



**Universidade de Brasília  
Faculdade de Tecnologia  
Departamento de Engenharia Elétrica**

**CNN para Reconhecimento de Atividades em Ambientes  
Prediais com Rede de Sensores Wi-Fi**

Leandro de Sousa Monteiro

**PROJETO FINAL DE CURSO  
ENGENHARIA DE CONTROLE E AUTOMAÇÃO**

Brasília  
2024

**Universidade de Brasília  
Faculdade de Tecnologia  
Departamento de Engenharia Elétrica**

**CNN para Reconhecimento de Atividades em Ambientes Prediais  
com Rede de Sensores Wi-Fi**

Leandro de Sousa Monteiro

Projeto Final de Curso submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação

Orientador: Prof. Dr. Adolfo Bauchspiess

Brasília  
2024

SS474c Sousa Monteiro, Leandro de.  
CNN para Reconhecimento de Atividades em Ambientes Pre-  
diais com Rede de Sensores Wi-Fi / Leandro de Sousa Monteiro;  
orientador Adolfo Bauchspiess. -- Brasília, 2024.  
57 p.

Projeto Final de Curso (Engenharia de Controle e Automação)  
-- Universidade de Brasília, 2024.

1. Automação Predial. 2. CSI Learning. 3. Reconhecimento  
de Atividades Humanas. 4. Machine Learning. I. Bauchspiess,  
Adolfo, orient. II. Título

**Universidade de Brasília**  
**Faculdade de Tecnologia**  
**Departamento de Engenharia Elétrica**

**CNN para Reconhecimento de Atividades em Ambientes  
Prediais com Rede de Sensores Wi-Fi**

Leandro de Sousa Monteiro

Projeto Final de Curso submetido como requi-  
sito parcial para obtenção do grau de Enge-  
nheiro de Controle e Automação

Trabalho aprovado. Brasília, 12 de Setembro de 2024:

---

**Prof. Dr. Adolfo Bauchspiess,**  
**UnB/FT/ENE**  
Orientador

---

**Prof. Dr. Geovany Araújo Borges,**  
**UnB/FT/ENE**  
Professor Associado

---

**Prof. Dr. Alexandre Ricardo S. Romariz,**  
**UnB/FT/ENE**  
Professor Associado

Brasília  
2024

# Resumo

A proposta deste trabalho é desenvolver um sistema de *Human Activity Recognition* (HAR) em ambientes prediais, utilizando a tecnologia de *Wi-Fi Sensing* para capturar dados de *Channel State Information* (CSI) e uma *Convolutional Neural Network* (CNN). A coleta de dados foi realizada por uma rede de sensores ESP32 distribuídos estrategicamente no ambiente, onde os sinais de CSI são convertidos em imagens que alimentam o modelo de CNN. O sistema visa identificar atividades humanas de forma não intrusiva (*device-free*) e preservando a privacidade dos ocupantes, ao dispensar o uso de câmeras e basear-se apenas nas alterações do CSI causadas pelo movimento humano. O trabalho explora o potencial do *Wi-Fi Sensing* como uma solução viável e econômica, aproveitando infraestruturas de redes sem fio já existentes em muitos edifícios, além de destacar os desafios da coleta de dados precisos e a implementação do modelo de CNN para HAR em ambientes prediais. Os resultados obtidos pela CNN indicaram uma acurácia máxima de 48,33% no conjunto de dados de validação, com ocorrência de *overfitting* após a época 163. A rede de sensores foi capaz de adquirir dados de CSI, porém o processo de limpeza de dados apresentou cerca de 30% de inserção de linhas para ajustar os arquivos, apontando a necessidade de melhorias na aquisição.

**Palavras-chave:** Automação Predial, CSI Learning, Reconhecimento de Atividades Humanas, Machine Learning.

# Abstract

The aim of this work is to develop a human activity recognition (HAR) system for building environments using Wi-Fi Sensing technology to capture Channel State Information (CSI) data and a Convolutional Neural Network (CNN). Data collection was carried out through a strategically distributed network of ESP32 sensors, where the CSI signals are transformed into images that feed the CNN model. The system seeks to identify human activities in a non-intrusive (device-free) manner, preserving occupants' privacy by eliminating the need for cameras and relying solely on CSI changes caused by human movement. This study explores the potential of Wi-Fi Sensing as a viable and cost-effective solution, leveraging existing wireless network infrastructures in many buildings, while also addressing the challenges of accurate data collection and CNN model implementation for HAR in building environments. The CNN results showed a maximum accuracy of 48,33% on the validation set, with overfitting starting after epoch 163. The sensor network was able to capture CSI data, but about 30% of the lines had to be added during the cleaning process to fix the files, indicating a need for better data acquisition methods.

**Keywords:** Building Automation, CSI Learning, Human Activity Recognition, Machine Learning.

# Lista de Figuras

Figura 1.1	Exemplo de automação usando HAR para eficiência energética.(Bauchspiess <i>et al.</i> , 2024) . . . . .	14
Figura 1.2	Visão geral de um sistema de HAR que utiliza CNN com rede de sensores. Em que os dados de CSI são adquiridos pela rede de sensores Wi-Fi, processados para limpeza e filtragem, as imagens são geradas para se adequarem à entrada da Rede Neural Convolucional (CNN) e, por fim, o reconhecimento das atividades humanas é feito através da classificação da CNN. . . . .	15
Figura 1.3	Processo genérico de uma tarefa HAR. Onde a primeira etapa é sensoria-mento, seguida do processamento dos dados brutos, adequando-os para o método de classificação escolhido. Após essas etapas de aquisição e pro-cessamento de dados, é feito o reconhecimento das atividades humanas e o sistema toma uma decisão (de acordo com a sua finalidade) baseada na atividade reconhecida. Adaptado de (Bian <i>et al.</i> , 2022). . . . .	16
Figura 1.4	O sinal OFDM pode ser descrito como um conjunto de subportadoras espaçadas de forma ortogonal, que enviam o mesmo pacote com o intuito de garantir que a informação chegue no receptor. As subportadoras pos-suem espectros sobrepostos, mas a ortogonalidade das frequências evita interferências.(Keysight, 2024) . . . . .	17
Figura 2.5	Visão geral do ReWiS em alto nível. (Bahadori; Ashdown; Restuccia, 2022)	23
Figura 2.6	Adaptação de Domínio Prototípica com Poucos Exemplos ( <i>Few-Shot Pro-totypical Domain Adaptation</i> ) a) Aprender características com um grande conjunto de treinamento no ambiente base E1 (contendo 4 protótipos: vermelho, verde, azul e amarelo). b) Adaptação com poucos exemplos ( <i>Few-Shot</i> ) para o novo ambiente E2, com 5 exemplos e 4 classes. Clas-sificação por distância euclidiana. c) Classificação euclidiana dos dados de consulta no ambiente E2. d) O novo ambiente E3 leva a protótipos adaptados (nota: pode alterar a classificação).(Bauchspiess <i>et al.</i> , 2024) .	24
Figura 3.7	ESP32-WROOM-32UE . . . . .	25
Figura 3.8	Vista superior da sala de estudos do LARA, em que está representada a topologia da rede de sensoriamento. M1 e M2 são os mestres, que recebem os pacotes dos sensores S1, S2, S3, S4. Os pacotes são enviados em ciclos, que começam com o "comando" do M1, enviado via transmissão ( <i>broadcast</i> ) para todos os sensores. A conexão utiliza o protocolo ESP-NOW.	28
Figura 3.9	Fluxograma do código de aquisição de CSI. (Bauchspiess <i>et al.</i> , 2024) . .	29

Figura 3.10	Diagrama no tempo de aquisição de CSI com perdas de dados, devido à colisões múltiplas. . . . .	30
Figura 3.11	Fotografia que mostra a disposição dos sensores na sala de estudos do LARA. . . . .	31
Figura 3.12	Esboço da região de 1ª zona de Fresnel e a topologia de sensores na sala de estudos do LARA (Vista superior). . . . .	32
Figura 3.13	Imagens capturadas do vídeo de <i>Ground Truth</i> dos experimentos na sala de estudos do LARA. Atividades de sala vazia, caminhando, sentado, em pé. . . . .	33
Figura 3.14	Representação da posição e orientação em que as atividades foram executadas na sala de estudos do LARA, durante os experimentos. (Vista superior). . . . .	33
Figura 3.15	Representação temporal da amplitude dos dados de CSI para algumas portadoras do sensor 6. Onde a cada janela marcada entre os traços vermelhos se passam 10 s. . . . .	34
Figura 3.16	Arquivo .csv contendo dados de CSI, mostrado apenas em parte na última coluna à direita, e informações adicionais. Em destaque, CSI_DATA (identificação para iniciar nova linha do arquivo .csv), idS (identificação do sensor), sC (ciclo em que esse dado foi adquirido), t_M(ms) (tempo decorrido desde o início das aquisições), tin_cycle (tempo decorrido desde o início do ciclo atual). . . . .	35
Figura 3.17	Imagens geradas para a atividade "em pé" de uma rotina de experimentos. À esquerda, as imagens .png utilizadas como entrada para a neural convolucional, e à esquerda, imagens geradas após a transformação PCA. . . . .	37
Figura 4.18	Estrutura da LSTM utilizada por Zhang. Onde há uma camada escondida <i>S-shaped</i> . (Zhang; Qu; Wang, 2020) . . . . .	39
Figura 4.19	Processo de treinamento do design de RL proposto por Yongsen Ma, para reconhecimento de atividades humanas. (Ma <i>et al.</i> , 2021) . . . . .	40
Figura 4.20	ProtoNet utilizada no trabalho de Bahadori. (Bahadori; Ashdown; Restuccia, 2022) . . . . .	41
Figura 4.21	CNN utilizada no trabalho de Zhang. (Zhang <i>et al.</i> , 2022) . . . . .	43
Figura 4.22	Arquitetura da CNN utilizada. Composta por uma camada convolucional seguida de uma função de ativação <i>ReLU</i> , uma camada de <i>pooling</i> , uma camada <i>Flatten</i> para adaptar os dados para entrada da próxima camada (totalmente conectada), mais uma função <i>ReLU</i> , e outra camada totalmente conectada com saída de dimensão 4 (as quatro classes). A função <i>Softmax</i> foi usada para classificação final das atividades baseada em probabilidades. Adaptada de (Bahadori; Ashdown; Restuccia, 2022). . . . .	44
Figura 4.23	Exemplo de identificação de <i>overfitting</i> no gráfico <i>loss x epochs</i> . . . . .	50

Figura 5.24	Gráfico de Perda x Época da CNN. Curva azul representa o treinamento, e a curva laranja representa a validação. O <i>overfitting</i> foi detectado a partir da época 163. . . . .	52
Figura 5.25	Matriz de confusão para o conjunto de dados de validação. Cada linha da matriz representa as atividades reais, enquanto as colunas indicam as atividades previstas pela rede neural. O conjunto de validação é constituído de 30 amostras para cada atividade. . . . .	53

# Lista de tabelas

Tabela 5.1	Taxas de inserções de linhas para alguns arquivos com dados CSI das rotinas de experimentos, adquiridos no LARA. . . . .	51
------------	--------------------------------------------------------------------------------------------------------------------------	----

# Lista de Abreviaturas, Acrônimos e Siglas

ABLSTM	Attention-Based Bi-directional <i>Long Short-Term Memory</i> Attention-Based Bi-directional Long Short-Term Memory .....	22
ABNT	Associação Brasileira de Normas Técnicas .....	20
CNN	Convolutional Neural Network .....	14
CSI	Channel State Information .....	15
ESP-IDF	Espressif IoT Development Framework .....	26
FSL	Few-Shot Learning .....	21
HAR	Human Activity Recognition .....	14
IoT	Internet of Things .....	14
LGPD	Lei Geral de Proteção de Dados .....	15
LSTM	Long Short-Term Memory .....	38
OFDM	Orthogonal Frequency Division Multiplexing .....	16
PCA	Análise de Componentes Principais .....	34
ProtoNet	Prototypical Networks .....	40
ReWiS	Reliable Wi-Fi Sensing Through Few-Shot Multi-Antenna Multi-Receiver CSI Learning .....	22
RL	Reinforcement Learning .....	39
SDK	Software Development Kit .....	26
SVM	Support Vector Machines .....	15
UnB	Universidade de Brasília .....	20

# Lista de símbolos

$\angle$	Fase.....	18
$\log$	Logaritmo.....	39
$\nabla$	Gradiente.....	39
$\otimes$	Produto Tensorial.....	38
$\pi$	pi.....	36
$\sum$	Somatório.....	39
$\theta$	theta.....	39
$e$	Número de Euler.....	18
$Hz$	Hertz.....	35
$j$	Unidade Imaginária.....	18

# Sumário

<b>1</b>	<b>Introdução</b>	<b>14</b>
1.1	Motivação	14
1.2	Contextualização	16
1.2.1	Reconhecimento de Atividades Humanas ( <i>Human Activity Recognition</i> — HAR)	16
1.2.2	<i>Orthogonal Frequency Division Multiplexing</i> — OFDM	16
1.2.3	<i>Channel State Information</i> — CSI	18
1.2.4	Implementação de <i>Wi-Fi Sensing</i> com Sensores Locais	18
1.2.5	Uso de CNN para Classificação de Atividades Humanas	19
1.3	Objetivos	19
1.3.1	Objetivos Gerais	19
1.3.2	Objetivos Específicos	19
<b>2</b>	<b>Estado da Arte</b>	<b>21</b>
2.1	HAR e <i>Wi-Fi Sensing</i>	21
2.2	CSI Learning	22
2.3	Few-Shot HAR	23
<b>3</b>	<b>Materiais e Métodos</b>	<b>25</b>
3.1	ESP32	25
3.1.1	ESP-IDF	26
3.1.2	ESP32 ESP-NOW	26
3.1.3	Aquisição de Dados de CSI com o ESP32	27
3.2	Python	28
3.3	Metodologia	29
3.3.1	Rotinas de Experimentos no Ambiente do LARA	31
3.3.2	Pré-processamento dos Dados	34
<b>4</b>	<b>Algoritmo da Rede Neural</b>	<b>38</b>
4.1	Arquiteturas de Redes Neurais para Reconhecimento de Atividades Humanas (HAR)	38
4.2	Escolha e Estrutura da Rede Neural Convolucional (CNN)	43
4.2.1	Pseudocódigo	44
4.2.2	Implementação	45
<b>5</b>	<b>Resultados e Análise Crítica</b>	<b>51</b>

5.1	<i>Wi-Fi Sensing</i> . . . . .	51
5.2	Desempenho da CNN na Classificação das Atividades . . . . .	52
<b>6</b>	<b>Conclusões e Perspectivas</b> . . . . .	<b>54</b>
6.1	Conclusões . . . . .	54
6.2	Perspectivas Futuras . . . . .	54
	<b>Referências</b> . . . . .	<b>56</b>

# 1 Introdução

Neste capítulo, será apresentada a introdução ao problema abordado, seguida da contextualização da relevância e dos objetivos deste trabalho. A abordagem proposta se destaca pela integração de tecnologias avançadas de redes neurais convolucionais (CNN) com a captura de dados via *Wi-Fi Sensing*, visando o HAR (*Human Activity Recognition*) em ambientes prediais.

## 1.1 Motivação

O uso de IoT (*Internet of Things*) na automação predial emerge como uma estratégia perspicaz para racionalizar o consumo de energia, empregando dados específicos sobre a ocupação de diversas áreas nos edifícios. Nesse contexto, o uso de HAR para automação predial (Figura 1.1) em sistemas como os de climatização e de iluminação tem suas ações baseadas em zonas de uso, onde a quantidade de indivíduos e as atividades desempenhadas pelos mesmos contribuem para as tomadas de decisões desses sistemas.

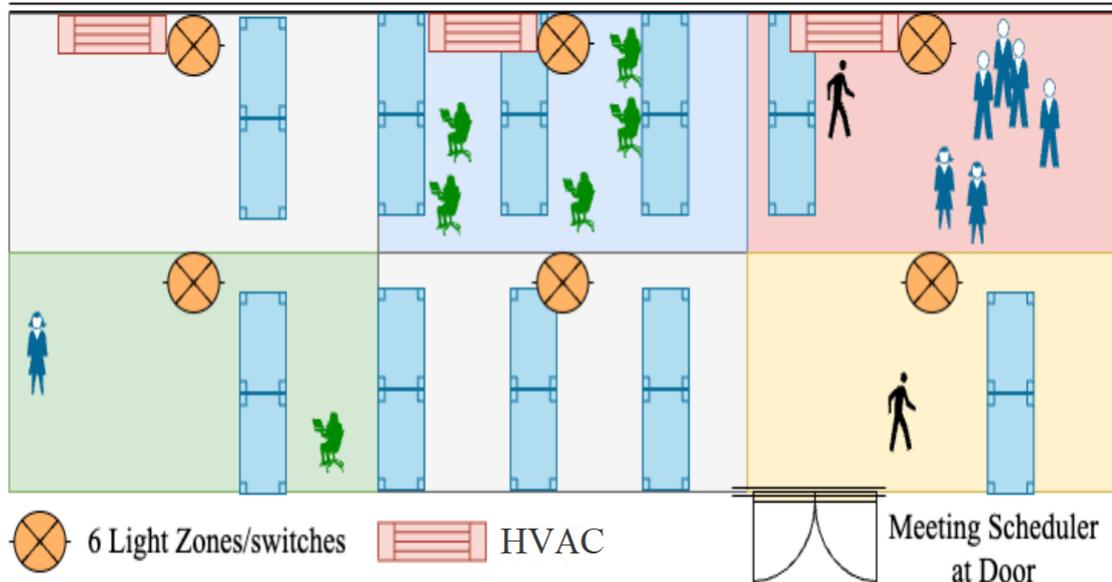


Figura 1.1 – Exemplo de automação usando HAR para eficiência energética. (Bauchspiess *et al.*, 2024)

Investigando trabalhos anteriores (Rodrigues; Silva, 2018), evidencia-se que controle antecipativo alinhado a perfis de ocupação previamente estabelecidos, proporciona uma redução de aproximadamente 26,4% no consumo energético. A aplicação de redes neurais artificiais para processar imagens do ambiente capturadas em tempo real, e reconhecer as atividades humanas, viabiliza a assimilação do padrão de ocupação, considerando tanto a quantidade de indivíduos quanto a natureza das atividades.

No entanto, a utilização de câmeras para realizar o reconhecimento de atividades humanas pode gerar preocupações relacionadas à privacidade, especialmente em função da LGPD (Lei Geral de Proteção de Dados), já que o sistema captura dados visuais diretamente associados às suas identidades. (Brasil, 2018)

Para contornar esse problema, a utilização do *Wi-Fi Sensing*, que utiliza a variação de *Channel State Information* (CSI — Informação de Estado do Canal, tradução livre) para detectar a presença e atividades de indivíduos, apresenta-se como uma alternativa vantajosa. Diferente de câmeras, essa técnica é impessoal, não capturando informações visuais ou biométricas, o que diminui os riscos de violações de privacidade. Além disso, o *Wi-Fi Sensing* oferece uma abordagem eficaz e não intrusiva para detecção de ocupação e atividades humanas, sendo compatível com a grande disposição de dispositivos Wi-Fi nas infraestruturas de redes já presentes em muitos ambientes, sem a necessidade de instalação de novos dispositivos de captura de imagem. (Chen *et al.*, 2019)

O uso de redes neurais convolucionais (CNNs) em HAR com dados de CSI tem se mostrado eficiente (Figura 1.2). As CNNs são capazes de extrair características dos sinais de CSI, transformados em espectrogramas ou imagens, que capturam variações de frequência causadas por movimentos humanos. Esse modelo tem vantagens sobre abordagens tradicionais, como Máquina de Vetores de Suporte (*Support Vector Machines* — SVM, do inglês), por sua capacidade de lidar com dados complexos, garantindo robustez em ambientes dinâmicos e proporcionando uma boa precisão no reconhecimento de atividades como andar, sentar e ficar de pé. (Muaaz *et al.*, 2022)

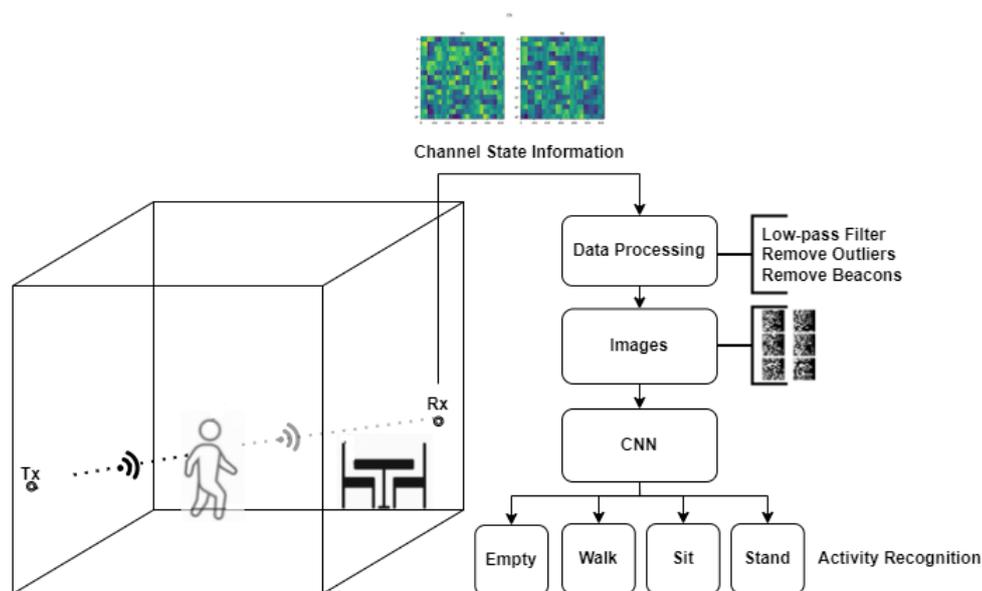


Figura 1.2 – Visão geral de um sistema de HAR que utiliza CNN com rede de sensores. Em que os dados de CSI são adquiridos pela rede de sensores Wi-Fi, processados para limpeza e filtragem, as imagens são geradas para se adequarem à entrada da Rede Neural Convolucional (CNN) e, por fim, o reconhecimento das atividades humanas é feito através da classificação da CNN.

## 1.2 Contextualização

Para um melhor entendimento do problema abordado neste trabalho, esta seção irá explorar conceitos fundamentais como HAR, OFDM, CSI e *Wi-Fi Sensing*. Esses tópicos são essenciais para compreender a base tecnológica utilizada no desenvolvimento da solução proposta.

### 1.2.1 Reconhecimento de Atividades Humanas (*Human Activity Recognition* — HAR)

HAR refere-se ao campo de estudo focado em capturar, reconhecer e analisar as atividades humanas por meio de sensores e técnicas de processamento de dados. HAR tem ganhado grande relevância nos últimos anos, sendo aplicado em uma ampla gama de cenários, como monitoramento de saúde, automação residencial, vigilância e na promoção de uma melhor interação entre humanos e máquinas.

De forma geral, o processo de HAR envolve três etapas principais: a captura de dados por sensores, o processamento desses dados e, finalmente, a tomada de decisões com base nos padrões reconhecidos (Figura 1.3). As técnicas de sensoriamento podem variar amplamente, desde sensores vestíveis, que capturam movimentos corporais, até sensores sem contato, como Wi-Fi e outros sinais de rádio, que detectam atividades através da interferência no caminho de propagação dos sinais. (Bian *et al.*, 2022)

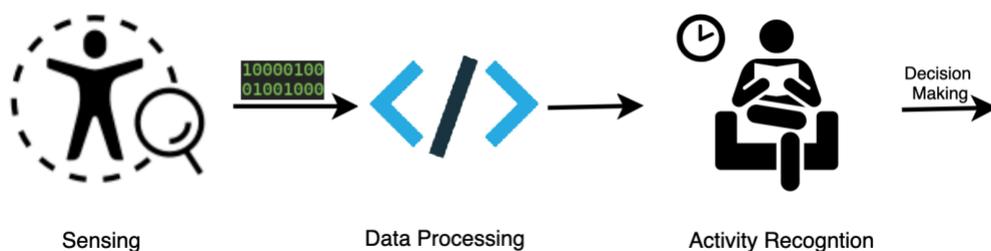


Figura 1.3 – Processo genérico de uma tarefa HAR. Onde a primeira etapa é sensoriamento, seguida do processamento dos dados brutos, adequando-os para o método de classificação escolhido. Após essas etapas de aquisição e processamento de dados, é feito o reconhecimento das atividades humanas e o sistema toma uma decisão (de acordo com a sua finalidade) baseada na atividade reconhecida. Adaptado de (Bian *et al.*, 2022).

### 1.2.2 *Orthogonal Frequency Division Multiplexing* — OFDM

A Multiplexação por Divisão de Frequências Ortogonais (*Orthogonal Frequency Division Multiplexing* — OFDM, do inglês) é uma técnica de modulação, sendo parte do protocolo IEEE 802.11, que engloba a tecnologia Wi-Fi. O OFDM divide o sinal em várias subportadoras, distribuídas de forma ortogonal entre si ao longo de uma faixa de frequências.

Essas subportadoras operam em frequências ligeiramente diferentes umas das outras, e essa ortogonalidade garante que o pico de uma portadora coincida com o ponto de mínimo das portadoras adjacentes, como é representado na Figura 1.4, o que minimiza significativamente a interferência entre as subportadoras. Como resultado, o OFDM consegue enviar o mesmo pacote em cada uma das portadoras, permitindo que múltiplos sinais sejam transmitidos simultaneamente sem causar interferência entre si, aumentando as chances de que o dado chegue de maneira íntegra no receptor. (Alsaify *et al.*, 2022)

Ao distribuir o sinal por múltiplas subportadoras, o OFDM é capaz de lidar melhor com condições de canal desfavoráveis, como o desvanecimento seletivo e as interferências causadas por múltiplos caminhos, que são comuns em ambientes internos. Isso ocorre porque, mesmo que algumas subportadoras sofram atenuação severa devido à interferência, outras podem continuar transmitindo com integridade suficiente, garantindo a recepção global do pacote de dados.

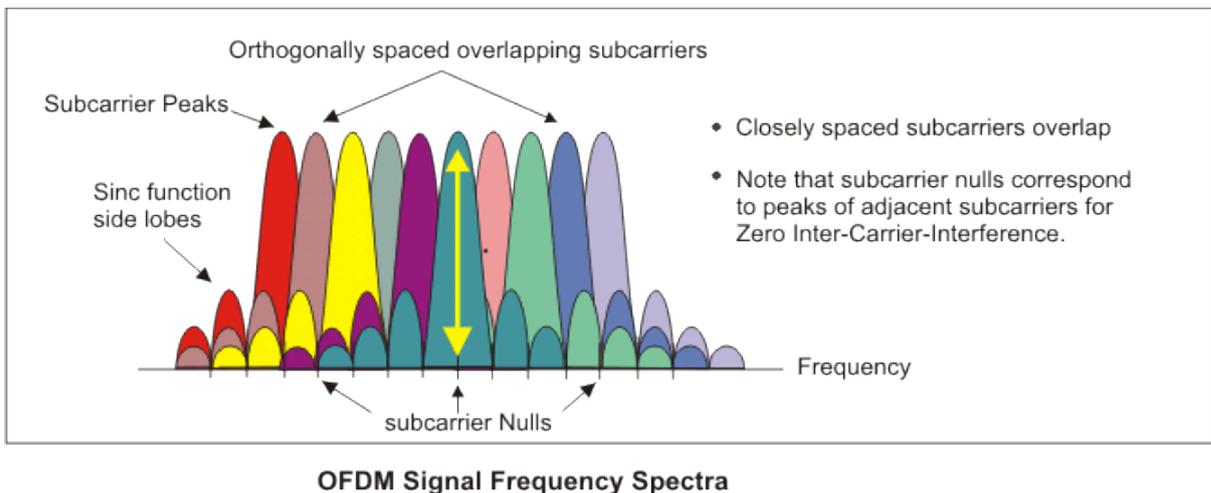


Figura 1.4 – O sinal OFDM pode ser descrito como um conjunto de subportadoras espaçadas de forma ortogonal, que enviam o mesmo pacote com o intuito de garantir que a informação chegue no receptor. As subportadoras possuem espectros sobrepostos, mas a ortogonalidade das frequências evita interferências. (Keysight, 2024)

Quando um pacote chega corretamente ao receptor, ele carrega os valores imaginários e real do CSI da sua subportadora, com o tamanho de 1 byte cada. Ao processar os dados imaginário e real, é possível calcular como o sinal de cada subportadora é afetado em amplitude (o quanto o sinal foi atenuado) e em fase (o quanto o sinal foi deslocado no tempo) durante a propagação através do canal. O CSI, então, fornece uma visão do estado do canal entre o transmissor e o receptor, permitindo a detecção de mudanças ambientais que afetam a propagação do sinal, como movimentos de pessoas, ocupação e posicionamento. (Alsaify *et al.*, 2022)

### 1.2.3 Channel State Information — CSI

Foi a partir da padronização do protocolo IEEE 802.11 que o CSI passou a estar disponível, permitindo a obtenção de dados detalhados sobre as condições de propagação do sinal entre o transmissor e o receptor. Essa camada captura o comportamento do canal em relação a fenômenos como reflexões, refrações, atenuações, e múltiplos caminhos que ocorrem entre as antenas. Esses dados são essenciais para aplicações como reconhecimento de atividades humanas (HAR) usando redes Wi-Fi. (Shen *et al.*, 2021)

O CSI fornece informações ricas sobre o ambiente físico, como mudanças no comportamento do sinal causadas por movimentos humanos. A partir dessas informações, é possível monitorar atividades sem a necessidade de dispositivos adicionados ao corpo dos seres humanos, o que torna essa abordagem não invasiva e protegida em termos de privacidade.

O processo de obtenção da CSI é baseado na medição da resposta de frequência do canal em diferentes subportadoras, resultando em informações que incluem tanto a amplitude quanto a fase do sinal. Alterações nessas características indicam modificações no ambiente, como a presença e o movimento de pessoas. Por exemplo, uma pequena mudança na amplitude ou fase pode ser interpretada como a detecção de uma pessoa caminhando ou mudando de posição em um ambiente controlado. A forma básica do CSI pode ser expressa por

$$H_i = \|H_i\| e^{j\angle H_i}, \quad (1.1)$$

em que a amplitude  $\|H_i\|$  e a fase  $\angle H_i$  do CSI  $H_i$  na  $i$  – ésima subportadora são facilmente afetadas pelo movimento e deslocamento dos transmissores, receptores e das pessoas ao redor. Portanto, em ambientes Wi-Fi densamente implantados, a extração das informações mais representativas das atividades humanas é crucial.

### 1.2.4 Implementação de *Wi-Fi Sensing* com Sensores Locais

Neste trabalho, a implementação do *Wi-Fi Sensing* envolve desenvolver uma rede de sensores Wi-Fi distribuídos em pontos estratégicos de ambientes prediais, com o objetivo de capturar dados de CSI que representam as variações nos sinais de Wi-Fi causadas pelas atividades humanas. O sistema deve ser capaz de detectar atividades como sentar, levantar, andar e a ausência de ocupação em tempo real. Bem como integrar os dispositivos ESP32 para criar uma infraestrutura de monitoramento, permitindo a comunicação entre sensores de forma eficiente via Wi-Fi, utilizando o CSI como principal fonte de dados.

Esta etapa visa garantir a privacidade dos usuários por meio da aplicação de *Wi-Fi*

*Sensing*, que evita o uso de câmeras ou sensores invasivos. Com isso, os dados obtidos se tornam impessoais, preservando a identidade dos ocupantes e respeitando regulamentações a respeito de privacidade.

### 1.2.5 Uso de CNN para Classificação de Atividades Humanas

Com o intuito de fazer a classificação das atividades humanas, este trabalho visa aplicar técnicas de pré-processamento de dados, como a transformação dos sinais CSI em imagens, que serão utilizadas para o treinamento da CNN. Esse pré-processamento deve incluir a remoção de ruídos, filtragem e aplicação de métodos como PCA para reduzir a dimensões dos dados. Bem como desenvolver um modelo de CNN treinado com dados de CSI coletados pela rede de sensores Wi-Fi para classificar as atividades humanas no ambiente monitorado. O modelo deve ser capaz de diferenciar atividades como caminhar, sentar, ficar em pé e identificar a ausência de pessoas no ambiente. É essencial avaliar a performance da CNN em termos de acurácia, generalização e robustez, ajustando parâmetros das camadas e a arquitetura da rede neural para melhorar os resultados.

## 1.3 Objetivos

Diante do desafio de integrar o Reconhecimento de Atividades Humanas (HAR) com *Wi-Fi Sensing*, bem como dos conceitos fundamentais que envolvem essa problemática, os objetivos estabelecidos nesta seção têm como propósito guiar e estruturar a escolha da metodologia adotada neste estudo.

### 1.3.1 Objetivos Gerais

O presente trabalho tem como objetivo desenvolver uma solução eficiente para o reconhecimento de atividades humanas em ambientes prediais, utilizando redes de sensores Wi-Fi e redes neurais convolucionais (CNN), permitindo abordar tanto a coleta de dados de forma não invasiva, quanto o processamento dos dados para reconhecimento de padrões de atividades humanas.

### 1.3.2 Objetivos Específicos

A divisão dos objetivos específicos se dá em duas frentes:

- Implementação do *Wi-Fi Sensing*:
  - Construir um banco de dados CSI com rotinas de ocupação e atividades humanas, para uso no treinamento e validação do modelo de reconhecimento de atividades com base em redes neurais convolucionais.

- Compreender e adaptar códigos pré-existentes, que apresentam pouca ou nenhuma documentação. Essa tarefa representa um grande desafio, pois exige uma compreensão detalhada da sua estrutura e funcionalidade, além de uma reescrita cuidadosa para garantir que o código seja compreensível e sustentável para futuras implementações.
- CNN para a classificação de atividades:
  - Avaliar a performance da CNN na classificação de atividades humanas a partir dos dados de CSI, ajustando os hiperparâmetros e explorando diferentes abordagens para melhorar a precisão e robustez do modelo em ambientes prediais dinâmicos.

## 2 Estado da Arte

Neste capítulo, serão discutidos os principais trabalhos relacionados à área de reconhecimento de atividades humanas (HAR), especialmente aqueles que utilizam técnicas de CSI e *Wi-Fi Sensing*. Além disso, abordaremos como diferentes arquiteturas de redes neurais, como CNN, LSTM e *Few-Shot Learning* (FSL), têm sido exploradas nessa área.

### 2.1 HAR e *Wi-Fi Sensing*

A tecnologia de reconhecimento de atividades humanas (HAR) baseada em Wi-Fi e no uso de CSI vem ganhando relevância por sua capacidade de monitorar atividades sem a necessidade de dispositivos de vestuário ou câmeras, o que preserva a privacidade e oferece praticidade em diversos cenários. Nesse contexto, o uso do CSI permite que as mudanças nos sinais de rádio, causadas pelo movimento humano, sejam analisadas para detectar e classificar atividades como caminhar, ficar em pé, sentar ou a ausência de pessoas em um ambiente. (Muaaz *et al.*, 2022)

HAR com Wi-Fi CSI utiliza a variação de sinais causada pelo corpo humano, capturando essas alterações nas subportadoras do canal de comunicação. O *Wi-Fi Sensing*, por exemplo, combina CSI com técnicas de aprendizado profundo, como redes neurais convolucionais (CNN), para detectar padrões no deslocamento Doppler dos sinais de Wi-Fi e classificar as atividades humanas. A arquitetura de HAR baseada em CSI se destaca por ser econômica, uma vez que aproveita infraestruturas de Wi-Fi já existentes, e por ser não invasiva, sem comprometer a privacidade dos usuários. Os desafios incluem garantir que as variações no ambiente não afetem a precisão do reconhecimento. (Muaaz *et al.*, 2022)

O *Wi-Fi Sensing* baseia-se na premissa que interações humanas entre emissores e receptores de Wi-Fi impactarão nas propriedades do sinal Wi-Fi, torna-se viável implementar o reconhecimento passivo de atividades humanas por meio dessa tecnologia. Devido à extensa presença de sinais de Wi-Fi em ambientes internos, o HAR através de Wi-Fi se revela uma possível solução econômica e isenta de custos adicionais. Adicionalmente, os sistemas passivos de HAR baseados em Wi-Fi dispensam a necessidade dos usuários utilizarem qualquer dispositivo para o reconhecimento. (Chen *et al.*, 2019)

Vários estudos têm demonstrado a eficácia do uso de dados CSI para HAR, com técnicas de aprendizado de máquina e aprendizado profundo sendo aplicadas para a classificação. (Bahadori; Ashdown; Restuccia, 2022), (Muaaz *et al.*, 2022), (Ma; Zhou; Wang, 2019) Notavelmente, o *Wi-Fi Sensing*, foco deste trabalho, apresenta uma abordagem inovadora para reconhecer atividades humanas a partir de interferências nos sinais e independentes do

ambiente, extraídas de dados Wi-Fi CSI. Essa abordagem visa superar as limitações dos sistemas existentes, fornecendo uma solução mais automatizada e amigável para monitoramento de atividades humanas em casa.

## 2.2 CSI Learning

Devido à elevada proporção de ruído, a coleta bruta de *Channel State Information* pode não ser suficientemente representativa para diversas atividades humanas. Uma prática comum consiste na extração manual de características discriminativas. Contudo, essa técnica manual demanda conhecimento extremamente especializado, e a garantia de generalização não é assegurada, uma vez que a extração de características e a etapa de reconhecimento não são executadas de maneira conjunta. Uma solução para esse problema é o uso de *machine learning* para fazer a classificação desses sinais. (Ma; Zhou; Wang, 2019)

O trabalho de (Muaaz *et al.*, 2022) destaca-se ao abordar a dependência ambiental observada em muitos sistemas de reconhecimento de atividades humanas baseados em Wi-Fi. Ao integrar uma CNN com informações de CSI ambientalmente independentes, o autor utiliza o *CSI ratio method* ("método da razão de CSI") e o PCA ("análise de componentes principais") para mitigar o impacto do ruído e da variação ambiental. Resultados como uma acurácia geral de treinamento de 97,78%, posicionam o *Wi-Sense* como uma solução promissora, especialmente ao considerar sua integração potencial em sistemas de informação de saúde.

Abordagens anteriores são também notáveis e demonstram passos iniciais para o uso desse tipo de tecnologia, especificamente uma (ABLSTM). (Chen *et al.*, 2019) Reconhecendo a complexa relação entre atividades humanas e medições de WiFi CSI, a ABLSTM destaca-se ao aprender características representativas bidirecionais a partir de medições sequenciais brutas de CSI. A introdução de um mecanismo de atenção, que atribui diferentes pesos a essas características aprendidas, melhora significativamente o desempenho de reconhecimento para todas as atividades avaliadas.

O *Reliable Wi-Fi Sensing Through Few-Shot Multi-Antenna Multi-Receiver CSI Learning* (ReWiS — "Sensoriamento Wi-Fi Confiável por Meio de Aprendizado de CSI com Múltiplas Antenas e Múltiplos Receptores com Pouco Treinamento", tradução livre) representa um marco no campo, propondo uma estrutura robusta e independente de ambiente para o *Wi-Fi Sensing*. Sua inovação central reside na utilização de *Few-Shot Learning* (FSL) como motor de inferência, reduzindo a necessidade de extensa coleta de dados e extração específica de características CSI. A capacidade de generalização rápida para novos ambientes, juntamente com a exploração de diversidade multi-antena e multi-receptor, demonstra avanços significativos em relação a abordagens anteriores. Além disso, o ReWiS apresentou uma queda de menos de 10% na acurácia ao ser testado em ambientes diferentes. (Bahadori;

Ashdown; Restuccia, 2022)

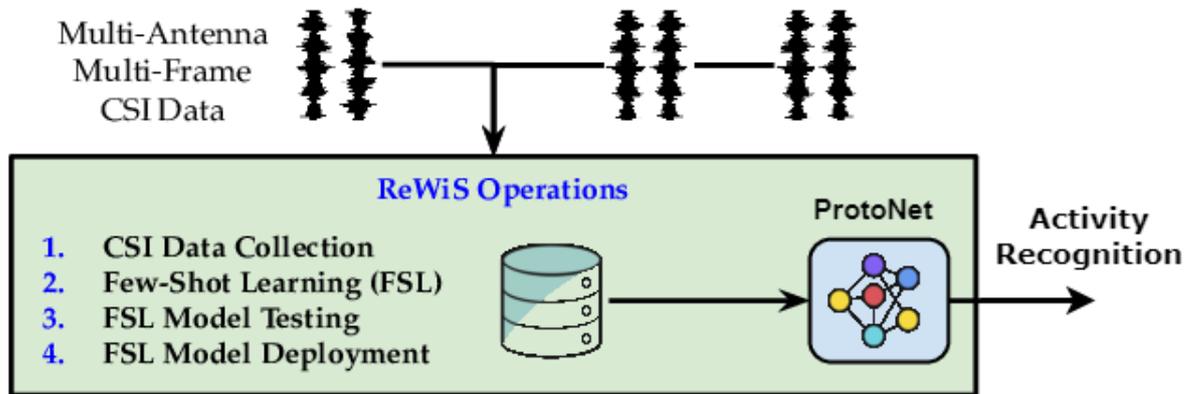


Figura 2.5 – Visão geral do ReWiS em alto nível. (Bahadori; Ashdown; Restuccia, 2022)

Em síntese, o campo de reconhecimento de atividades humanas por meio de Wi-Fi CSI está evoluindo rapidamente, com o ReWiS liderando como estado da arte, seguido por abordagens avançadas de *deep learning*, como ABLSTM, e soluções inovadoras, como o Wi-Sense, que buscam superar desafios ambientais para aplicações mais amplas. Esses avanços reforçam a viabilidade e o potencial transformador dessas tecnologias em uma variedade de contextos.

## 2.3 Few-Shot HAR

Diante da dificuldade em obter um grande volume de dados em ambientes prediais, o *Few-Shot Learning* é uma abordagem de aprendizado de máquina projetada para treinar modelos com um número limitado de exemplos de treinamento, ou seja, quando há apenas algumas amostras disponíveis para cada classe ou categoria. Em muitos cenários do mundo real, é difícil ou impraticável coletar grandes conjuntos de dados rotulados para treinar modelos de aprendizado de máquina tradicionais. O *Few-Shot Learning* busca superar essa limitação, permitindo que os modelos aprendam com poucos exemplos de treinamento. (Zhang *et al.*, 2022)

Existem diferentes técnicas e abordagens dentro do *Few-Shot Learning*, sendo uma das mais comuns o *Transfer Learning*, que é um modelo pré-treinado em um conjunto de dados grande e genérico é ajustado para o novo conjunto de dados com poucos exemplos de treinamento disponíveis. O conhecimento adquirido durante o treinamento prévio é transferido para a tarefa de *Few-Shot Learning*, permitindo que o modelo se adapte mais rapidamente aos novos dados. Há também o *Meta-Learning*, que envolve treinar um modelo para aprender a aprender. Em vez de apenas aprender a partir dos exemplos de treinamento, o modelo é treinado em uma variedade de tarefas de aprendizado e deve ser capaz de generalizar rapidamente para novas tarefas com poucos exemplos de treinamento. O modelo

é treinado para extrair características relevantes e aprender estratégias de generalização que podem ser aplicadas a novas tarefas com eficácia. (Zhang *et al.*, 2022)

O *Few-Shot Learning* em HAR é especialmente útil em cenários onde as atividades são altamente variáveis e podem mudar com o tempo. Por exemplo, em um ambiente de assistência médica domiciliar, pode ser necessário reconhecer novas atividades que não foram vistas anteriormente com apenas alguns exemplos. Nesses casos, o *few-shot learning* permite que os modelos se adaptem rapidamente às mudanças nas atividades humanas, garantindo uma detecção precisa e eficaz das atividades em tempo real. (Bauchspiess *et al.*, 2024)

A Figura 2.6 mostra a metodologia prototípica *Few-Shot Learning*, que usa modelos de aprendizado de máquina para adaptar protótipos a novos ambientes com poucos exemplos de treinamento. Consultas são classificadas com base na distância aos protótipos disponíveis, e a adaptação pode resultar em classificações diferentes para diferentes ambientes.

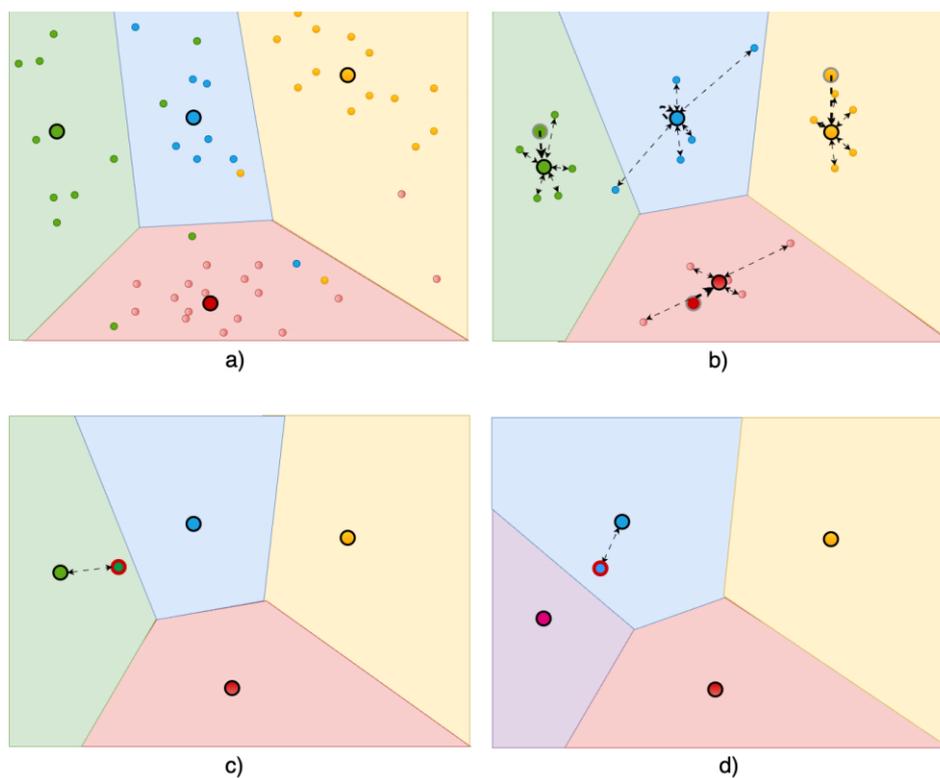


Figura 2.6 – Adaptação de Domínio Prototípica com Poucos Exemplos (*Few-Shot Prototypical Domain Adaptation*) a) Aprender características com um grande conjunto de treinamento no ambiente base E1 (contendo 4 protótipos: vermelho, verde, azul e amarelo). b) Adaptação com poucos exemplos (*Few-Shot*) para o novo ambiente E2, com 5 exemplos e 4 classes. Classificação por distância euclidiana. c) Classificação euclidiana dos dados de consulta no ambiente E2. d) O novo ambiente E3 leva a protótipos adaptados (nota: pode alterar a classificação). (Bauchspiess *et al.*, 2024)

## 3 Materiais e Métodos

O atual capítulo discorre sobre equipamentos e a metodologia utilizados na execução do projeto, detalhando as etapas experimentais e computacionais realizadas neste trabalho. A seguir, será descrito o processo de escolha do ESP32, bem como o *framework* onde serão implementados os algoritmos de coleta de dados CSI, as estratégias de pré-processamento e a implementação da rede neural convolucional (CNN) aplicada à tarefa de HAR.

### 3.1 ESP32

A obtenção de sinais CSI é realizada mediante uma rede de microcontroladores ESP32, empregando essa funcionalidade com a finalidade de instrumentar o *Wi-Fi Sensing*.

O ESP32 é um microcontrolador de alto desempenho amplamente utilizado na implementação de projetos IoT. Desenvolvido pela *Espressif Systems*, o ESP32 se destaca por sua versatilidade e capacidade de integração de comunicação sem fio, suportando padrões como Wi-Fi e *Bluetooth*. Com um processador *dual-core Xtensa 32-bit*, o ESP32 oferece uma capacidade computacional adequada para uma variedade de aplicações, desde automação residencial até monitoramento remoto. Além disso, sua arquitetura modular e recursos avançados, como sensores integrados, tornam o ESP32 uma escolha popular para desenvolvedores e engenheiros envolvidos na criação de soluções IoT.

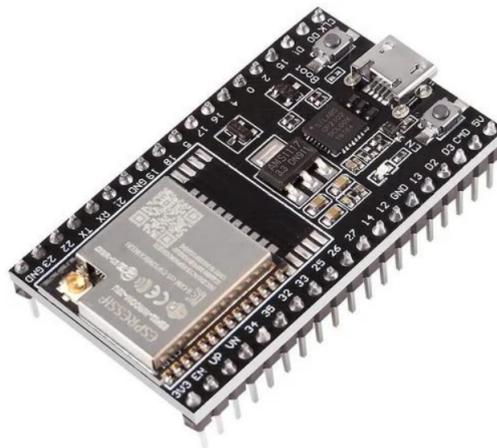


Figura 3.7 – ESP32-WROOM-32UE

Para esse projeto, foi escolhida a versão mostrada na Figura 3.7, o módulo ESP32-WROOM-32UE, pois ele possui um conector do tipo coaxial, onde pode ser conectada uma antena externa, o que é essencial para a qualidade do sinal CSI adquirido.

### 3.1.1 ESP-IDF

O *Espressif IoT Development Framework* (ESP-IDF), é um conjunto de ferramentas de desenvolvimento de software criado pela *Espressif Systems* para programar microcontroladores da série ESP32 e ESP8266. Estes microcontroladores são amplamente utilizados em projetos de IoT devido à sua versatilidade, eficiência energética e conectividade WiFi e *Bluetooth* integrada.

O ESP-IDF é uma estrutura de código aberto e é mantido pela *Espressif Systems*. Ele fornece uma coleção abrangente de bibliotecas, *drivers* de *hardware* e ferramentas de desenvolvimento que facilitam a criação de aplicativos para dispositivos IoT baseados em ESP32 e ESP8266.

Algumas características principais do ESP-IDF incluem:

- *Software Development Kit* (SDK): O ESP-IDF inclui um SDK completo que contém bibliotecas e ferramentas necessárias para desenvolver aplicativos para os microcontroladores ESP32 e ESP8266.
- *Drivers de Hardware*: O *framework* inclui uma vasta coleção de drivers de hardware que simplificam a interação com periféricos como GPIOs, UART, I2C, SPI, ADC, PWM, entre outros.
- Componentes: Os componentes são módulos de software reutilizáveis que podem ser usados para construir aplicativos mais complexos. Eles facilitam a organização e a manutenção do código, permitindo que os desenvolvedores dividam seu projeto em partes menores e mais gerenciáveis.
- Ferramentas de Compilação e Depuração: O ESP-IDF inclui ferramentas de linha de comando para compilar, *flash* e depurar aplicativos. Isso permite que os desenvolvedores escrevam, compilem e carreguem código facilmente nos dispositivos ESP32 e ESP8266.
- Documentação Abundante: A documentação do ESP-IDF é extensa e bem mantida, oferecendo guias detalhados, exemplos de código e referências para ajudar os desenvolvedores a aproveitar ao máximo o *framework*.

Nos testes iniciais de coleta de dados, a versão 5.2 do ESP-IDF foi utilizada, operando no Windows OS.

### 3.1.2 ESP32 ESP-NOW

O ESP-Now é um protocolo de conexão desenvolvido pela Espressif, que possibilita a comunicação direta entre dispositivos Wi-Fi sem a necessidade de estabelecer uma conexão

padrão. Ele faz uso das camadas PHY e MAC do padrão 802.11n, simplificando a configuração em comparação com a conexão WiFi padrão AP-STA, que requer verificação de usuário e senha a cada vez que a conexão é feita. Com o ESP-NOW, a conexão está pronta logo após ligar o dispositivo, tornando-o a escolha ideal para os experimentos.

Esse protocolo opera com uma taxa de dados baixa, o que garante alta eficiência, sendo ideal para dispositivos IoT alimentados por bateria e redes de sensores. Para garantir a segurança, os quadros de ação do ESP-NOW são protegidos pelo protocolo CTR com CBC-MAC (CCMP). Isso faz do ESP-NOW uma escolha popular em aplicações como iluminação inteligente, controle remoto e diversas tarefas relacionadas a sensores.

Usando o ESP-NOW, vários dispositivos podem se conectar sem a necessidade de Wi-Fi. Porém, muitos dispositivos sem fio utilizam a conexão sem fio de baixa potência na frequência de 2.4 GHz. Como resultado, é necessário realizar um emparelhamento antes de os dispositivos se comunicarem.

Após o emparelhamento, a conexão entre dois dispositivos é segura e *peer-to-peer* (ponto a ponto), sem necessidade de *handshake*, sendo uma conexão robusta. Ou seja, se uma das placas sofrer uma queda de energia inesperada ou reiniciar, ela imediatamente restabelece a conexão com seu par e continua a comunicação.

Os dados podem ser transmitidos de forma unidirecional ou bidirecional. Não há necessidade de um roteador ou fonte Wi-Fi externa, e os dados podem ser criptografados ou não. Dependendo da aplicação, é possível ter de 2 a 20 pontos interagindo entre si, através de um conexão via *broadcast* (Figura 3.8). O ambiente ao redor pode impactar significativamente o alcance, mas, em condições adequadas (e com antenas apropriadas), é possível alcançar mais de 400 metros de distância. (Espressif, 2024)

### 3.1.3 Aquisição de Dados de CSI com o ESP32

A aquisição de *CSI data* foi feita através dos códigos ESP32 *csi261.c*, o fluxograma deste código está representado na Figura 3.9, e consiste em:

- M1 inicia o processo e coleta os sinais CSI
- M2 dispara logo após receber o *broadcast de M1*.
- Um Acesso Múltiplo por Divisão de Tempo (*Time Division Multiple Access*) dá a cada sensor uma janela preferencial para transferir seus dados para M1.

Em dispositivos de telecomunicação, a integridade dos dados é mais importante que eventuais atrasos, e como neste trabalho o ESP32 é utilizado como um dispositivo *sensing*, isso pode ser um problema, pois para essa aplicação a sincronização das aquisições em multi-canais é mais importante (e crucial para o reconhecimento correto de atividades). Os

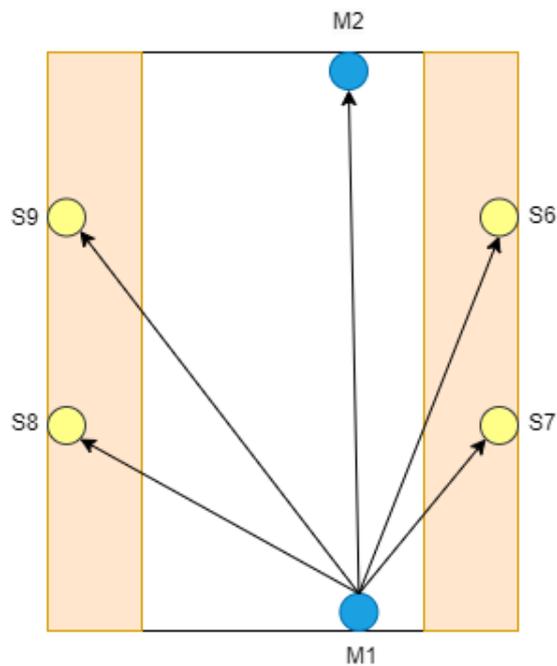


Figura 3.8 – Vista superior da sala de estudos do LARA, em que está representada a topologia da rede de sensoriamento. M1 e M2 são os mestres, que recebem os pacotes dos sensores S1, S2, S3, S4. Os pacotes são enviados em ciclos, que começam com o "comando" do M1, enviado via transmissão (*broadcast*) para todos os sensores. A conexão utiliza o protocolo ESP-NOW.

módulos ESP-32 não esperam o ACK para sinais de *broadcast*. Na camada MAC, os eventos ocorrem com diferentes prioridades e habilitam tarefas correspondentes. Cada tarefa, quando ativa, tenta processar dados antigos, reenviando os dados várias vezes. Assim, a aquisição de CSI só é habilitada por uma janela curta. Portanto, alguns dados também são perdidos. Como é possível ver na Figura 3.10, o diagrama mostra as transmissões esperadas do TDMA (elipses em azul), porém nas elipses vermelhos os "outros" módulos ainda transmitem pacotes, por dois ciclos, por isso se faz necessário o procedimento representado na Figura 3.9.

## 3.2 Python

Os códigos desenvolvidos para a aquisição e tratamento de dados CSI e para a implementação de algoritmos de *machine learning* foram criados utilizando a linguagem de programação Python. Essa escolha linguística proporcionou uma flexibilidade significativa no desenvolvimento, permitindo uma integração eficiente com bibliotecas especializadas para processamento de sinais, aprendizado de máquina e análise de dados. A utilização do Python facilita a implementação de rotinas robustas para o processamento de dados CSI, assim como a criação e treinamento de modelos de *machine learning*, contribuindo para uma abordagem coesa e eficaz na condução das tarefas propostas no âmbito do projeto.

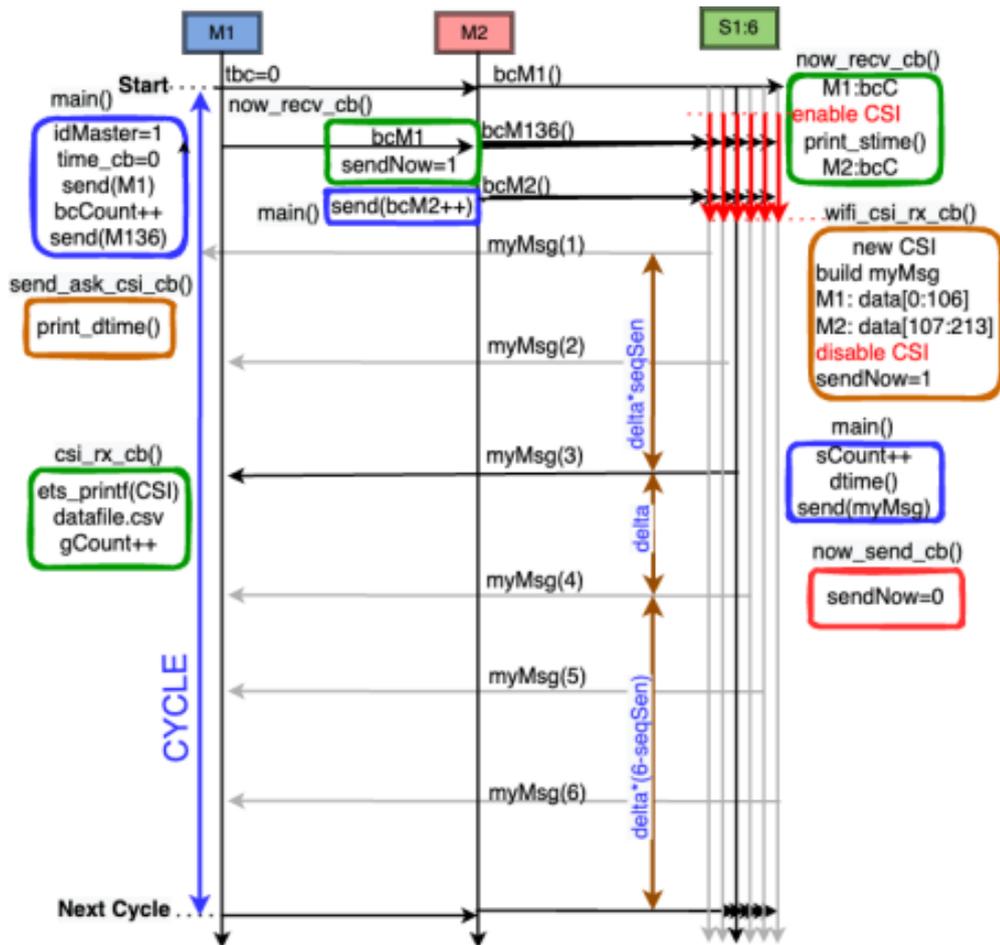


Figura 3.9 – Fluxograma do código de aquisição de CSI. (Bauchspiess *et al.*, 2024)

### 3.3 Metodologia

O fluxo de atividades desse trabalho consiste, resumidamente, em adquirir dados de CSI no ambiente da sala de estudos do LARA, treinar uma rede neural (CNN) com esses dados para fazer a classificação das atividades humanas, e validar o modelo.

Iniciando o processo, propõe-se o levantamento de um Banco de Dados para o treinamento e validação, mediante a utilização de módulos ESP32 estrategicamente distribuídos na sala de estudos do Laboratório de Automação e Robótica (LARA), para criar um ambiente de automação predial baseado em HAR análogo ao mostrado na Figura 1.1.

O emprego dessas atividades requer a utilização de seis módulos ESP32, equipados com antenas, subdivididos em duas categorias de mestres (M1 e M2) e quatro sensores. O M1, atuando como mestre inicial, inicia o ciclo ao realizar uma transmissão *broadcast* para todos os módulos, dando início à sequência de ações. Após a recepção do sinal inaugural pelo M1, o mestre 2 (M2) entra em operação, emitindo um sinal de *broadcast*. Posteriormente, após a obtenção dos sinais provenientes dos mestres, os sensores transmitem ao M1 as informações detalhadas de CSI coletadas de cada sinal. Essa transferência de dados é executada de forma sequencial, um sensor por vez, a fim de garantir a preservação integral das informações sem

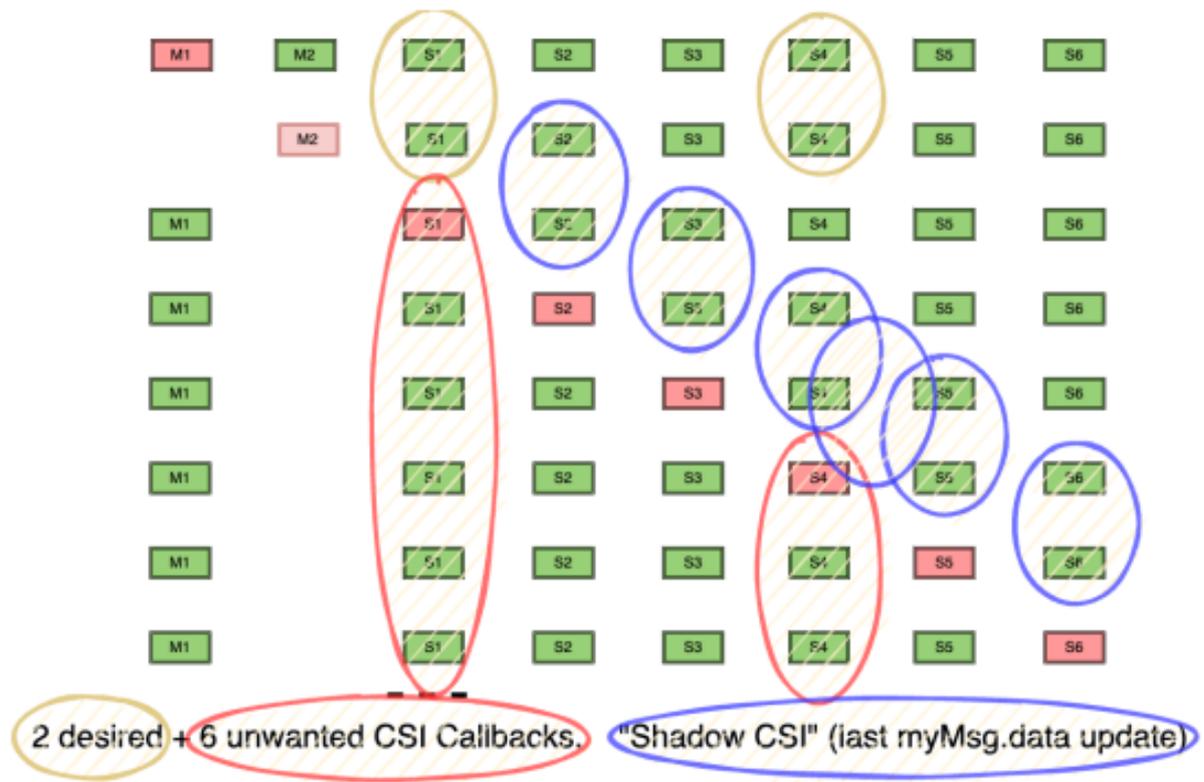


Figura 3.10 – Diagrama no tempo de aquisição de CSI com perdas de dados, devido à colisões múltiplas.

qualquer perda de dados.

Para testes iniciais, foram utilizados quatro ESPs iguais ao mostrado na Figura 3.7, sendo o mestre 1, mestre 2 e 2 sensores, gerando 4 *links*. A configuração dos *links* foi análoga à configuração mostrada na Figura 3.11.

Em um primeiro momento, os dados de CSI mostrados apresentavam erros de identificação dos sensores de origem, foi observado que esses problemas estavam relacionados às partes nos códigos *appmain.c* carregados nos dispositivos ESP32, especificamente nas partes relacionadas à identificação dos endereços MAC de cada sensor. Como solução, foram atualizadas as linhas de códigos com os endereços dos ESPs disponíveis, como é mostrado no trecho de código abaixo.

Código 3.1 – Função de identificação dos sensores pelos seus endereços MAC.

```

1 uint8_t na4MAC(uint8_t *mac) {
2     uint8_t name;
3     uint8_t mac4=mac[4], mac5=mac[5];
4
5     if (myMAC[5]==0x58) myMsg.idSens=1;           // M1
6     else if (myMAC[5]==0xcc) myMsg.idSens=2;     // M2
7
8     else if (myMAC[5]==0xf8) myMsg.idSens=6;     // S6

```

```

9     else if (myMAC [5]==0xb8) myMsg.idSens=7;      // S7
10    else if (myMAC [5]==0x6c) myMsg.idSens=8;      // S8
11    else if (myMAC [5]==0x34) myMsg.idSens=9;      // S9
12
13    else name=20;
14    return name;
15 }

```

A aquisição de dados foi feita dentro de um quarto, em rotinas de aproximadamente 10 minutos, onde uma pessoa ficou alternando a cada 10 segundos entre as seguintes atividades: sentado, caminhando, em pé e sala vazia.

### 3.3.1 Rotinas de Experimentos no Ambiente do LARA

No ambiente da sala de estudos do Laboratório de Automação e Robótica Avançada (LARA), foram configurados dois mestres (M1 e M2) e quatro sensores, de acordo com a topologia mostrada na Figura 3.11. A Figura 3.12 mostra um esboço da região de primeira zona de Fresnel nessa disposição, a escolha desses pontos de monitoramento foi estratégica para a automação predial, visando concentrar as Zonas de Fresnel em áreas de alto tráfego ou onde as interações humanas são frequentes, o que é essencial para a efetividade do HAR. (Chen *et al.*, 2019)

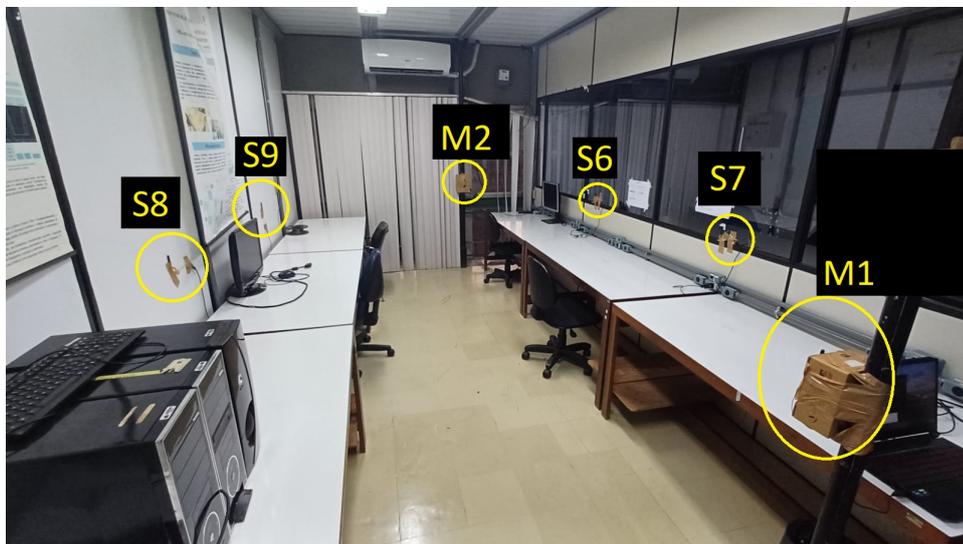


Figura 3.11 – Fotografia que mostra a disposição dos sensores na sala de estudos do LARA.

A estrutura deste experimento foi inspirado em abordagens semelhantes vistas na literatura de *Wi-Fi Sensing*, que frequentemente utilizam rotinas curtas e bem distribuídas para garantir dados suficientes e confiáveis para o treinamento de modelos de reconhecimento de atividades. (Bauchspiess *et al.*, 2024)

Para os experimentos realizados no LARA, foram implementadas adaptações nas rotinas experimentais visando uma coleta de dados mais prática e eficiente, garantindo uma

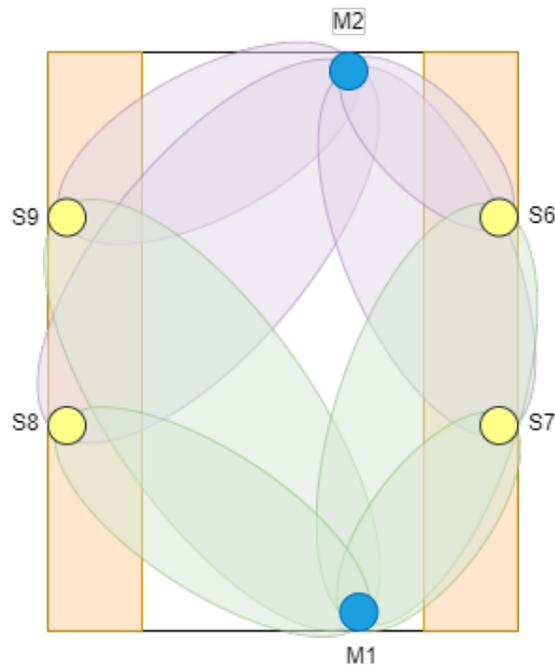


Figura 3.12 – Esboço da região de 1ª zona de Fresnel e a topologia de sensores na sala de estudos do LARA (Vista superior).

fluidez melhor na transição entre as atividades humanas. As rotinas foram configuradas para uma duração total de 4 minutos, o que permitiu reduzir o tempo do processo de coleta, facilitando a verificação de erros nos dados coletados. A Figura 3.13 mostra 4 imagens capturadas do vídeo de *Ground Truth* dos experimentos realizados no LARA, e a Figura 3.14 mostra uma representação da posição e orientação em que as atividades foram executadas.

A divisão temporal das atividades foi organizada da seguinte forma:

- 60s Vazio;
- 30s Caminhando;
- 60s Sentado (parado);
- 60s Em pé (parado);
- 30s Caminhando.

Com essa divisão, em uma rotina de experimento são obtidos 6 quadros de 10 segundos para cada uma das quatro atividades.

Na Figura 3.15 é mostrada uma representação temporal das amplitudes dados de CSI para algumas subportadoras do sensor 6. Observando as transições entre atividades, destacadas em vermelho, nota-se a dificuldade de classificar as atividades apenas pela

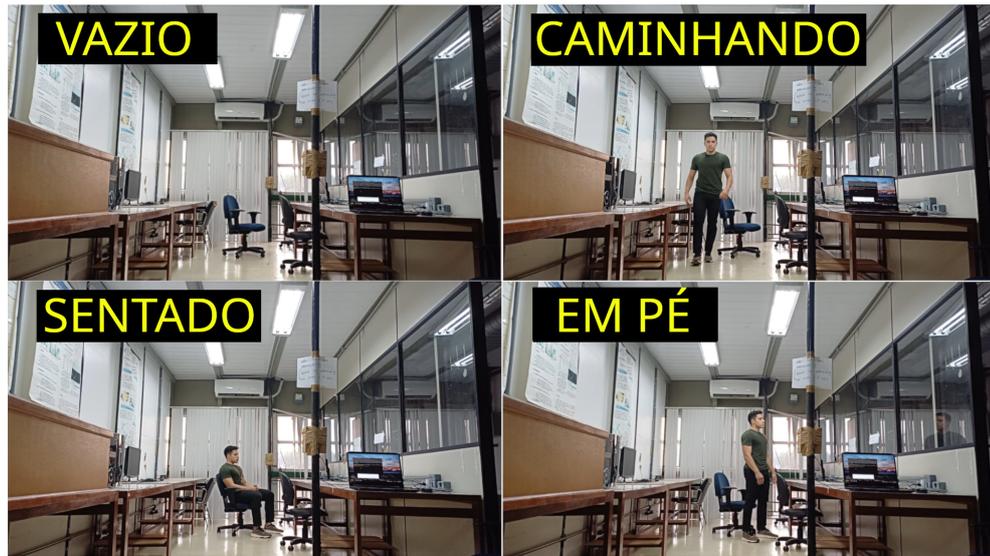


Figura 3.13 – Imagens capturadas do vídeo de *Ground Truth* dos experimentos na sala de estudos do LARA. Atividades de sala vazia, caminhando, sentado, em pé.

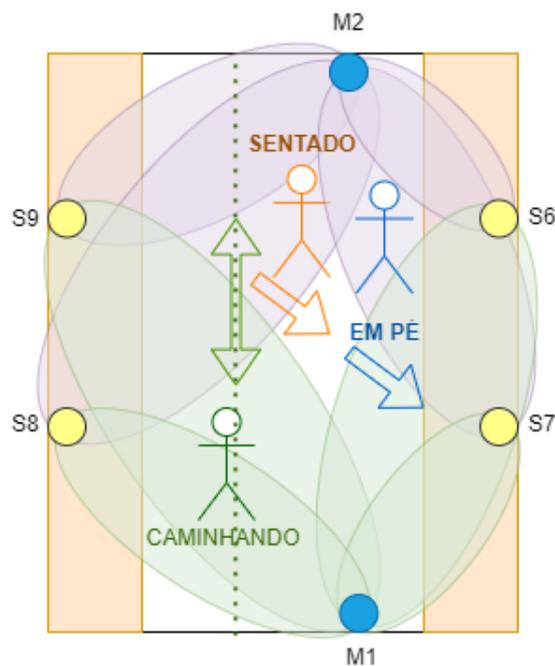


Figura 3.14 – Representação da posição e orientação em que as atividades foram executadas na sala de estudos do LARA, durante os experimentos. (Vista superior).

observação da variação no CSI. É nesse cenário que o uso de redes neurais se torna essencial para realizar a classificação de forma eficaz.

Os dados de CSI foram salvos em arquivos .csv, cujas linhas contém dados da aquisição de um sensor dentro de um *broadcast*, como é mostrado na Figura 3.16. Nesta figura, apenas o início do dados de CSI são mostrados mais à direita, e a primeira linha do arquivo é um *header* contendo os *lables* de cada coluna, onde pode-se destacar: CSI\_DATA (identificação para iniciar nova linha do arquivo .csv), idS (identificação do sensor), sC (ciclo em que esse

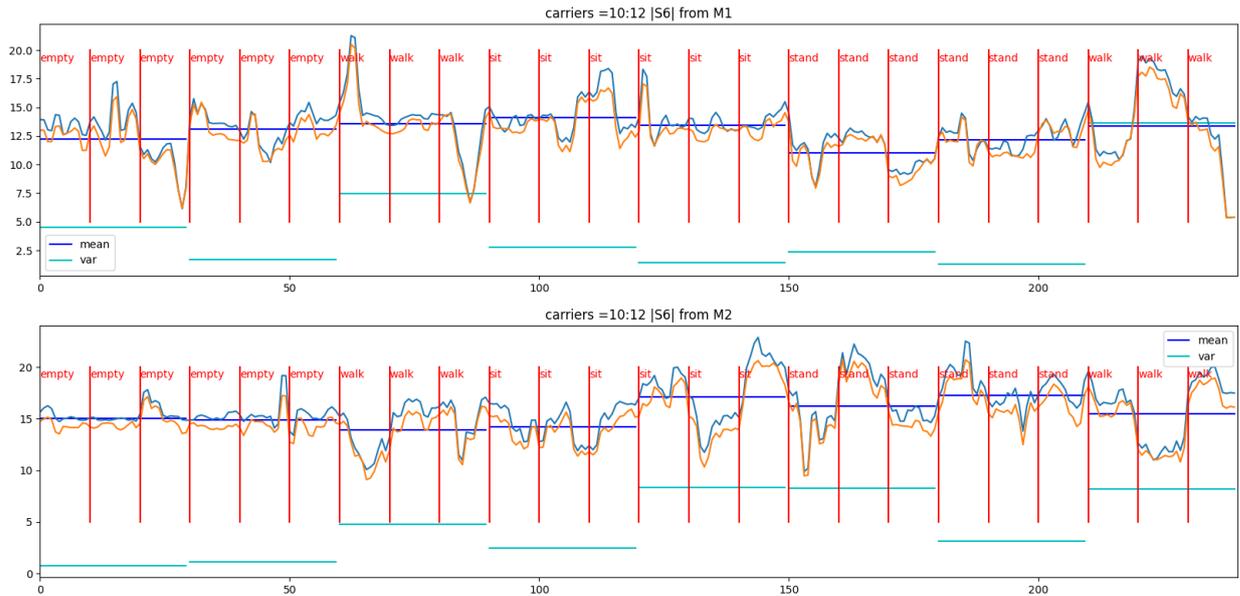


Figura 3.15 – Representação temporal da amplitude dos dados de CSI para algumas portadoras do sensor 6. Onde a cada janela marcada entre os traços vermelhos se passam 10 s.

dado foi adquirido),  $t_M(\text{ms})$  (tempo decorrido desde o início das aquisições),  $t_{in\_cycle}$  (tempo decorrido desde o início do ciclo atual).

### 3.3.2 Pré-processamento dos Dados

No contexto deste projeto, os dados de CSI são inicialmente coletados em seu estado bruto, contendo possíveis inconsistências. Assim, uma etapa de pré-processamento de dados se faz necessária. Esta consiste em limpeza de dados, seguida de uma transformação Análise de Componentes Principais (PCA) para reduzir a dimensionalidade, facilitando a conversão dos dados em imagens que serão utilizadas para o treinamento da rede neural.

#### 3.3.2.1 Limpeza de Dados

Durante a coleta de dados de CSI para atividades humanas, é comum haver linhas de dados faltantes ou repetidas, o que compromete a sequência de tempo e as leituras dos sensores. Para resolver esse problema, foi desenvolvido um algoritmo que ajusta as linhas dos dados de CSI com o objetivo de manter a sequência correta dos sensores (6, 7, 8, 9). Esse algoritmo deleta linhas redundantes, e utiliza uma tolerância temporal para inserir linhas de um sensor em sequências que ele esteja faltando.

Para lidar com o ruído presente nos dados, é utilizado um filtro *Butterworth* passa-baixa. O filtro é aplicado. Esse tipo de filtro permite que componentes de baixa frequência do sinal passem, enquanto atenua ou bloqueia as componentes de alta frequência. A configuração do filtro *Butterworth* é a seguinte:

- $f_s = 10\text{Hz}$ ; (frequência de amostragem)

CSI_DATA: gC	idS	sC	sC_M	bc1	bc2	bcM1	retry	RSSI1	RSSI2	tCSI1_S	tCSI2_S	tdma_S	t_M(ms)	dt_rxM	tin_cycle	[data]
CSI_DATA: 3	6	2	2	3	2	3	0	-28	-20	2091.818	2096.1	15.636	1503.821	1290.2	23.817	[13,-6,13,-6,
CSI_DATA: 4	7	2	2	3	2	3	0	-52	-43	2090.754	2094.946	107.379	1588.753	1375.132	108.749	[-9,13,-9,13,
CSI_DATA: 5	8	2	2	3	2	3	0	-36	-41	2094.204	2098.396	211.269	1692.704	1479.83	212.7	[0,11,0,11,1,
CSI_DATA: 6	9	2	2	3	2	3	0	-36	-34	2095.132	2099.325	308.99	1793.525	1579.904	313.521	[-16,-5,-17,-1,
CSI_DATA: 7	6	3	3	4	3	4	0	-28	-20	2830.248	2839.849	19.609	2243.384	449.859	23.396	[-4,-16,-4,-1,
CSI_DATA: 8	7	3	3	4	3	4	0	-51	-43	2829.182	2838.784	117.464	2341.767	548.242	121.779	[6,-10,6,-10,
CSI_DATA: 9	8	3	3	4	3	4	0	-36	-41	2832.635	2842.237	211.406	2432.725	639.2	212.737	[-4,10,-4,10,
CSI_DATA: 10	9	3	3	4	3	4	0	-37	-33	2833.562	2843.164	318.252	2539.558	746.33	319.57	[-6,-14,-6,-1,
CSI_DATA: 7	6	3	3	4	3	4	0	-28	-20	2830.248	2839.849	19.609	2243.384	449.859	23.396	[-4,-16,-4,-1,
CSI_DATA: 11	7	4	4	5	4	5	0	-51	-43	3569.169	3577.501	117.555	3078.78	539.222	118.795	[6,7,6,7,6,5,
CSI_DATA: 12	8	4	4	5	4	5	0	-36	-40	3572.626	3580.957	211.521	3173.356	633.798	213.371	[-8,-9,-9,-9,-4,
CSI_DATA: 13	9	4	4	5	4	5	0	-37	-33	3573.552	3581.883	318.356	3279.495	739.937	319.51	[-3,16,-4,16,
CSI_DATA: 14	6	4	4	6	4	7	0	-28	-20	5056.11	5065.619	25.79	4475.573	1196.78	35.569	[-6,-14,-6,-1,
CSI_DATA: 15	7	5	5	7	5	7	0	-51	-43	5054.94	5064.548	117.194	4558.747	1279.252	118.743	[-3,10,-2,9,-1,
CSI_DATA: 16	8	5	5	7	5	7	0	-35	-40	5058.402	5068.1	221.127	4662.691	1383.196	222.687	[3,-12,3,-13,
CSI_DATA: 17	9	5	5	7	5	7	0	-36	-33	5059.326	5068.934	317.961	4776.227	1496.732	336.223	[0,16,1,16,2,
CSI_DATA: 14	6	4	4	6	4	7	0	-28	-20	5056.11	5065.619	25.79	4475.573	1196.78	35.569	[-6,-14,-6,-1,
CSI_DATA: 15	7	5	5	7	5	7	0	-51	-43	5054.94	5064.548	117.194	4558.747	1279.252	118.743	[-3,10,-2,9,-1,
CSI_DATA: 18	8	6	6	8	6	8	0	-36	-40	5792.624	5797.729	209.231	5397.151	620.924	217.163	[5,13,5,15,5,
CSI_DATA: 19	9	6	6	8	6	8	0	-37	-33	5793.547	5798.652	306.9	5489.973	713.746	309.985	[13,4,13,5,1,
CSI_DATA: 14	6	4	4	6	4	7	0	-28	-20	5056.11	5065.619	25.79	4475.573	1196.78	35.569	[-6,-14,-6,-1,
CSI_DATA: 15	7	5	5	7	5	7	0	-51	-43	5054.94	5064.548	117.194	4558.747	1279.252	118.743	[-3,10,-2,9,-1,
CSI_DATA: 18	8	6	6	8	6	8	0	-36	-40	5792.624	5797.729	209.231	5397.151	620.924	217.163	[5,13,5,15,5,
CSI_DATA: 20	9	7	7	12	7	12	0	-37	-33	8755.539	8763.638	317.41	8459.567	2969.594	319.563	[-7,13,-7,12,
CSI_DATA: 14	6	4	4	6	4	7	0	-28	-20	5056.11	5065.619	25.79	4475.573	1196.78	35.569	[-6,-14,-6,-1,
CSI_DATA: 15	7	5	5	7	5	7	0	-51	-43	5054.94	5064.548	117.194	4558.747	1279.252	118.743	[-3,10,-2,9,-1,
CSI_DATA: 21	8	8	8	13	8	13	0	-35	-40	9498.19	9507.754	219.666	9102.762	643.195	222.777	[10,3,11,3,1,
CSI_DATA: 20	9	7	7	12	7	12	0	-37	-33	8755.539	8763.638	317.41	8459.567	2969.594	319.563	[-7,13,-7,12,
CSI_DATA: 22	6	6	6	12	6	16	0	-28	-20	11711.106	11717.349	6.459	11118.417	2015.655	18.413	[7,-13,6,-12,
CSI_DATA: 23	7	9	9	16	9	16	0	-52	-43	11710.21	11716.264	96.837	11214.65	2111.303	114.61	[-12,11,-10,9,
CSI_DATA: 24	8	9	9	16	9	16	0	-35	-41	11713.508	11719.751	200.776	11318.177	2215.415	218.173	[-2,-11,-3,-1,
CSI_DATA: 25	9	9	9	16	9	16	0	-37	-34	11714.422	11720.665	297.621	11414.403	2311.641	314.399	[7,14,7,13,9,
CSI_DATA: 26	6	7	7	13	7	17	0	-28	-19	12451.677	12455.307	12.574	11856.649	442.246	16.663	[4,13,4,13,5,
CSI_DATA: 27	7	10	10	17	10	17	0	-51	-43	12450.59	12454.221	104.986	11957.257	542.854	117.271	[-14,-3,-14,-4,
CSI_DATA: 28	8	10	10	17	10	17	0	-36	-41	12454.8	12457.711	208.914	12053.214	638.811	213.228	[-5,7,-5,8,-4,
CSI_DATA: 29	9	10	10	17	10	17	0	-36	-34	12454.992	12458.623	305.763	12150.113	735.71	310.127	[3,-16,2,-16,

Figura 3.16 – Arquivo .csv contendo dados de CSI, mostrado apenas em parte na última coluna à direita, e informações adicionais. Em destaque, CSI\_DATA (identificação para iniciar nova linha do arquivo .csv), idS (identificação do sensor), sC (ciclo em que esse dado foi adquirido), t\_M(ms) (tempo decorrido desde o início das aquisições), tin\_cycle (tempo decorrido desde o início do ciclo atual).

- $cutoff = 2Hz$ ; (frequência de corte)
- $nyq = 0.5 * f_s$  (frequência de Nyquist);
- $Order = 2$  (Ordem do filtro);

### Código 3.2 – Código do filtro Butterworth passa-baixa.

```

1 def butter_lowpass_filter(data, cutoff, fs, order):
2     normal_cutoff = cutoff / nyq
3     # Get the filter coefficients
4     b, a = butter(order, normal_cutoff, btype='low', analog=False)
5     y = filtfilt(b, a, data)
6     return y

```

Como é mostrado no Código 3.3.2.1, o filtro *Butterworth* passa-baixa é aplicado entre as amostras das subportadoras. O código itera sobre as amostras de amplitude ou fase ( $ab[i]$ ), e o filtro é aplicado individualmente a cada linha da matriz, que representa o CSI de cada subportadora. Assim, ele filtra as componentes de frequência mais altas para reduzir o ruído em cada uma dessas amostras, garantindo que apenas as frequências abaixo da frequência de corte sejam preservadas. O filtro não está atuando no domínio do tempo diretamente, mas sim sobre as amostras de CSI correspondentes às subportadoras.

Código 3.3 – Código do filtro Butterworth passa-baixa.

```

1 if wFilter:
2     for i in range(rang):
3         ab[i] = butter_lowpass_filter(ab[i], cutoff, fs, 3)

```

Além disso, para garantir que os valores de fase permaneçam dentro de uma faixa específica, realiza-se uma *phase sanitization*, onde valores fora do intervalo são ajustados pela adição ou subtração de  $2\pi$ .

Código 3.4 – Código para *phase sanitization*.

```

1 if oAbs:
2     ab=np.zeros((li,co)) # only Amp
3     for i in range(li):
4         for c in range(co):
5             ab[i][c]=abs(complex(imgI[i,c],imgR[i,c]))
6 else:
7     ab=np.zeros((2*li,co)) # both: Amp & Phase
8     for i in range(li):
9         for c in range(co):
10            ab[i][c]=abs(complex(imgI[i,c],imgR[i,c]))
11            ab[i+li][c]=cmath.phase(complex(imgI[i,c],imgR[i,c]))

```

Essa abordagem garante que o conjunto de dados final esteja devidamente padronizado e pronto para as etapas subsequentes de pré-processamento e treinamento da rede neural.

### 3.3.2.2 Geração de Imagens PNG após Transformação PCA

Após a limpeza dos dados, os sinais CSI são transformados em imagens utilizando a técnica de PCA para redução da dimensionalidade. Essa etapa é essencial para representar visualmente os dados de CSI, facilitando o treinamento do modelo de rede neural convolucional (CNN).

Cada atividade monitorada (sentado, em pé, caminhando e vazio) é convertida em uma série de imagens PNG. Para cada rotina de atividade, são geradas 6 imagens, representando janelas de 10 segundos cada, como é exemplificado na Figura 3.17. Essas imagens são então organizadas em pastas separadas por atividade, de modo que cada pasta contenha as imagens correspondentes ao seu respectivo estado. Essa organização por pastas garante que o modelo CNN receba os dados devidamente categorizados por atividade, facilitando o processo de classificação.

Assim, o banco de dados foi criado, com 25 rotinas de experimentos de 4 minutos. E foi dividido na proporção de 80% para treinamento e 20% para validação

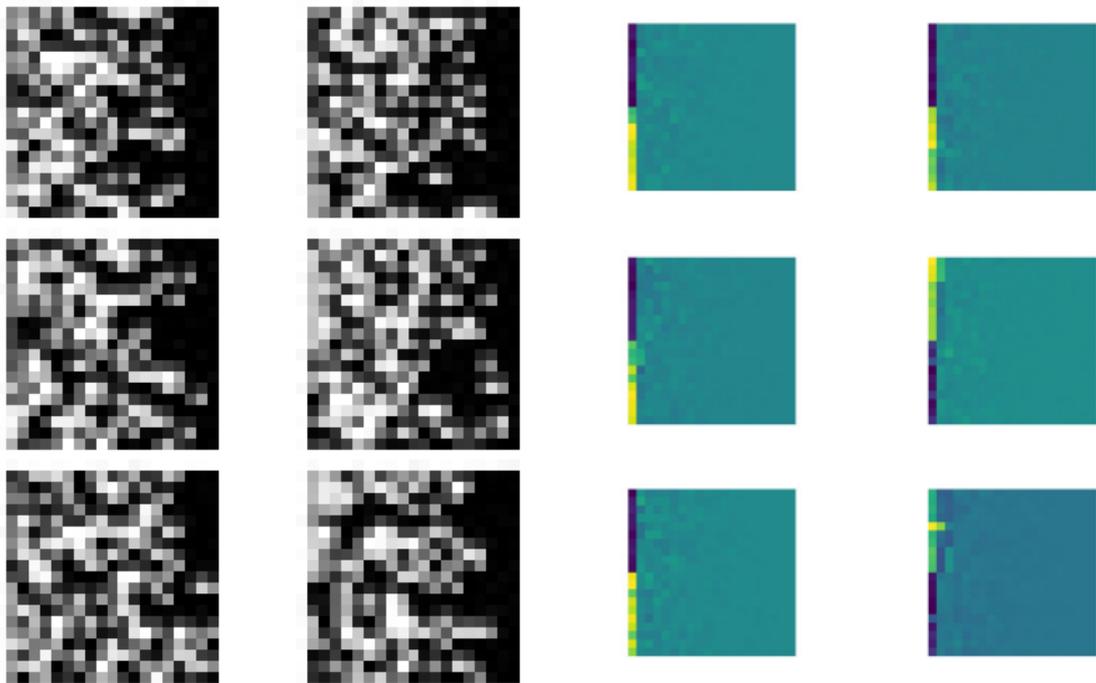


Figura 3.17 – Imagens geradas para a atividade "em pé" de uma rotina de experimentos. À esquerda, as imagens .png utilizadas como entrada para a neural convolucional, e à esquerda, imagens geradas após a transformação PCA.

## 4 Algoritmo da Rede Neural

No contexto de HAR, a escolha da arquitetura de rede neural é fundamental para garantir o desempenho do sistema e sua aplicabilidade em cenários reais. Neste capítulo, serão discutidas as principais arquiteturas utilizadas em estudos recentes que empregam *CSI Learning* para HAR, além de discorrer sobre a escolha da rede neural convolucional (CNN) como a arquitetura adotada neste trabalho.

### 4.1 Arquiteturas de Redes Neurais para Reconhecimento de Atividades Humanas (HAR)

O reconhecimento de atividades humanas (HAR) é uma área de pesquisa ativa que faz o uso de diversas arquiteturas de redes neurais disponíveis. A escolha da arquitetura correta pode ter um impacto significativo na eficácia e eficiência do modelo em termos de precisão, robustez e capacidade de generalização. Neste contexto, serão abordadas as arquiteturas LSTM, *Reinforcement learning*, *ProtoNet* e CNN, discutindo suas vantagens e limitações, bem como justificando a escolha da CNN para este projeto.

As redes *Long Short-Term Memory* (LSTM) são especialmente adequadas para lidar com dados sequenciais, como sequências temporais, devido à sua capacidade de manter e atualizar informações ao longo do tempo por meio de uma estrutura de memória controlada por portas. Essas portas permitem que a rede LSTM selecione quais informações devem ser mantidas ou esquecidas nas células em cada iteração. Uma célula é a unidade básica que compõe a rede, responsável por armazenar o estado da memória no tempo. Em HAR isso é particularmente útil para capturar padrões temporais nas atividades. A Figura 4.18 mostra a estrutura da LSTM escolhida por Zhang para *indoor positioning* baseado em CSI. A função de atualização das células de uma LSTM é a função de atualização da célula, descrita por

$$C_t = C_{t-1} \otimes f_t \otimes i_t \otimes \overline{C}_t, \quad (4.1)$$

em que  $C_t$  representa o estado da célula no tempo  $t$ ,  $f_t$  é a porta de esquecimento,  $i_t$  é a porta de entrada e  $\overline{C}_t$  é o candidato a novo conteúdo da célula. (Zhang; Qu; Wang, 2020)

Um dos principais desafios no uso de LSTMs para HAR com dados de CSI é a necessidade de um grande volume de dados para capturar adequadamente as dependências temporais e espaciais. Com um conjunto de dados limitado, as LSTMs podem sofrer de *overfitting*, aprendendo padrões específicos do treinamento sem generalizar bem para novos dados. Além disso, o alto custo computacional das LSTMs, devido ao processamento sequen-

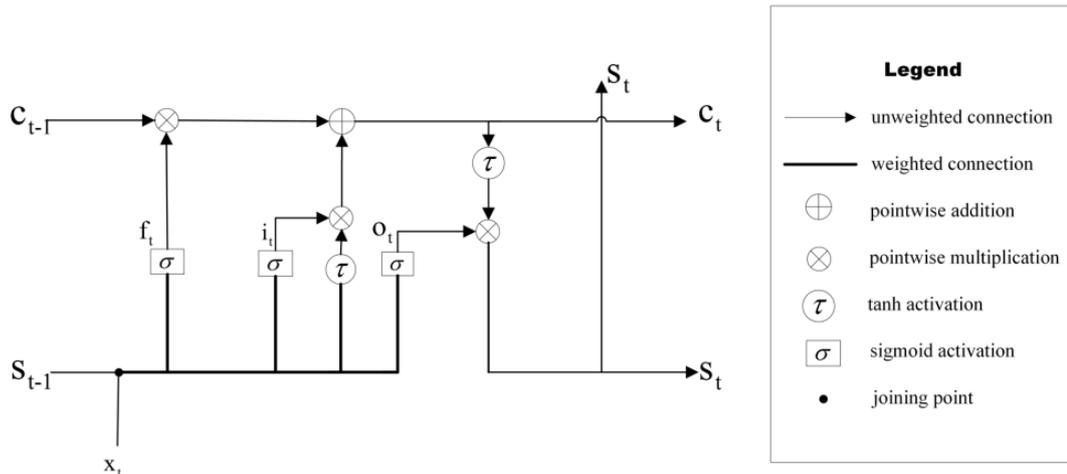


Figura 4.18 – Estrutura da LSTM utilizada por Zhang. Onde há uma camada escondida *S-shaped*. (Zhang; Qu; Wang, 2020)

cial e à atualização constante do estado da célula, torna o treinamento e a inferência mais lentos, dificultando sua aplicação em sistemas de tempo real. (Zhang; Qu; Wang, 2020)

O *Reinforcement Learning* (RL), é uma técnica em que um agente aprende a tomar decisões otimizadas por meio de interações com um ambiente. A cada interação, o agente recebe um estado do ambiente, escolhe uma ação, e então recebe uma recompensa ou punição, ajustando suas decisões futuras para maximizar as recompensas acumuladas. Em vez de ser explicitamente instruído, o agente explora o ambiente, aprendendo a melhorar seu desempenho a partir do *feedback*. O equilíbrio entre exploração de novas ações e exploração de ações conhecidas é um dos principais desafios do RL, permitindo ao modelo aprimorar suas decisões a longo prazo. O processo de otimização da política de aprendizado é descrito por

$$\nabla_{\theta_c} J(\theta_c) \approx \frac{1}{m} \sum_{k=1}^m \sum_{t=1}^T \nabla_{\theta_c} \log P(a_t | a_{(t-1):1}; \theta_c) (R_k - b), \quad (4.2)$$

em que  $\nabla_{\theta_c} J(\theta_c)$  é o gradiente da recompensa esperada  $J(\theta_c)$  associada aos parâmetros  $\theta_c$ ,  $m$  é o número de arquiteturas neurais testadas em um *batch*,  $T$  é o número de hiperparâmetros,  $\nabla_{\theta_c}$  é o gradiente em relação aos parâmetros  $\theta_c$ ,  $P(a_t | a_{(t-1):1}; \theta_c)$  é a probabilidade de tomar a ação  $a_t$  dado o histórico de ações  $a_{(t-1):1}$ ,  $R_k$  é a recompensa (acurácia) da  $k$ -ésima arquitetura neural, e  $b$  é a acurácia média das arquiteturas anteriores para evitar altas variâncias. (Ma et al., 2021)

No contexto do reconhecimento de atividades humanas (HAR) utilizando dados de CSI, o RL se mostra eficaz para adaptar o sistema a ambientes dinâmicos e variados. Ao aplicar RL, o modelo pode ajustar continuamente seus parâmetros conforme o ambiente ou as condições do cenário mudam, como variações na posição dos sensores ou nos movimentos

humanos, como acontece no design mostrado na Figura 4.19, proposto por Yongsen Ma. Essa adaptabilidade permite que o sistema de reconhecimento melhore sua precisão ao longo do tempo, tornando-o mais robusto em situações reais onde a localização ou as condições podem mudar drasticamente. Porém, os trabalhos recentes que demonstram os melhores resultados são os que aplicam uma técnica específica de RL, o *Deep Reinforcement learning* (DRL), que demanda uma grande quantidade de dados. Isso reflete num pequeno percentual de trabalhos recentes publicados com aplicações reais em HAR. (Shaqour; Hagishima, 2022)

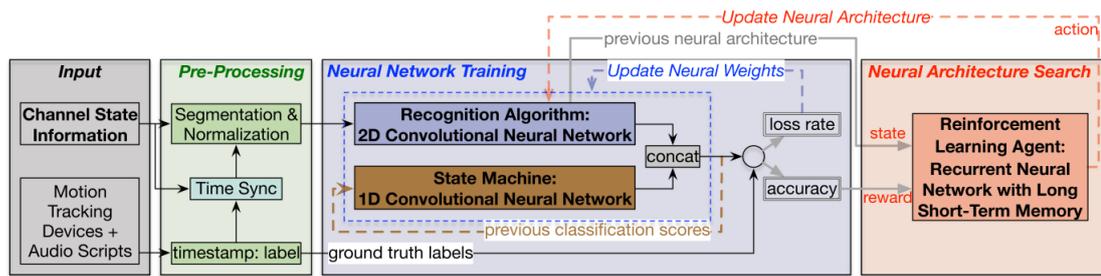


Figura 4.19 – Processo de treinamento do design de RL proposto por Yongsen Ma, para reconhecimento de atividades humanas. (Ma et al., 2021)

O *Few-Shot Learning* tem ganhado destaque no reconhecimento de atividades humanas usando CSI por abordar dois desafios principais: a necessidade de grandes conjuntos de dados e a queda de desempenho em novos ambientes. Modelos tradicionais de aprendizado de máquina e aprendizado profundo para HAR geralmente exigem grandes quantidades de dados de treinamento, e sua acurácia tende a cair quando o cenário ou ambiente muda, exigindo o re-treinamento dos modelos. Especialmente aplicado a tarefas de HAR usando CSI, essa técnica busca mitigar esses problemas ao permitir que o modelo reconheça atividades em novos ambientes (*X-Cross-Environment*) com pouquíssimos exemplos de treinamento. Uma abordagem comum envolve o uso de técnicas baseadas em grafos, com mecanismos de atenção dupla para capturar as relações entre diferentes tipos de atividades, minimizando o impacto de ruídos ambientais. Conforme demonstrado em estudos recentes, essa técnica pode alcançar alta acurácia no reconhecimento de atividades mesmo com pouquíssimos exemplos, tornando-a altamente adequada para ambientes dinâmicos e reais. (Zhang et al., 2022)

O *Prototypical Networks* (ProtoNet) é uma arquitetura de aprendizado de poucos exemplos (*Few-Shot Learning*) que visa generalizar a classificação com poucas amostras de cada classe. Em vez de aprender um classificador diretamente a partir dos dados, como fazem redes neurais tradicionais, o ProtoNet se baseia na ideia de calcular protótipos representativos para cada classe e, em seguida, fazer a classificação com base na proximidade a esses protótipos. (Bahadori; Ashdown; Restuccia, 2022)

Para cada classe do conjunto de suporte (amostras), o ProtoNet calcula o protótipo dessa classe. O protótipo é simplesmente a média dos *embedding* (representação vetorial

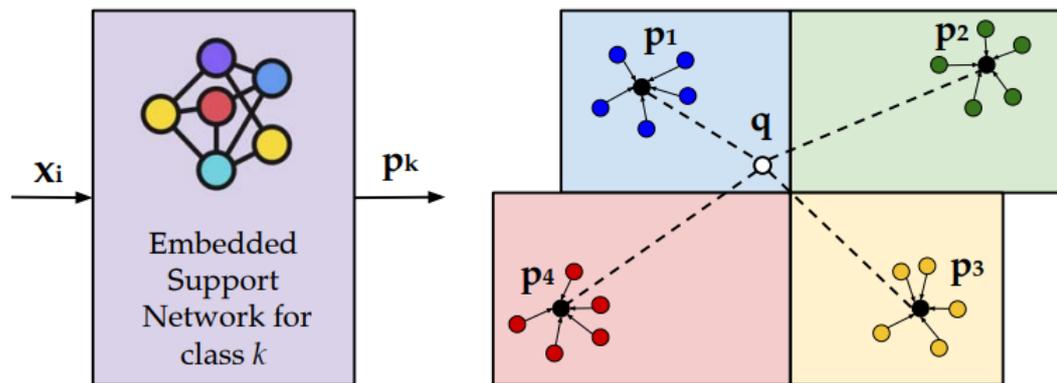


Figura 4.20 – ProtoNet utilizada no trabalho de Bahadori. (Bahadori; Ashdown; Restuccia, 2022)

de dimensões mais baixas do que os dados originais) de todas as amostras dessa classe. Essencialmente, o protótipo tenta capturar uma representação média da classe, de acordo com

$$c_k = \frac{1}{|S_k|} \sum_{x_i \in S_k} f_\theta(x_i), \quad (4.3)$$

em que  $c_k$  é o protótipo da classe  $k$ ,  $S_k$  são as amostras da classe  $k$ , e  $f_\theta(x_i)$  é o *embedding* da amostra  $x_i$  gerado pela rede neural  $f_\theta$ . (Bahadori; Ashdown; Restuccia, 2022)

Após calcular os protótipos, a classificação de uma nova amostra é feita com base na distância entre o *embedding* da amostra e cada um dos protótipos. O algoritmo usa, normalmente, a distância euclidiana entre o *embedding* da nova amostra e os protótipos das classes. A amostra é então atribuída à classe cujo protótipo estiver mais próximo, e os parâmetros do modelo são obtidos por

$$\Theta = \arg \min_k d(f_\theta(x_{\text{query}}), c_k), \quad (4.4)$$

em que  $d$  é a função de distância (geralmente, a distância euclidiana), e  $x_{\text{query}}$  é a amostra que está sendo classificada. (Bahadori; Ashdown; Restuccia, 2022)

As Redes Neurais Convolucionais (CNNs) são amplamente utilizadas em tarefas de HAR devido à sua capacidade de capturar características espaciais a partir de dados complexos, como as variações de amplitude e fase do CSI. Uma CNN típica para HAR com CSI consiste em várias camadas convolucionais seguidas por camadas de *pooling*, que reduzem a dimensionalidade das imagens geradas a partir dos dados CSI, preservando as características mais relevantes. As camadas convolucionais aplicam filtros que identificam características locais, como mudanças na amplitude ou fase associadas ao movimento humano. Após essas camadas, a saída é achatada em um vetor unidimensional, que é processado por camadas

totalmente conectadas, responsáveis pela classificação final das atividades humanas.(Zhang *et al.*, 2022)

Uma rede neural convolucional possui uma arquitetura hierárquica. A partir do sinal de entrada  $x$ , cada camada subsequente  $x_j$  é calculada como

$$x_j = \rho W_j x_{j-1}, \quad (4.5)$$

em que  $W_j$  é um operador linear e  $\rho$  é uma não-linearidade. Tipicamente, em uma CNN,  $W_j$  é uma convolução, e  $\rho$  é uma função retificadora  $\max(x,0)$  ou sigmoide  $\frac{1}{1+\exp(-x)}$ . O operador  $W_j$  pode ser interpretado como uma pilha de filtros convolucionais. Dessa forma, as camadas são mapas de filtros e cada camada pode ser descrita como uma soma de convoluções da camada anterior,

$$x_j(u, k_j) = \rho \left( \sum_k (x_{j-1}(\cdot, k) * W_{j,k_j}(\cdot, k)) (u) \right), \quad (4.6)$$

em que  $*$  é o operador de convolução

$$(f * g)(x) = \sum_{u=-\infty}^{\infty} f(u)g(x - u). \quad (4.7)$$

Assim, tipicamente, os pesos  $W_j$  são aprendidos por descida estocástica de gradiente, utilizando o algoritmo de retro-propagação (*backpropagation*) para calcular os gradientes. (Koushik, 2016)

A classificação é realizada com base nas características extraídas pela CNN durante o processo de aprendizado. O modelo ajusta seus parâmetros ao longo do treinamento para minimizar o erro de predição. Uma arquitetura CNN utilizada para HAR é mostrada na Figura 4.21.

As CNNs são relativamente eficientes em termos de computação e eficazes na identificação de padrões espaciais. No entanto, uma limitação das CNNs é a dificuldade em capturar dependências temporais nos dados, o que pode ser mitigado pela combinação com outras redes, como LSTMs. A CNN apresenta um equilíbrio entre precisão e eficiência computacional, além de uma capacidade de lidar com a complexidade das imagens CSI. A combinação de CNNs com outras técnicas, como o uso de janelas deslizantes para capturar a temporalidade, pode ser uma direção futura para melhorar a aplicação em automação predial.

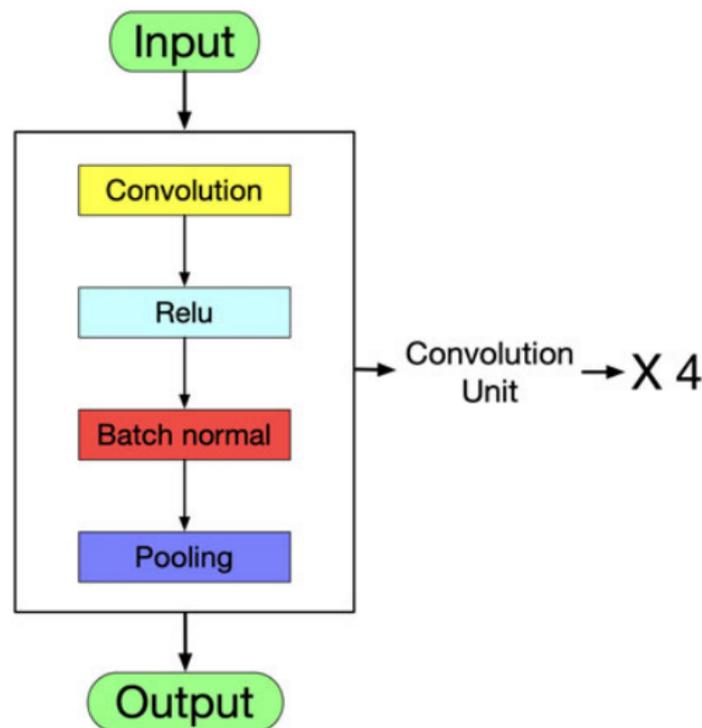


Figura 4.21 – CNN utilizada no trabalho de Zhang. (Zhang *et al.*, 2022)

## 4.2 Escolha e Estrutura da Rede Neural Convolutacional (CNN)

No contexto do reconhecimento de atividades humanas, há a necessidade de uma abordagem robusta e eficiente para a classificação de padrões complexos. As CNNs são amplamente reconhecidas por sua capacidade superior de capturar hierarquias de características em dados de imagem, tornando-as ideais para tarefas de classificação visual. Essa capacidade é particularmente relevante neste projeto, onde foi escolhido converter os dados de CSI em imagens .png. Assim, as variações sutis nas imagens geradas a partir dos dados precisam ser interpretadas para identificar atividades como "vazio", "sentado", "em pé" e "caminhando".

A decisão de utilizar uma CNN foi fundamentada em sua arquitetura especializada, que combina camadas convolucionais para extração de características com camadas totalmente conectadas para a classificação final. Além disso, a CNN pode ser treinada para generalizar a partir de exemplos, tornando-a eficaz em ambientes onde os dados de entrada podem variar por interferências, reflexões e ruído.

A arquitetura da CNN utilizada é composta por camadas convolucionais, seguidas por camadas de *pooling* e *fully connected* (totalmente conectada). A função de ativação ReLU foi empregada para introduzir não-linearidade, e a função *softmax* foi usada na última camada para calcular as probabilidades associadas a cada classe de atividade. O modelo foi treinado

com o algoritmo de otimização Adam, usando a função de perda *CrossEntropyLoss* (perda de entropia cruzada), adequada para o caso de várias classes. Sua estrutura é mostrada na Figura 4.22.

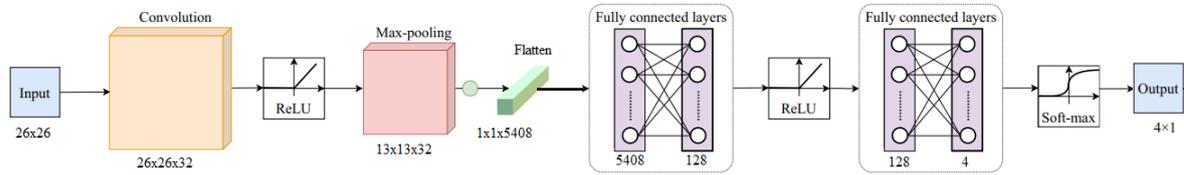


Figura 4.22 – Arquitetura da CNN utilizada. Composta por uma camada convolucional seguida de uma função de ativação *ReLU*, uma camada de *pooling*, uma camada *Flatten* para adaptar os dados para entrada da próxima camada (totalmente conectada), mais uma função *ReLU*, e outra camada totalmente conectada com saída de dimensão 4 (as quatro classes). A função *Softmax* foi usada para classificação final das atividades baseada em probabilidades. Adaptada de (Bahadori; Ashdown; Restuccia, 2022).

Para facilitar a compreensão do processo de desenvolvimento e implementação do modelo, foi elaborado um pseudocódigo que descreve a lógica fundamental do algoritmo. Após o pseudocódigo, cada parte do código é explicada de maneira mais detalhada, oferecendo uma visão mais aprofundada das decisões e implementações realizadas ao longo do desenvolvimento.

### 4.2.1 Pseudocódigo

O pseudocódigo apresentado a seguir serve como uma representação simplificada e abstrata do algoritmo implementado. Ele oferece uma visão geral das principais etapas e fluxos de trabalho, facilitando o entendimento do código para leitores que buscam compreender a lógica fundamental sem se aprofundar nos detalhes técnicos específicos da implementação em Python.

INICIO

DEFINIR lista Tamanhos\_Recorte = [16, 20, 24]

INICIALIZAR lista Melhores\_Acuracias

CRIAR dataset personalizado

PARA cada Tamanho em Tamanhos\_Recorte FAÇA

    INICIAR Transformação com Recorte Aleatório de Tamanho x Tamanho

    CARREGAR imagens e rótulos das pastas de acordo com as atividades

    REDIMENSIONAR imagem para 34x34 pixels

    TRANSFORMAR imagem se Transformação estiver definida

DEFINIR CNN com:

    Camada convolucional Conv1

```

Função de ativação ReLU
Camada de pooling MaxPool
Camadas totalmente conectadas FC1 e FC2
Função Softmax para classificação

```

```

PARA cada época de 1 até 300 FAÇA
  TREINAR o modelo nos dados de treino
  VALIDAR o modelo nos dados de validação
  CALCULAR Acurácia
  SE Acurácia > Melhor_Acurácia ENTÃO
    SALVAR o modelo como 'best_cnn_model.pth'
    ARMAZENAR Melhor_Acurácia

```

```

PLOTAR gráfico de Perda_Treino e Perda_Validação por época

```

```

FIM

```

## 4.2.2 Implementação

Agora as partes mais importantes do código em Python, que foram desenvolvidas para a implementação da CNN, serão explicadas de maneira mais detalhada. Cada etapa do código, desde a definição das dimensões de recorte até a arquitetura da CNN e o processo de treinamento, será desmembrada e discutida.

### 4.2.2.1 Pré-processamento dos Dados com *RandomCrop*

O primeiro passo para a implementação do modelo é o pré-processamento dos dados de entrada. As imagens geradas a partir dos dados de CSI são carregadas e transformadas em tensores para serem processadas pela CNN. Além disso, diferentes tamanhos de recorte (16x16, 20x20, 24x24) são aplicados para experimentação e otimização.

Código 4.1 – Crop\_sizes.

```

1  crop_sizes = [16, 20, 24]
2  for crop_size in crop_sizes:
3      transform = transforms.Compose([transforms.ToTensor(),
                                     transforms.RandomCrop((crop_size, crop_size))])

```

O código define três tamanhos de recorte diferentes (16, 20 e 24 *pixels*) e, para cada um, aplica uma transformação que inclui a conversão das imagens em tensores e o recorte aleatório. Essas transformações permitem que a CNN aprenda a partir de diferentes porções das imagens de entrada, o que ajuda na generalização do modelo.

#### 4.2.2.2 Dataset Personalizado

O *Dataset* personalizado, contendo as imagens associadas às atividades "vazio", "sentado", "em pé" e "caminhando", é carregado por meio da classe *CustomDataset*. Esta classe percorre os diretórios de imagens, mapeia cada uma delas para sua respectiva classe e aplica as transformações de pré-processamento.

Código 4.2 – CustomDataset.

```

1  class CustomDataset(Dataset):
2  def __init__(self, root, transform=None):
3
4      folders = os.listdir(root)
5      self.data = []
6      self.labels = []
7
8      self.label_map = {"empty": 0,      # VAZIO
9                        "sit": 1,        # SENTADO
10                       "stand": 2,     # EM P
11                       "walk": 3}     # CAMINHANDO
12
13     for folder in folders:
14         path = os.path.join(root, folder)
15         label_folders = os.listdir(path)
16         for label_folder in label_folders:
17             label_path = os.path.join(path, label_folder)
18             for file in os.listdir(label_path):
19                 data_file = os.path.join(path, label_folder,
20                                           file)
21                 if "image" in file:
22                     self.data.append(data_file)
23                     self.labels.append(label_folder)
24
25     self.transform = transform
26
27     def __len__(self):
28         return len(self.data)
29
30     def __getitem__(self, idx):
31         image_path = self.data[idx]
32         image = Image.open(image_path)
33         image = image.resize((34, 34), Image.NEAREST)
34
35         label = self.label_map[self.labels[idx]]
36
37         if self.transform:
38             image = self.transform(image)
39
40         return image, label

```

- A classe percorre as pastas do diretório de imagens, associando as imagens às classes

correspondentes usando o dicionário `label_map`. As imagens são armazenadas na lista `data`, e os rótulos, na lista `labels`.

- Para cada índice fornecido, a imagem correspondente é carregada, redimensionada para  $34 \times 34$  *pixels*, transformada, e retornada junto com seu rótulo numérico. Esse processo prepara as imagens para serem utilizadas na CNN.

#### 4.2.2.3 Arquitetura da CNN

A arquitetura da rede neural convolucional é composta por uma camada convolucional com 32 filtros, seguida por uma camada de *pooling*, duas camadas totalmente conectadas (*fully connected*) e uma camada *softmax* para normalizar a saída da rede em uma distribuição de probabilidades.

Código 4.3 – CNN.

```

1  class CNN(nn.Module):
2  def __init__(self):
3      super(CNN, self).__init__()
4      self.conv1 = nn.Conv2d(1, 32, kernel_size=3, padding=1)
5      self.relu = nn.ReLU()
6      self.maxpool = nn.MaxPool2d(kernel_size=2, stride=2)
7      self.fc1 = nn.Linear(32 * 12 * 12, 128)
8      self.fc2 = nn.Linear(128, 4) # 10 output classes
9
10
11 def forward(self, x):
12     x = self.conv1(x)
13     x = self.relu(x)
14     x = self.maxpool(x)
15     x = x.view(x.size(0), -1) # Flatten the tensor
16     x = self.fc1(x)
17     x = self.relu(x)
18     x = self.fc2(x)
19     x = F.softmax(x, dim=1)
20     return x

```

- Camada convolucional (`conv1`): A primeira camada convolucional aplica 32 filtros às imagens de entrada, extraindo características locais como bordas e texturas.
- Função de ativação ReLU: Após a convolução, a função ReLU introduz não-linearidade no modelo, permitindo que ele aprenda relações complexas.
- Camada de *pooling maxpool*: A camada de *pooling* reduz a dimensão da imagem ao selecionar valores máximos em blocos  $2 \times 2$ , reduzindo a complexidade computacional e ajudando a evitar *overfitting*.

- Camadas *fully connected* (fc1 e fc2): Essas camadas são responsáveis por combinar as características extraídas pela convolução e produzir as classes de saída.
- *softmax*: A função *softmax* na última camada converte as saídas em probabilidades, que indicam a probabilidade de a imagem pertencer a cada uma das classes de atividades.

#### 4.2.2.4 Treinamento e Validação

O treinamento da rede neural foi realizado ao longo de um número pré-definido de épocas. A função de perda de entropia cruzada (*CrossEntropyLoss*) é usada para calcular a diferença entre as previsões do modelo e os rótulos reais. O otimizador Adam é utilizado para atualizar os pesos da rede com base no gradiente calculado. A cada época, o modelo é treinado no conjunto de dados de treinamento e validado no conjunto de validação. Caso a acurácia de validação melhore, o modelo é salvo.

Código 4.4 – Training Loop.

```

1     optimizer = optim.Adam(model.parameters(), lr=0.0001)
2     criterion = nn.CrossEntropyLoss()
3     best_acc = 0
4
5     for epoch in range(num_epochs):
6         train_loss = 0
7         for i, (inputs, labels) in enumerate(dataloader):
8             optimizer.zero_grad()
9             outputs = model(inputs)
10            loss = criterion(outputs, labels)
11            train_loss += loss.item()
12            loss.backward()
13            optimizer.step()
14
15            val_loss = 0
16            correct = 0
17            total = 0
18
19            for i, (inputs, labels) in enumerate(val_dataloader):
20                with torch.no_grad():
21                    outputs = model(inputs)
22                    loss = criterion(outputs, labels)
23                    val_loss += loss.item()
24
25                    total += labels.size(0)
26                    _, predicted = torch.max(outputs, dim=1)
27                    correct += (predicted == labels).sum().item()
28
29            accuracy = 100 * correct / total
30
31            if accuracy > best_acc:
32                best_acc = accuracy
33                torch.save(model.state_dict(), 'best_cnn_model.pth')

```

- **Otimização e Função de Perda:** O otimizador Adam ajusta os pesos da CNN com base no gradiente calculado pela função de perda `CrossEntropyLoss`, que mede a diferença entre as previsões e os rótulos reais.
- **Treinamento:** A cada época, os dados do conjunto de treinamento são passados pela rede, e os pesos são ajustados para minimizar a perda.
- **Validação:** Após o treinamento em cada época, os dados de validação são usados para avaliar o desempenho do modelo, e a acurácia é calculada.
- **Salvamento do Modelo:** Caso a acurácia de validação supere a melhor acurácia anterior, o modelo é salvo.

#### 4.2.2.5 Acurácia

A acurácia é uma métrica de desempenho usada para avaliar a performance de um modelo de classificação. Ela mede a proporção de previsões corretas em relação ao número total de previsões feitas, expressando a taxa de acertos do modelo. De forma resumida, a acurácia  $A$  pode ser expressa como

$$A = \frac{N_{\text{corretos}}}{N_{\text{total}}} \times 100, \quad (4.8)$$

em que  $N_{\text{corretos}}$  é o número de amostras corretamente classificadas, e  $N_{\text{total}}$  é o número total de amostras. Essa métrica fornece uma visão geral do quão bem o modelo está classificando os dados, em cada época de validação.

#### 4.2.2.6 Visualização dos Resultados

Após o treinamento, os resultados foram visualizados através de um gráfico que mostra a perda de treinamento e de validação ao longo das épocas. Com esse gráfico, podemos identificar um possível *overfitting*, que acontece quando a curva de perda da validação começa a crescer novamente, enquanto a curva de perda do treinamento segue diminuindo, como é mostrado na Figura 4.23.

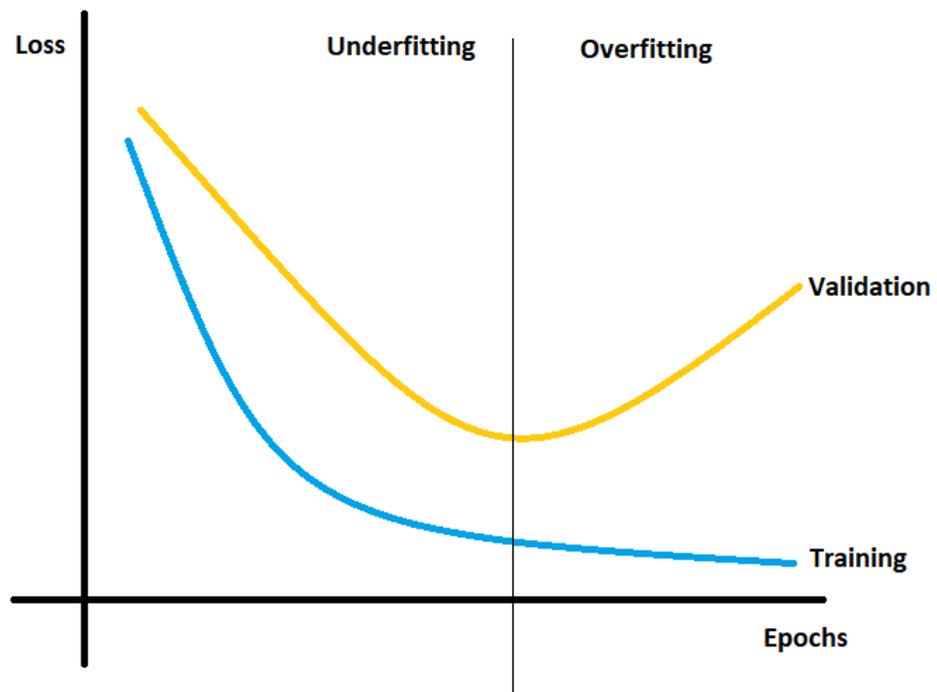


Figura 4.23 – Exemplo de identificação de *overfitting* no gráfico *loss x epochs*.

## 5 Resultados e Análise Crítica

Neste capítulo, os resultados obtidos durante a realização dos experimentos serão apresentados e discutidos. A análise crítica visa identificar as principais contribuições do trabalho e os desafios encontrados ao longo do processo de implementação.

### 5.1 *Wi-Fi Sensing*

Mesmo diante do desafio de compreender e adaptar os códigos com pouca documentação para o ESP32, e também para o pré-processamento de dados, foi possível realizar com sucesso a aquisição dos dados de CSI com a rede de sensores mostrada na Figura 3.12 durante os experimentos. No entanto, foi observado que o algoritmo de limpeza de dados precisou inserir, em média, 30% de linhas adicionais para garantir que os arquivos seguissem o padrão da sequência de sensores (6, 7, 8, 9), como é mostrado na Tabela 5.1. Além da inserção de linhas, o algoritmo também fez deleções de linhas repetidas, esses comportamentos podem ser indicativos de possíveis problemas durante o processo de aquisição, como interferências, falhas nos sensores, ou até erros de programação nos dispositivos ESP32 responsáveis pela captura dos dados.

Tabela 5.1 – Taxas de inserções de linhas para alguns arquivos com dados CSI das rotinas de experimentos, adquiridos no LARA.

Arquivo	Inserções (%)
cEu_rot1_04092024	28.7
cEu_rot2_04092024	32.6
cEu_rot3_04092024	33.8
cEu_rot4_04092024	29.6
cEu_rot5_04092024	30.7
cEu_rot6_04092024	28.3
cEu_rot7_04092024	25.2
cEu_rot8_04092024	25.6
cEu_rot9_04092024	22.8
cEu_rot10_04092024	26.7
cEu_rot11_04092024	34.8
cEu_rot12_04092024	27.2

Apesar desses ajustes, os dados de CSI, uma vez limpos, se mostraram adequados para as etapas subsequentes de pré-processamento e modelagem. O filtro *Butterworth* aplicado na fase de limpeza garantiu a eliminação de ruídos de alta frequência, enquanto a técnica de Análise de Componentes Principais (PCA) foi eficaz para reduzir a dimensionalidade

dos dados (de 34X432 para 34x34), resultando em uma representação mais compacta para a criação de imagens de treinamento.

## 5.2 Desempenho da CNN na Classificação das Atividades

As imagens geradas a partir dos dados de CSI foram utilizadas com sucesso para o treinamento do modelo de rede neural convolucional (CNN). Durante os experimentos, foram utilizados um total de 25 conjuntos de dados (com 4 minutos de duração), cada um contendo amostras de 1 minuto para cada atividade, como exemplificado na Figura 3.17. Desses, 20 conjuntos foram destinados ao treinamento, enquanto os outros 5 conjuntos foram reservados para validação.

O modelo de CNN foi capaz de treinar e ajustar seus pesos para classificar as atividades humanas, atingindo uma acurácia máxima de 48,33%, para o conjunto de validação. No entanto, foi detectado *overfitting* a partir da época 163, com a curva da perda no conjunto de validação estabilizando e indicando que o modelo começou a se ajustar excessivamente aos dados de treinamento, perdendo a capacidade de generalizar para novos dados.

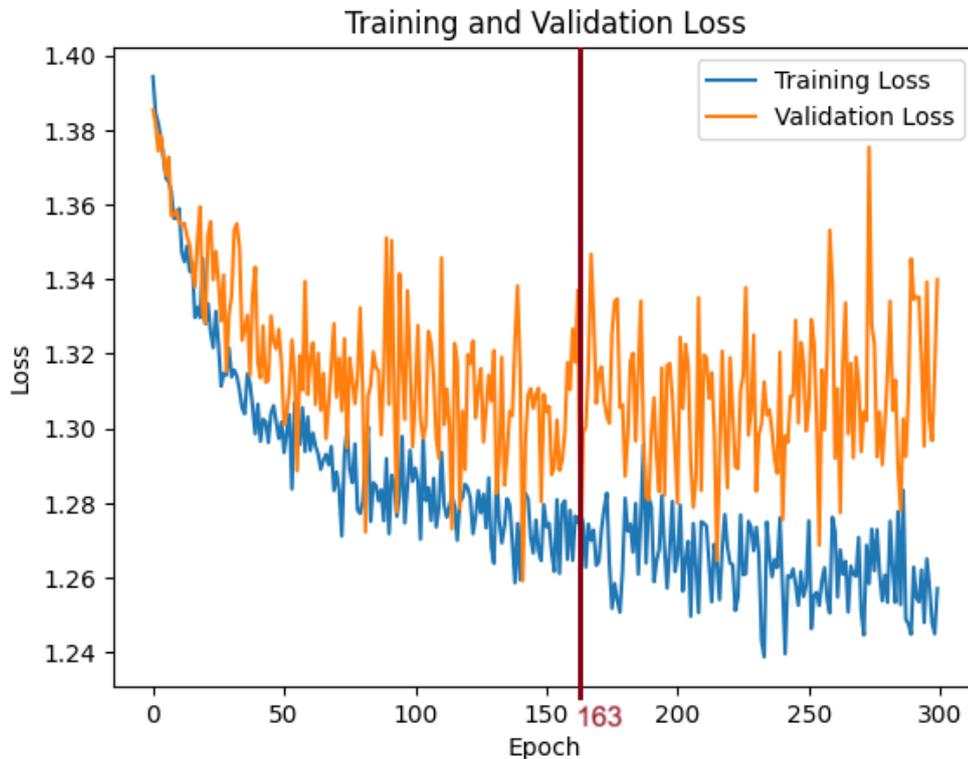


Figura 5.24 – Gráfico de Perda x Época da CNN. Curva azul representa o treinamento, e a curva laranja representa a validação. O *overfitting* foi detectado a partir da época 163.

A Figura 5.25 apresenta a matriz de confusão do modelo da CNN com melhor acurácia na validação (48,33%), evidenciando as dificuldades do modelo em classificar corretamente

as atividades humanas com base nos dados de CSI disponíveis. Cada linha da matriz representa as atividades reais, enquanto as colunas indicam as atividades previstas pela rede neural. O conjunto de validação possui 120 amostras ao todo, com 30 amostras de cada atividade. O modelo obteve 58 acertos, dos 120 possíveis. Observa-se que a rede neural teve um desempenho misto, em que a atividade "caminhando" foi a melhor classificada, com 24 acertos, enquanto as outras apresentaram maior sobreposição nos resultados. Isso reforça a necessidade de ajustes na arquitetura da rede neural ou na aquisição de dados para melhorar a acurácia do sistema.

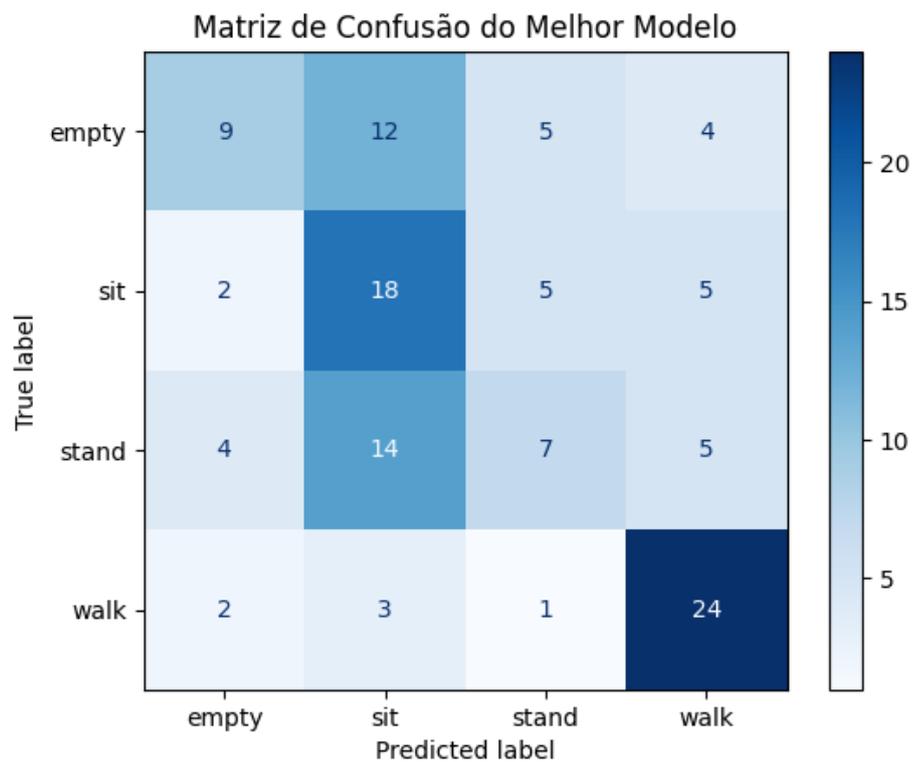


Figura 5.25 – Matriz de confusão para o conjunto de dados de validação. Cada linha da matriz representa as atividades reais, enquanto as colunas indicam as atividades previstas pela rede neural. O conjunto de validação é constituído de 30 amostras para cada atividade.

Esses resultados destacam tanto os sucessos quanto os desafios encontrados no desenvolvimento do sistema de reconhecimento de atividades usando *Wi-Fi Sensing* e CNN, dando espaço para melhorias nas etapas futuras.

## 6 Conclusões e Perspectivas

No último capítulo, serão apresentadas as conclusões deste trabalho, destacando as contribuições feitas para a área de reconhecimento de atividades humanas. Além disso, serão discutidas possíveis melhorias e direções futuras para o avanço desta pesquisa.

### 6.1 Conclusões

O presente trabalho apresentou a implementação de um sistema de reconhecimento de atividades humanas (HAR) em ambientes prediais, utilizando redes de sensores Wi-Fi para captar dados de CSI e uma rede neural convolucional (CNN) para classificar as atividades. A abordagem se mostrou eficaz em explorar o *Wi-Fi Sensing* como uma alternativa não invasiva e que preserva a privacidade dos indivíduos, eliminando a necessidade de dispositivos visuais, como câmeras, que podem comprometer a conformidade com regulamentos como a LGPD.

Embora os dados de CSI tenham sido capturados corretamente, observou-se a necessidade de inserir cerca de 30% de linhas adicionais durante a fase de limpeza de dados, indicando possíveis problemas de interferência no ambiente ou falhas nos sensores ESP32. Ainda assim, o sistema de pré-processamento de dados foi bem-sucedido em gerar imagens adequadas para o treinamento do modelo.

O treinamento da CNN resultou em uma acurácia máxima de 48,33% no conjunto de validação, o que aponta para a viabilidade do uso de redes convolucionais para a classificação de atividades com base em dados de CSI. No entanto, o fenômeno de *overfitting* observado a partir da época 163 indica que ajustes adicionais no modelo, como aumento da quantidade de dados de treinamento ou técnicas de regularização, são necessários para melhorar a generalização.

### 6.2 Perspectivas Futuras

Com base nos resultados obtidos, existem diversas frentes promissoras para o aprimoramento deste projeto:

- **Aprimoramento da Aquisição de Dados:** É essencial investigar as causas dos problemas de aquisição que resultaram na necessidade de inserção exacerbada de linhas durante a limpeza de dados. Possíveis melhorias incluem ajustes na configuração dos sensores ESP32 e na topologia da rede de sensores.

- **Aumento do Conjunto de Dados:** Para combater o *overfitting*, uma perspectiva importante é aumentar a quantidade de dados de treinamento. A coleta de mais rotinas de atividades, em diferentes cenários e com mais participantes, pode contribuir significativamente para melhorar a generalização do modelo.
- **Exploração de Arquiteturas de Rede Alternativas:** Embora as CNNs tenham mostrado bons resultados, explorar outras arquiteturas de redes neurais, como as Redes Neurais Recorrentes (RNNs) ou LSTMs, pode ser vantajoso para lidar com a natureza temporal dos dados de CSI, uma vez que essas redes são capazes de capturar dependências temporais de forma eficiente.
- **Melhorias no Modelo de Classificação:** Para superar o problema de *overfitting* e aumentar a acurácia do modelo, pode-se explorar técnicas como *data augmentation* (aumentar artificialmente a variação dos dados), regularização (*dropout*, *weight decay*), e ajuste mais fino dos hiperparâmetros da rede neural.

Essas melhorias podem tornar o sistema ainda mais robusto e aplicável em um cenário real, proporcionando uma solução eficiente para o reconhecimento de atividades em ambientes prediais, com potencial de reduzir o consumo de energia e automatizar ambientes de maneira inteligente.

# Referências

- ALSAIFY, B. A.; ALMAZARI, M. M.; ALAZRAI, R.; ALOUNEH, S.; DAOUD, M. I. A csi-based multi-environment human activity recognition framework. **Applied Sciences**, v. 12, n. 2, 2022. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/12/2/930>. Citado na p. 17.
- BAHADORI, N.; ASHDOWN, J.; RESTUCCIA, F. Rewis: Reliable wi-fi sensing through few-shot multi-antenna multi-receiver csi learning. **WoWMoM 2022**, p. 50–59, 2022. Citado nas pp. 6, 7, 21, 23, 40, 41 e 44.
- BAUCHSPIESS, A.; MANE, T.; FÜRST-WAKTER, I.; BECKER, J. Distributed esp-now csi human activity recognition for few-shot energy saving in building automation. **IEEE CYBER 2024**, IEEE, p. 1–6, 2024. Citado nas pp. 6, 14, 24, 29 e 31.
- BIAN, S.; LIU, M.; ZHOU, B.; LUKOWICZ, P. The state-of-the-art sensing techniques in human activity recognition: A survey. **Sensors**, v. 22, n. 12, p. 4596–4625, 2022. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/22/12/4596>. Citado nas pp. 6 e 16.
- BRASIL. Lei n.º 13.709, de 14 de agosto de 2018. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 2018. Disponível em: [https://www.planalto.gov.br/ccivil\\_03/\\_ato2015-2018/2018/lei/l13709.htm](https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm). Citado na p. 15.
- CHEN, Z.; ZHANG, L.; JIANG, C.; CAO, Z.; CUI, W. Wifi csi based passive human activity recognition using attention based blstm. **IEEE TMC**, v. 18, n. 11, p. 2714–2724, 2019. Citado nas pp. 15, 21, 22 e 31.
- ESPRESSIF. **ESP-NOW FAQ**. 2024. Accessed on August 9, 2024. Disponível em: <https://docs.espressif.com/projects/esp-faq/en/latest/application-solution/esp-now.html>. Citado na p. 27.
- KEYSIGHT. **Concepts of Orthogonal Frequency Division Multiplexing (OFDM) and 802.11 WLAN**. 2024. Accessed on September 9, 2024. Disponível em: [https://helpfiles.keysight.com/csg/89600B/Webhelp/Subsystems/wlan-ofdm/content/ofdm\\_basicprinciplesoverview.htm](https://helpfiles.keysight.com/csg/89600B/Webhelp/Subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm). Citado nas pp. 6 e 17.
- KOUSHIK, J. Understanding convolutional neural networks. **Cornell University**, p. 1–6, 2016. Disponível em: <https://arxiv.org/abs/1605.09081>. Citado na p. 42.
- MA, Y.; ARSHAD, S.; MUNIRAJU, S.; TORKILDSON, E.; RANTALA, E.; DOPPLER, K.; ZHOU, G. Location- and person-independent activity recognition with wifi, deep neural networks, and reinforcement learning. **ACM TIT**, Association for Computing Machinery, v. 2, n. 1, p. 1–25, jan 2021. Disponível em: <https://doi.org/10.1145/3424739>. Citado nas pp. 7, 39 e 40.
- MA, Y.; ZHOU, G.; WANG, S. Wifi sensing with channel state information: A survey. **ACM Computing Surveys**, v. 52, n. 46, p. 1–36, 2019. Citado nas pp. 21 e 22.

---

MUAAZ, M.; CHELLI, A.; GERDES, M. W.; PATZOLD, M. Wi-sense: a passive human activity recognition system using wi-fi and convolutional neural network and its integration in health information systems. **Annals of Telecommunications**, IEEE, v. 77, p. 163–175, 2022. Citado nas pp. 15, 21 e 22.

RODRIGUES, A. S.; SILVA, M. P. M. da. **Controle Antecipativo por estimativa de carga térmica em vídeo visando eficiência energética na climatização de ambiente predial**. 1-134 p. Dissertação (Monografia) — Universidade de Brasília, Brasília, 2018. Citado na p. 14.

SHAQOUR, A.; HAGISHIMA, A. Systematic review on deep reinforcement learning-based energy management for different building types. **Energies**, v. 15, n. 22, p. 1–27, 2022. ISSN 1996-1073. Disponível em: <https://www.mdpi.com/1996-1073/15/22/8663>. Citado na p. 40.

SHEN, X.; GUO, L.; LU, Z.; WEN, X.; ZHOU, S. Wiagent: Link selection for csi-based activity recognition in densely deployed wi-fi environments. **IEEE WCNC 2021**, p. 1–6, 2021. Citado na p. 18.

ZHANG, Y.; CHEN, Y.; WANG, Y.; LIU, Q.; CHENG, A. Csi-based human activity recognition with graph few-shot learning. **IEEE ITJ**, v. 9, n. 6, p. 4139–4151, 2022. Citado nas pp. 23, 24 e 40.

ZHANG, Y.; QU, C.; WANG, Y. An indoor positioning method based on csi by using features optimization mechanism with lstm. **IEEE Sensors**, v. 20, n. 9, p. 4868–4878, 2020. Citado nas pp. 7, 38 e 39.

ZHANG, Y.; WANG, X.; WANG, Y.; CHEN, H. Human activity recognition across scenes and categories based on csi. **IEEE TMC**, v. 21, n. 7, p. 2411–2420, 2022. Citado nas pp. 7, 42 e 43.