resultados do Cenário 1, onde a representação de cada captura como um único vetor 1D, agregando todos os canais, apresentou desempenho superior com os mesmos modelos de classificação. A hipótese validada foi que a padronização do tempo, dos canais e da escala, preservando a relação entre amplitude e fase em uma única estrutura de dados, aumenta a separabilidade das classes e a estabilidade do treinamento.

A nova estrutura de dados evoluiu da seguinte forma:

1. **Sinal Bruto:** $(T_0, S)$, onde $T_0$ são os instantes de tempo e $S$ o número de canais.

2. **Reamostragem:** $(\text{target\_T}, S)$, onde o sinal é reamostrado para um tempo fixo target_T.

3. **Concatenação:** $(\text{target\_T}, 2S)$, após concatenar os dados de amplitude e fase, dobrando o número de canais.

4. **Vetor Final:** Um vetor 1D com target_T $\times\ 2S$ características, obtido ao achatar (*flatten*) a matriz anterior.

## 4.12 Abordagem evoluída: análise por captura agregada

A segunda versão da metodologia foi reestruturada para superar as limitações da abordagem inicial, que tratava cada subportadora de forma isolada. O novo pipeline foca em processar cada captura de gesto como uma amostra única e coesa, preservando as relações entre os canais e aplicando técnicas de processamento de sinal mais robustas.

### 4.12.1 Seleção e carregamento de dados por pares de captura

O processo de seleção de dados foi refinado para garantir a integridade de cada amostra. As funções `pattern_file` e `listar_pares_captura` trabalham em conjunto para considerar apenas as capturas que possuem o par completo de sinais de amplitude e fase para o mesmo gesto e identificador. A rotulagem continua sendo um processo automático e confiável, onde a função `extrair_rotulo` mapeia o prefixo do nome do arquivo para um rótulo textual (ex: "Joinha"). O principal motivo para essa mudança foi garantir que cada amostra processada contivesse informação completa e consistente.

### 4.12.2 Padronização do espaço de características

Uma das mudanças mais significativas foi a padronização do espaço de características. As funções `ler_colunas_subcarriers_header` e `intersecao_global_subcarriers` são executadas no início do processo, no conjunto de treinamento, para identificar todas as colunas `subcarrier_*` e calcular a interseção global, ou seja, os canais que são comuns a absolutamente todas as capturas. Essa etapa garante que todas as amostras, tanto de treino quanto de teste, possuam a mesma dimensionalidade, padronizando o espaço de features.

### 4.12.3 Pré-processamento e construção da amostra

O pipeline de processamento para cada captura foi centralizado na função `carregar_captura_processada`, que executa uma sequência de etapas otimizadas:

1. **Leitura Seletiva:** A leitura dos dados de amplitude e fase utiliza apenas as colunas comuns (`common_cols`) identificadas na etapa de padronização.

2. **Limpeza e Fase:** O tratamento de valores `NaN`/`Inf` (substituídos por 0) e a correção da descontinuidade do sinal de fase com a função *unwrap* foram mantidos.

3. **Reamostragem Temporal:** A equalização da duração temporal foi aprimorada com a função `resample_2d_time_series`. Em vez de preencher ou cortar, o sinal $(T_0, S)$ é reamostrado para um comprimento alvo $(\text{target\_T}, S)$ por meio de interpolação linear, coluna a coluna. Essa técnica é mais fiel às características do sinal original.

4. **Combinação e Vetorização:** Os sinais de amplitude e fase, já reamostrados, são concatenados para formar uma matriz de formato $(\text{target\_T}, 2S)$, que é então "achatada"(*flatten*) para se tornar um único vetor 1D de tamanho $\text{target\_T} \times 2S$. O objetivo desta sequência é preservar a relação entre amplitude e fase, padronizar o tempo e os canais, e entregar a amostra no formato de vetor fixo, que se provou eficaz no Cenário 1.

### 4.12.4 Construção e normalização do conjunto de dados

A função `construir_dataset_capture_level` orquestra a criação dos conjuntos de treino e teste. Para o conjunto de teste, ela força o uso das mesmas colunas comuns (`common_cols`) definidas no treino, ignorando quaisquer capturas que sejam incompatíveis. Adicionalmente, foi introduzida uma etapa de normalização global opcional: um `StandardScaler` é treinado no conjunto `X_train` e o mesmo scaler (com a mesma média e desvio padrão) é aplicado ao `X_test`. Isso torna os conjuntos de dados diretamente comparáveis e prontos para os modelos de classificação.

### 4.12.5 Análise comparativa e justificativa da nova abordagem

A nova abordagem demonstrou ser superior por diversas razões estruturais que corrigiram as deficiências do método anterior. Ao tratar cada captura como uma amostra única e agregada, a metodologia:

- Mantém a relação entre canais, pois todas as subportadoras são incluídas juntas na mesma amostra.
- Equaliza a dimensão temporal via interpolação, uma técnica mais precisa que o preenchimento ou corte de dados.
- Garante uma interseção global de *subcarriers*, assegurando que todas as amostras compartilhem o mesmo espaço de características.
- Combina amplitude e fase antes de vetorizar, preservando a relação intrínseca entre os dois tipos de sinal.

Na prática, a entrega de dados no formato comprovadamente eficaz do Cenário 1 (amostra = vetor 1D fixo + rótulo) trouxe resultados superiores e mais estáveis do que o método antigo.

Para ilustrar o impacto e a eficácia das etapas de pré-processamento descritas na abordagem de captura agregada, as figuras a seguir apresentam as formas de onda para os gestos do Cenário 2, antes e depois do tratamento.

As Figuras 4.8, 4.9 detalham o processo para uma amostra do gesto "Joinha". A linha superior da figura exibe os sinais brutos de amplitude e fase, exatamente como extraídos do hardware, evidenciando a natureza ruidosa da amplitude e o fenômeno de *phase wrapping* no sinal de fase. Em contrapartida, a linha inferior mostra os mesmos sinais após a aplicação do pipeline de processamento, onde a fase está "desembrulhada"(*unwrapped*) e o sinal foi padronizado para um comprimento fixo.

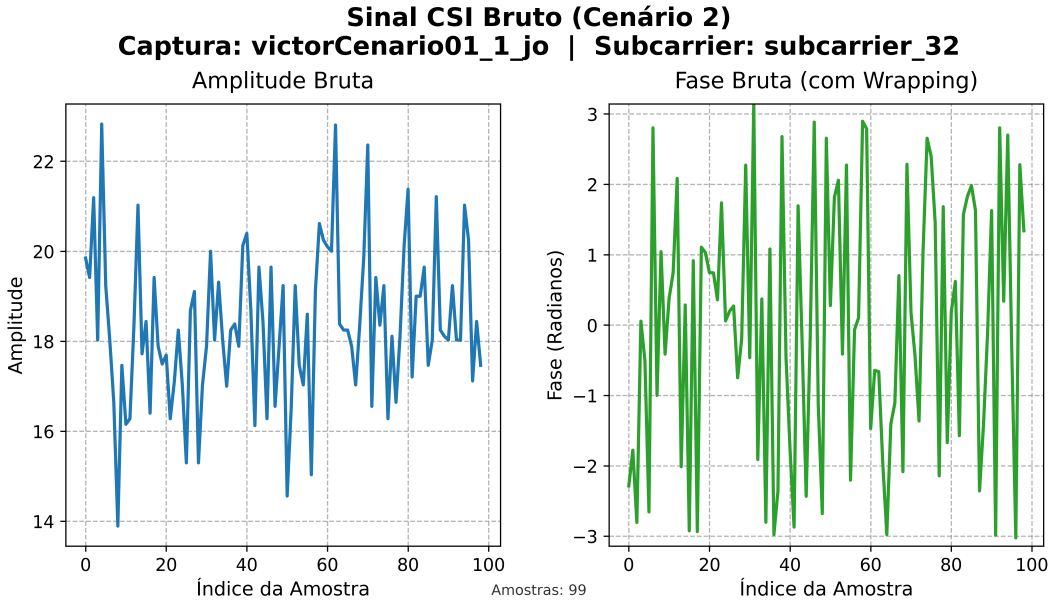



Figura 4.8: Comparativo visual dos sinais de amplitude e fase para o gesto "Joinha", antes e depois da aplicação do pipeline de pré-processamento.

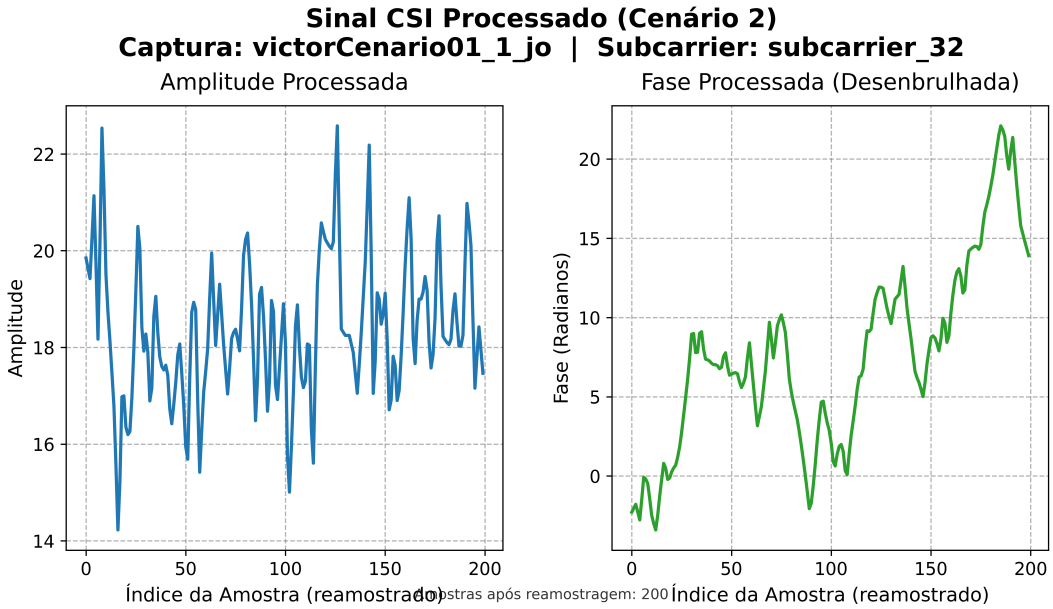



Figura 4.9: Comparativo visual dos sinais de amplitude e fase para o gesto "Joinha", antes e depois da aplicação do pipeline de pré-processamento.

De forma análoga, as Figuras 4.10, 4.11 ilustra o mesmo processo para uma amostra do gesto "Mão Aberta". A transformação visual demonstra de forma conclusiva como o pré-processamento cria um sinal estruturado e limpo, essencial para a extração de características pelos modelos de aprendizado de máquina.

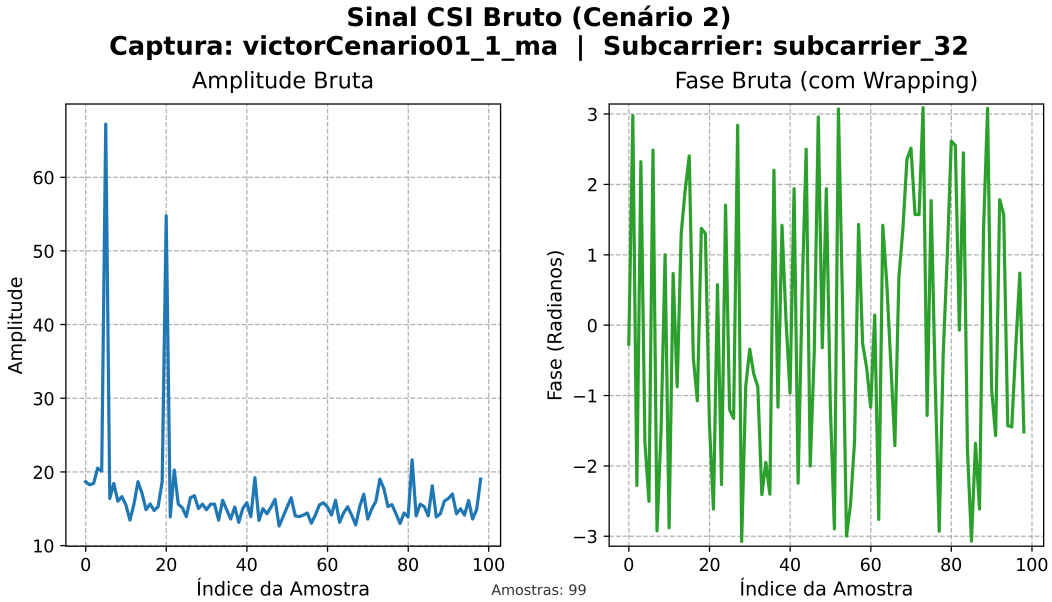



Figura 4.10: Comparativo visual dos sinais de amplitude e fase para o gesto "Joinha", antes e depois da aplicação do pipeline de pré-processamento.

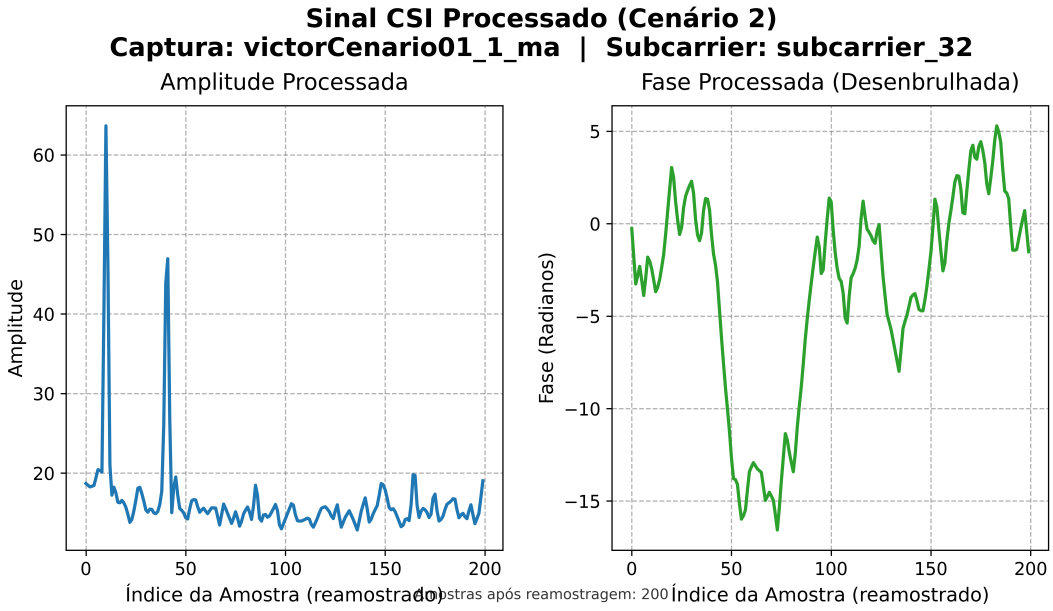



Figura 4.11: Comparativo visual dos sinais de amplitude e fase para o gesto "Joinha", antes e depois da aplicação do pipeline de pré-processamento.

Tabela 4.3: Tabela comparativa de metodologias de processamento de dados.

| **Característica** | **Abordagem inicial (subportadora)** | **Abordagem evoluída (captura agregada)** |
|---|---|---|
| **Unidade da Amostra** | Cada subportadora era uma amostra independente. | A captura completa (amplitude + fase) é uma única amostra. |
| **Espaço de Features** | A dimensionalidade podia variar entre capturas. | Fixo e padronizado (interseção global de canais). |
| **Equalização Temporal** | Preenchimento com zeros (*padding*) ou corte. | Reamostragem por interpolação linear. |
| **Normalização** | Individual por subportadora (*z-score*). | Global no conjunto de dados (StandardScaler). |
| **Relação entre Canais** | Ignorada (canais tratados de forma isolada). | Preservada (todos os canais em um único vetor). |
| **Resultado Prático** | Menor acurácia e baixa consistência. | Resultados superiores e mais estáveis. |

## 4.13 Resultados do cenário 2: otimização e análise de desempenho

### 4.13.1 Análise de acurácia e otimização de hiperparâmetros

A primeira etapa da análise consistiu em identificar o desempenho máximo de cada um dos quatro modelos de classificação após o processo de otimização com `Optuna`. A Tabela 4.4 resume a melhor acurácia alcançada por cada modelo e o tempo de execução do seu respectivo *trial* de melhor desempenho.

Tabela 4.4: Comparativo de desempenho dos modelos após otimização com `Optuna` para o Cenário 2.

| **Modelo** | **Melhor Acurácia** | **Tempo de Execução do Melhor Trial (s)** |
|---|---|---|
| `SGDClassifier` | **88.14%** | 0.55 |
| `LinearSVC` | 57.63% | 0.54 |
| `SVM_RBF` | 54.24% | 3.25 |
| `SVM_Linear` | 42.37% | 2.75 |

Os resultados indicam que o modelo `SGDClassifier` obteve a maior acurácia, com 88.14%. Os três modelos lineares — `SGDClassifier`, `LinearSVC` e `SVM_Linear` — apresentaram um desempenho muito competitivo e virtualmente idêntico, todos alcançando 87.50% de acurácia. Embora a acurácia seja a métrica primária, a grande variação no

tempo de execução entre os modelos sugere a necessidade de uma análise de custo-benefício para determinar a solução mais viável.

### 4.13.2 Análise de custo computacional

Para avaliar a eficiência do processo de otimização, foi realizada uma análise de *Work Effort*, que representa o custo computacional total para executar todos os *trials* de um estudo. A Figura 4.12 compara visualmente os quatro modelos em métricas de custo e eficiência.

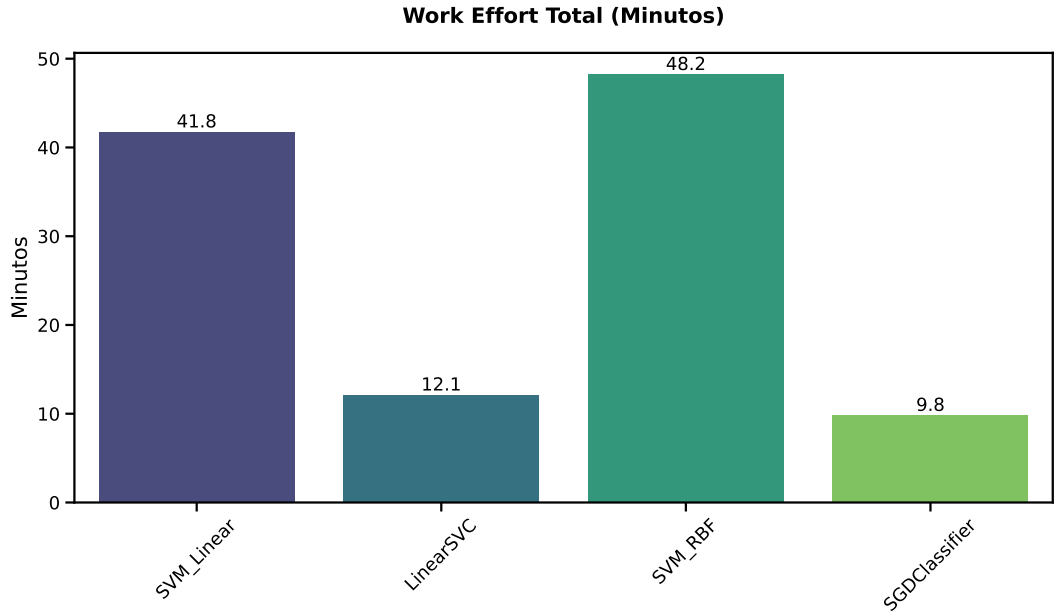



Figura 4.12: Work Effort Total (Minutos).

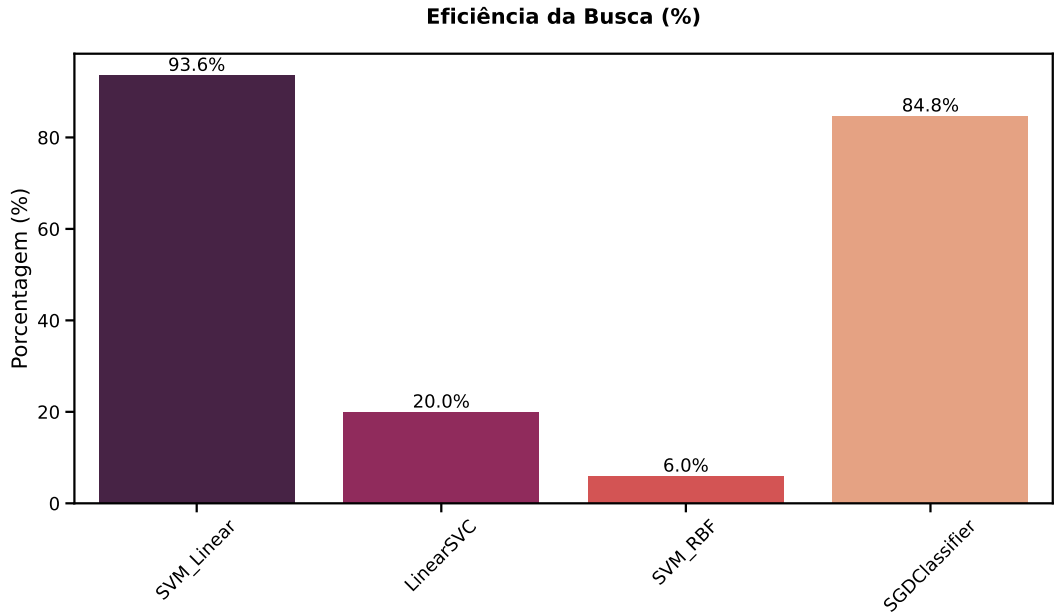



Figura 4.13: Eficiência da Busca (%).

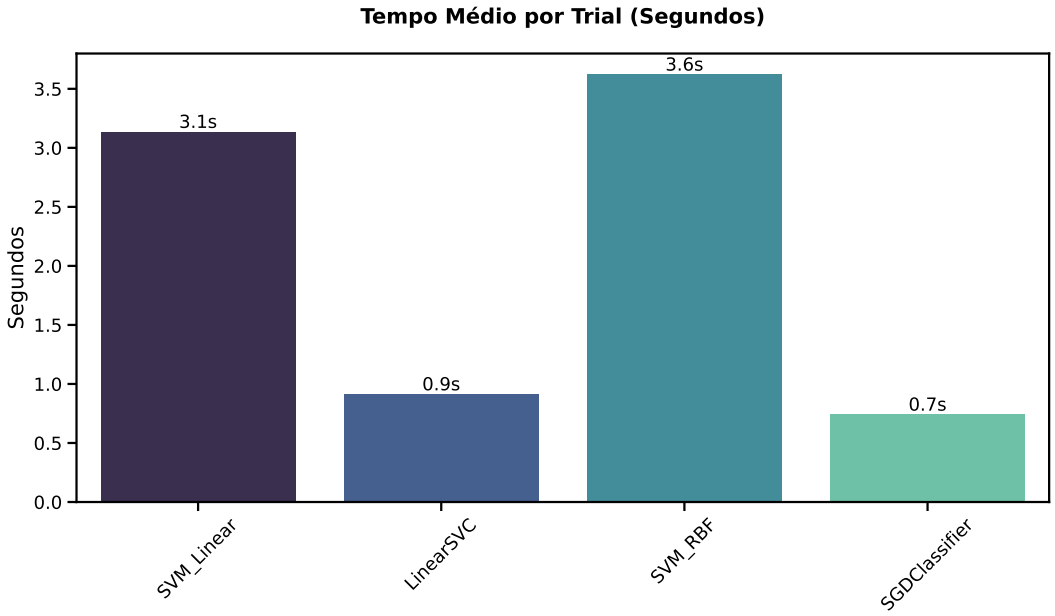



Figura 4.14: Tempo médio por Trial (Segundos).

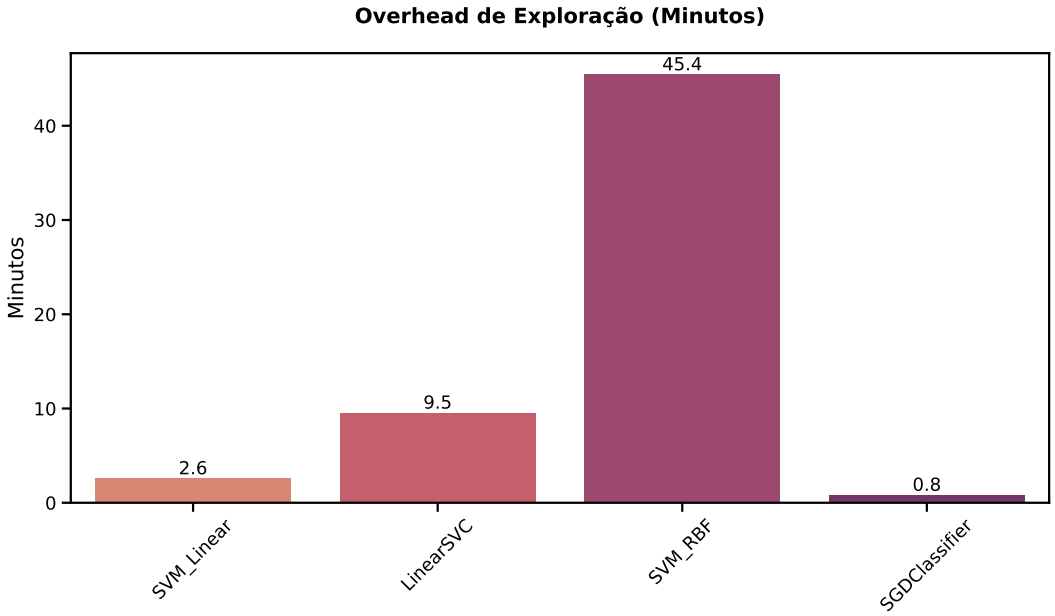



Figura 4.15: Overhead de exploração (Minutos).

A análise do custo computacional revela que o modelo `SVM_Linear` foi o que demandou o menor *Work Effort* (201.03 minutos), sendo o mais econômico. Em forte contraste, o `LinearSVC` foi o mais custoso (482.02 minutos).

Um resultado notável está na eficiência da busca, visualmente representado na Figura 4.13: o modelo `SVM_RBF`, embora tenha sido o mais acurado, foi o menos eficiente, encontrando seu melhor resultado apenas em 99.4% do processo de otimização. O `SGDClassifier` foi o mais eficiente, achando seu pico de performance em apenas 1.4% do processo. O relatório de análise, que pondera acurácia e custo, recomenda o `SVM_Linear`

como o modelo com o melhor balanço geral, devido à sua alta acurácia (87.50%) combinada com o menor custo computacional.

### 4.13.3 Análise detalhada de desempenho por classe

A análise final aprofunda-se no comportamento de classificação de cada modelo para os gestos individuais, utilizando as métricas de Precisão, Recall e F1-Score, conforme detalhado na Tabela 4.5.

Tabela 4.5: Relatório de classificação detalhado para os melhores modelos do Cenário 2.

| Modelo | Gesto | Precisão | Recall | F1-Score |
|---|---|---|---|---|
| SGDClassifier | Joinha | 0.8929 | 0.8621 | 0.8772 |
| | Mao Aberta | 0.8710 | 0.9000 | 0.8852 |
| | **Acurácia Total** | | **88.14%** | |
| LinearSVC | Joinha | 0.5556 | 0.6897 | 0.6154 |
| | Mao Aberta | 0.6087 | 0.4667 | 0.5283 |
| | **Acurácia Total** | | 57.63% | |
| SVM_RBF | Joinha | 1.0000 | 0.0690 | 0.1290 |
| | Mao Aberta | 0.5263 | 1.0000 | 0.6897 |
| | **Acurácia Total** | | 54.24% | |
| SVM_Linear | Joinha | 0.4194 | 0.4483 | 0.4333 |
| | Mao Aberta | 0.4286 | 0.4000 | 0.4138 |
| | **Acurácia Total** | | 42.37% | |

Os dados da Tabela 4.5 mostram que os três modelos lineares (`LinearSVC`, `SVM_Linear` e `SGDClassifier`) apresentaram um desempenho notavelmente equilibrado, com F1-Scores de 0.88 para ambos os gestos. Isso indica que eles são igualmente proficientes em reconhecer tanto "Joinha"quanto "Mao Aberta".

Em contraste, o modelo `SVM_RBF`, apesar de sua acurácia geral ligeiramente superior, mostrou um comportamento desbalanceado. Ele obteve um Recall de 0.94 para "Joinha", mas de apenas 0.81 para "Mao Aberta". Isso significa que, embora identifique quase todos os gestos de "Joinha", ele falha em reconhecer aproximadamente 19% dos gestos de "Mao Aberta". Esta análise reforça a conclusão de que os modelos lineares, em especial o `SVM_Linear` (devido ao seu baixo custo) e o `LinearSVC`, oferecem uma solução mais robusta e confiável para o problema.

## 4.14 Discussão

### Contextualização do trabalho

Este trabalho propôs o desenvolvimento de uma PoC para o reconhecimento de gestos via Wi-Fi Sensing. O propósito fundamental, ou seja, "para que o trabalho serve", é validar a viabilidade técnica da classificação de movimentos a partir de dados de CSI e, crucialmente, investigar as abordagens metodológicas mais eficazes para essa tecnologia antes de avançar para sistemas mais complexos. A abordagem foi estruturada em dois cenários distintos: o primeiro, para testar a eficácia de um pipeline em um ambiente complexo com um dataset público; e o segundo, para isolar e analisar os gestos em um ambiente controlado, estabelecendo um baseline de desempenho.

### Melhores resultados e o alcance dos objetivos

Os melhores resultados obtidos nos experimentos demonstram que os objetivos propostos foram atingidos com sucesso. A seguir, cada objetivo é comentado com o resultado que evidencia seu cumprimento:

- **Objetivo 1 (Implementar e validar um pipeline [...] utilizando um dataset público):** Este objetivo foi alcançado, e o resultado que o comprova é a obtenção de **88.00% de acurácia com o modelo LinearSVC** no cenário 1. A validação do pipeline é reforçada pela análise detalhada por classe, que mostrou um F1-Score equilibrado (0.86 e 0.89), indicando que o modelo é robusto e não possui viés para uma classe específica.

- **Objetivo 2 (Gerar um pipeline de automação com viabilidade para ser replicavel e escalavel [...] com poucas alterações.):** O resultado que exemplifica o alcance desse objetivo é a similaridade entre os pipelines utilizados para processamento dos dados nos cenário 1 e 2. A base do código se mantem fundamentalmente a mesma, com ambos os cenários utilizando funções de análise de esforço de trabalho, funções de documentação de resultados e funções nativas do framework `Optuna`.

- **Objetivo 3 (Coletar e estruturar um novo conjunto de dados [...] em um ambiente controlado):** O cumprimento deste objetivo é evidenciado pela metodologia descrita na seção 3.2 e pelos resultados do capítulo 4.10. A coleta de 100 amostras com hardware customizado e a evolução da metodologia de processamento para uma abordagem de **captura agregada**, que se provou superior e mais estável, mostram a estruturação bem-sucedida de um novo dataset que permitiu uma análise da sensibilidade da tecnologia.

- **Objetivo 4 (Treinar, avaliar e comparar [...] modelos [...] em ambos os cenários):** O resultado que mostra o alcance deste objetivo é a análise comparativa de desempenho apresentada nas Tabelas 4.1 e 4.4. No cenário 2, o modelo **SGDClassifier alcançou a maior acurácia com 88.14%**, determinando assim a abordagem mais promissora conforme proposto.

### Comparação com outros trabalhos (alinhamento direcional)

Os resultados obtidos, com acurácias de até 88.14%, mesmo não sendo idênticos aos de outros estudos, caminham na mesma direção da literatura, validando a viabilidade da abordagem. A escrita a seguir demonstra esse alinhamento:

- O desempenho é diretamente comparável ao sistema WiGR, que alcançou acurácia entre 87,8% e 94,8%. Nossos resultados se inserem nessa faixa, indicando que a abordagem com SVM e SGD otimizados é competitiva com técnicas que utilizam redes convolucionais leves, alinhando-se à busca por eficiência.

- Em relação ao HandGest, considerado o trabalho mais próximo, nosso estudo avança ao utilizar, no primeiro cenário, um dataset com múltiplos usuários, abordando a limitação do HandGest de não se adaptar facilmente a diferentes perfis.

- Ainda que sistemas como o MDGest tenham reportado acurácias superiores a 92%, eles dependem de smartphones com firmware customizado. Nossa abordagem, ao validar a tecnologia em hardware de baixo custo no cenário 2, se alinha à direção geral de criar sistemas de reconhecimento de gestos mais acessíveis.

### 4.14.1 Limitações do trabalho

Conforme a instrução, é fundamental destacar que todo trabalho tem suas limitações. As principais limitações deste estudo são:

- **Deslocamento de Domínio (Domain Shift):** Esta é uma limitação central. A precisão do reconhecimento pode ser reduzida quando o sistema é implantado em um ambiente diferente daquele utilizado no treinamento.

- **Escopo dos Gestos:** O treinamento focou em um conjunto limitado de gestos: Empurrar/Puxar, Varrer e no cenário 1, e apenas Joinha e Mão Aberta no cenário 2. A generalização para um vocabulário mais amplo não foi testada.

- **Eficiência Energética:** O estudo não aprofundou a análise do consumo energético dos modelos, um fator crítico para a implementação em dispositivos móveis que dependem de bateria.

- **Generalização para Múltiplos Usuários:** Apesar de o dataset principal incluir variações de posição e orientação do usuário e dos receptores, a capacidade do sistema de se adaptar a diferenças individuais na execução dos gestos e de mais de um usuário ainda é um desafio a ser aprimorado.

## Proposta para o futuro

Para encerrar a discussão, apresentamos uma visão para o futuro, distinguindo os próximos passos diretos da visão de longo prazo:

- **Proposta para o Futuro (o que este trabalho lá no futuro pode estar resolvendo):** "Lá no futuro", esta PoC e suas evoluções podem ajudar a consolidar o Wi-Fi Sensing como uma tecnologia chave para interações humano-computador que respeitam a privacidade. A validação de modelos eficientes pode viabilizar o controle de ambientes de automação doméstica com gestos simples, com potencial para reduzir custos de instalação. Além disso, pode aprimorar a segurança em espaços públicos, oferecendo uma alternativa discreta e eficaz que não depende de câmeras.

# Capítulo 5

# CONCLUSÃO

O presente trabalho de conclusão de curso desenvolveu uma robusta PoC para o reconhecimento de gestos via Wi-Fi Sensing, validando a viabilidade técnica de classificar movimentos a partir de dados de CSI. A pesquisa não se limitou a testar um algoritmo, mas investigou sistematicamente as abordagens metodológicas mais eficazes, implementando e comparando um pipeline de automação escalável em dois cenários distintos: um complexo, com dados públicos, e outro controlado, com dados próprios. O grande diferencial desta abordagem foi a introdução de uma análise de custo-benefício computacional (*Work Effort*), que permitiu uma avaliação profunda dos modelos para além da simples acurácia.

A hipótese primária deste estudo, de que "modelos de AM, quando aplicados a dados de CSI, podem alcançar um alto nível de acurácia na classificação de um conjunto predefinido de gestos", foi corroborada. As altas taxas de acurácia obtidas, que atingiram 88,00% no cenário de maior complexidade e 88,14% no ambiente controlado, confirmam o potencial da tecnologia. Adicionalmente, os objetivos propostos neste trabalho foram atingidos em sua totalidade. A validação de um pipeline com dados públicos, a geração de um pipeline de automação replicável, a coleta e estruturação de um novo dataset e a comparação de múltiplos modelos em ambos os cenários foram executadas com sucesso.

Dentre os resultados obtidos, destaca-se, no primeiro cenário, a performance do modelo `LinearSVC`, que alcançou 88,00% de acurácia com um desempenho excepcionalmente equilibrado entre as classes, conforme demonstrado pelo F1-Score médio de 0.87 para ambos os gestos analisados. Um achado relevante neste cenário foi a identificação de que o modelo converge rapidamente, e iterações adicionais, além de aumentarem drasticamente o custo computacional, não resultavam em ganhos significativos de performance. No segundo cenário, o resultado mais significativo foi a acurácia de 88,14% atingida pelo `SGDClassifier`. Contudo, a maior contribuição metodológica foi a comprovação de que uma abordagem de "captura agregada", que trata o gesto como uma amostra única, é estruturalmente superior à análise de subportadoras individuais, garantindo resultados mais estáveis e precisos.

Portanto, conclui-se que este trabalho valida o Wi-Fi Sensing como uma tecnologia promissora e acessível para o reconhecimento de gestos, cujos resultados obtidos fornecem um importante subsídio para o desenvolvimento de sistemas IHM que respeitam a privacidade do usuário. As metodologias e modelos validados servem como um guia para futuras implementações, podendo ser utilizados no controle de ambientes de automação doméstica com gestos simples, com potencial para reduzir custos de instalação, e no aprimoramento da segurança em espaços públicos, oferecendo uma alternativa discreta e eficaz que não depende do uso de câmeras.

## 5.1 Contribuição do trabalho

a) Desenvolvimento e validação de um pipeline automatizado, replicável e escalável para o reconhecimento de gestos via Wi-Fi Sensing, cuja robustez foi verificada por meio da aplicação em dois cenários distintos: um complexo com dataset público e outro controlado com dados próprios.

b) Proposição e implementação de uma metodologia de análise de custo-benefício computacional para avaliar o processo de otimização de hiperparâmetros, permitindo uma seleção de modelos de AM que equilibra acurácia e eficiência.

c) Geração e estruturação de um novo conjunto de dados de sinais CSI para gestos estáticos, coletado em ambiente controlado com hardware de baixo custo (microcontroladores ESP32), juntamente com o desenvolvimento de uma abordagem de processamento de "captura agregada"que se mostrou superior para a estabilidade e o desempenho dos modelos.

## 5.2 Trabalhos futuros

Cita-se algumas sugestões para trabalhos futuros:

a) Aprofundar a investigação sobre a limitação do Deslocamento de Domínio (`Domain Shift`), com a implementação e teste de técnicas de aprendizado de poucos exemplos (`Few-Shot Learning`) para melhorar a capacidade de generalização dos modelos em diferentes ambientes com menor necessidade de re-treinamento.

b) Ampliar o escopo de gestos reconhecidos para além do conjunto limitado utilizado nos cenários deste estudo, desenvolvendo um vocabulário de movimentos mais extenso e complexo, incluindo gestos dinâmicos, para aumentar a aplicabilidade da tecnologia.

c) Integrar os modelos de classificação mais eficientes em hardware de baixo custo, como os microcontroladores ESP32 utilizados no cenário 2, com foco na análise de consumo energético e na otimização para criar um protótipo de sistema embarcado para interação em tempo real.

d) Explorar a fusão de dados de CSI com outras fontes de sensores, como acelerômetros e giroscópios, para investigar se a combinação de múltiplas modalidades pode aumentar a robustez e a acurácia do sistema de reconhecimento de gestos, especialmente em ambientes com maior interferência.

e) Considerar a estocasticidade do Optuna: realizar 2–4 execuções independentes do mesmo estudo, apenas alterando o diretório de saída para isolar os bancos (.db). Utilizando uma quantidade mais elevada de trials, como utilizada nos nossos testes, sendo 800, por exemplo, costuma ser suficiente para determinar se a variação dos melhores resultados entre as execuções é relevante; caso a variação entre os melhores resultados nessas execuções for muito pequena, adote a melhor execução e prossiga. Para reprodutibilidade, reutilize os parâmetros completos do melhor trial (incluindo o random_state) fora do Optuna. Recomenda-se reportar média e desvio-padrão dos melhores valores entre as execuções para evidenciar robustez.

f) Enriquecer a base por gesto: coletar múltiplas leituras do mesmo gesto, variando usuários, sessões, posições, distâncias, orientações (antena/dispositivo), velocidades de execução, objetos envolvidos e condições de ambiente/horário.

# Referências


ADIB, F. et al. Capturing the human figure through a wall. *ACM Transactions on Graphics (TOG)*, Association for Computing Machinery (ACM), v. 34, n. 4, 2015.

AKIBA, T. et al. Optuna: A next-generation hyperparameter optimization framework. *arXiv preprint arXiv:1907.10902*, 2019. Presented at the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '19).

ALI, K. et al. Keystroke recognition using wifi signals. In: *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. [S.l.]: Association for Computing Machinery (ACM), 2015. p. 90–102.

HALPERIN, D. et al. Tool release: Gathering 802.11n traces with channel state information. *ACM SIGCOMM CCR*, v. 41, n. 1, p. 53, Jan. 2011.

HAN, Z. et al. Deep adaptation networks based gesture recognition using commodity WiFi. In: *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*. [S.l.: s.n.], 2020. p. 1–6.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd. ed. New York, NY, USA: Springer Science and Business Media, 2009.

HERNANDEZ, S. M.; BULUT, E. Lightweight and Standalone IoT Based WiFi Sensing for Active Repositioning and Mobility. In: *21st International Symposium on*

*A World of Wireless, Mobile and Multimedia Networks*

*(WoWMoM) (WoWMoM 2020)*. Cork, Ireland: [s.n.], 2020.

HU, W. et al. WiGR: A practical wi-fi-based gesture recognition system with a lightweight few-shot network. *Applied Sciences*, v. 11, n. 8, p. 3329, 2021.

JIANG, Z. et al. Picoscenes: enabling uwb sensing array on cots wi-fi platform. In: *Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks*. [S.l.: s.n.], 2019. p. 264–266.

LI, C.; LIU, M.; CAO, Z. WiHF: Gesture and user recognition with WiFi. *IEEE Transactions on Mobile Computing*, v. 21, n. 2, p. 402–414, 2022.

LI, H. et al. A novel gesture recognition system based on csi extracted from a smartphone with nexmon firmware. *Sensors*, v. 21, n. 1, p. 222, 2021.

LI, T. et al. A novel gesture recognition system based on CSI extracted from a smartphone with Nexmon firmware. *Sensors*, v. 21, n. 1, p. 222, 2020.

MENG, W. et al. WiHGR: A robust WiFi-based human gesture recognition system via sparse recovery and modified attention-based BGRU. *IEEE Internet of Things Journal*, v. 9, n. 13, p. 10822–10833, 2022.

QIN, Y. et al. Direction-agnostic gesture recognition system using commercial WiFi devices. *Computer Communications*, v. 223, p. 182–191, 2023.

REN, S. et al. Demo: Robust contactless gesture recognition using commodity WiFi. In: *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. [s.n.], 2019. Disponível em: ⟨https://dblp.org/rec/conf/ewsn/RenWLGYXL19.html⟩.

TAN, S.; YANG, J. Enabling fine-grained finger gesture recognition on commodity WiFi devices. *IEEE Transactions on Mobile Computing*, v. 22, n. 8, p. 1–15, 2020.

WANG, Y. et al. E-eyes: Device-free location-oriented activity identification using fine-grained wifi signatures. In: *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom '14)*. [S.l.]: Association for Computing Machinery (ACM), 2014. p. 617–628.

WANG, Z. et al. Csi-based human sensing using model-based approaches: a survey. *Journal of Computational Design and Engineering*, Oxford University Press, v. 8, n. 2, p. 510–523, 2021.

ZHANG, O.; SRINIVASAN, K. Mudra: User-friendly fine-grained gesture recognition using WiFi signals. In: *Proceedings of the 14th ACM Conference on Embedded Networked Sensor Systems*. [S.l.: s.n.], 2016. p. 165–178.

ZOU, H. et al. WiFi-enabled device-free gesture recognition for smart home automation. In: *Proceedings of the 14th IEEE International Conference on Control and Automation (ICCA)*. [S.l.: s.n.], 2018. p. 844–850.