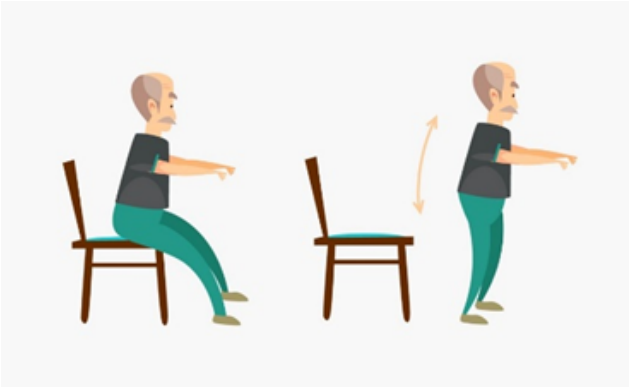
NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Aplicações Avançadas de Engenharia Biomédica

Docentes:

Pedro Vieira e Hugo Gamboa

Sistema de Monitorização do Movimento



Mestrado Integrado em Engenharia Biomédica

Ana Fernandes 59995

Marta Simão 60666

2024/2025

Índice

Page

1 Introdução **2**

1.1 Objetivo e Motivação 2

2 Arquitetura do Sistema **3**

2.1 Comunicação MQTT 3

2.2 Sensor M5Stick 3

2.3 Raspberry Pi 4

2.4 Aquisição de dados 5

3 Machine Learning **6**

3.1 Extração de Features 6

3.2 Dataset de Treino e Escolha do Classificador 6

4 Interface do Utilizador **9**

4.1 Streamlit 9

5 Discussão **13**

6 Conclusão **14**

1 Introdução

1.1 Objetivo e Motivação

O sedentarismo é um problema crescente na sociedade atual, especialmente entre a população idosa, com consequências significativas para a saúde e a autonomia. Este grupo enfrenta frequentemente limitações de mobilidade e uma maior predisposição para condições de saúde associadas à inatividade física. A situação agrava-se quando muitos idosos vivem sozinhos, dificultando o acompanhamento regular por parte dos familiares e aumentando os riscos de quedas ou complicações de saúde não detetadas a tempo.

Movimentos básicos do dia-a-dia, como sentar, levantar e andar, são indicadores fundamentais da condição física e funcional de uma pessoa. Alterações nestes padrões podem sinalizar problemas como fraqueza muscular, dificuldades de equilíbrio ou até condições de saúde mais graves. Contudo, atualmente, existe uma carência de soluções tecnológicas acessíveis que permitam monitorizar estes movimentos de forma remota, prática e eficaz.

Com base neste contexto, o objetivo deste projeto é desenvolver um sistema inteligente que utilize algoritmos de machine learning para identificar e classificar automaticamente os movimentos de sentar, levantar e andar. Este sistema, baseado em dados reais recolhidos previamente, pretende permitir o acompanhamento remoto da atividade física dos idosos, promovendo a deteção precoce de comportamentos sedentários ou alterações nos padrões de movimento. A solução visa melhorar a qualidade de vida e a segurança dos idosos, garantindo simultaneamente maior tranquilidade aos seus familiares, ao possibilitar um acompanhamento contínuo e à distância.

2 Arquitetura do Sistema

2.1 Comunicação MQTT

O protocolo MQTT (Message Queuing Telemetry Transport) desempenha um papel fundamental na arquitetura do sistema, facilitando uma comunicação eficiente, rápida e bidirecional entre os diferentes componentes. Este protocolo segue um modelo de publicação/subscrição, o que permite a troca de mensagens de forma flexível e com baixo consumo de recursos, características essenciais para sistemas que operam em tempo real.

No âmbito deste projeto, o Raspberry Pi atua como publicador, enviando mensagens para o broker, enquanto o Streamlit atua como subscritor, recebendo essas mensagens para processar e exibir os resultados. O tópico utilizado tanto para subscrição como para publicação é ***"AAI/MARTANA/cmd"***.

Na aplicação desenvolvida, a comunicação MQTT inicia-se com a interação do utilizador na interface gráfica, ao selecionar o botão **"Iniciar Programa"**. As principais mensagens publicadas no MQTT acontecem quando o utilizador carrega em dois botões:

- **Iniciar Leitura de Dados com o M5Stick**: Indicar o início da recolha de dados. Quando o Arduino recebe esta mensagem, o sensor M5Stick é ativado para captar informações de movimento e enviá-las para o sistema. Os dados obtidos pelo sensor são publicados em tempo real no MQTT.
- **Classificar Último Movimento**: Acede aos últimos dados adquiridos e faz a previsão do movimento, publicando o resultado no MQTT.

2.2 Sensor M5Stick

O sensor M5Stick é um dispositivo compacto e versátil, equipado com acelerómetro e giroscópio, permitindo a captura precisa de dados de movimento em três eixos. Este sensor desempenha um

papel crucial no sistema, sendo responsável pela recolha de informações detalhadas sobre os movimentos realizados pelo utilizador. Os dados capturados são transmitidos para o sistema central através de comunicação BLE (Bluetooth Low Energy), que garante eficiência e baixo consumo de energia.

2.2.1 Funcionamento do Sensor M5Stick

No caso da nossa aplicação, os sensores do M5Stick utilizados foram o acelerómetro e o giroscópio que captam informações detalhadas sobre os movimentos do utilizador:

1. Acelerómetro: Mede as acelerações lineares nos eixos X, Y e Z, permitindo avaliar deslocamentos, alterações de velocidade e padrões de movimento.

2. Giroscópio: Regista as taxas de rotação nos mesmos eixos, captando o pitch (inclinação vertical), roll (inclinação lateral) e yaw (rotação horizontal), o que permite determinar a orientação e direção do movimento.

A transmissão dos dados recolhidos pelo M5Stick é feita através da comunicação BLE (Bluetooth Low Energy), que liga o sensor ao servidor Raspberry Pi. Esta tecnologia é ideal para aplicações móveis, pois oferece uma conexão fiável e de baixo consumo energético. O processo inicia-se quando o utilizador escolhe a opção **Iniciar Leitura de Dados com o M5Stick** na interface gráfica, o que, através de um subprocesso, ativa o sensor para recolher e enviar dados de movimento em tempo real para o Raspberry Pi. Esta integração permite uma comunicação fluida e eficiente, garantindo que os dados são recolhidos com precisão e transmitidos sem perdas.

2.3 Raspberry Pi

O Raspberry Pi atua como o núcleo de processamento e coordenação do sistema, integrando e processando os dados recebidos do sensor M5Stick. A sua função é essencial para transformar os dados brutos em informações úteis e apresentar os resultados ao utilizador.

1. Receção e Armazenamento de Dados: Após a transmissão via BLE, o Raspberry Pi organiza os dados de movimento recebidos, assegurando que estão prontos para análise. Este passo inclui a verificação de integridade dos dados e a gestão do armazenamento temporário.

2. Processamento com Machine Learning: O Raspberry Pi utiliza um modelo de machine learning previamente treinado para analisar os dados recebidos. Este modelo é capaz de classificar a atividade realizada com base nas características extraídas dos movimentos captados pelo sensor.

3. Comunicação com a Interface via MQTT: Uma vez processados, os resultados são enviados para a interface do utilizador através do protocolo MQTT. Esta comunicação bidirecional permite que os utilizadores visualizem os resultados em tempo real e interajam com o sistema.

A arquitetura edge baseada no Raspberry Pi proporciona uma plataforma eficiente e acessível para o processamento de dados, reduzindo a latência e tornando o sistema mais autónomo. Este design garante que todas as operações, desde a receção de dados até à apresentação dos resultados, são realizadas de forma coordenada e eficaz.

2.4 Aquisição de dados

No total, foram realizadas 168 aquisições, distribuídas entre as atividades de sentar (50 aquisições), levantar (50 aquisições) e andar (68 aquisições). Cada aquisição teve uma duração fixa de 5 segundos, iniciada manualmente através do botão A do sensor M5Stick.

Durante cada aquisição, os dados de movimento foram recolhidos pelo sensor M5Stick com uma taxa de amostragem contínua a cada 100 ms, o que corresponde a 10 amostras por segundo. Este intervalo permitiu uma cobertura detalhada dos movimentos, registando informações de aceleração e rotação nos três eixos. Após o término dos 5 segundos, o processo de aquisição parava automaticamente, e os dados eram armazenados num ficheiro CSV gerado atraves de um código em python auxiliar.

No final, o processo resultou em 168 ficheiros CSV, com cada ficheiro correspondendo a uma aquisição individual. Esta organização clara e estruturada dos dados permitiu a criação de um conjunto de treino robusto, representativo das diferentes atividades, facilitando as etapas subsequentes de análise e modelagem.

3 Machine Learning

As bibliotecas utilizadas no projeto foram: `pandas`, `tsfel` (Time Series Feature Extraction Library), `numpy`, `matplotlib`, `joblib`, `seaborn` e `sklearn`. A biblioteca `sklearn` foi especialmente utilizada para tarefas como a criação da matriz de confusão, divisão do conjunto de dados em treino e teste, cálculo da *accuracy*, bem como para os métodos `classification_report` e `cross_val_score`.

3.1 Extração de Features

Após a recolha dos dados, que gerou um total de 168 ficheiros CSV correspondentes às medições realizadas, estes foram organizados em três pastas distintas, separadas pelo tipo de movimento. Foi então desenvolvido um código Python que processou cada ficheiro em cada pasta, extraindo 900 features de cada aquisição. Esse processo resultou na criação de 168 novos ficheiros CSV, correspondentes às features extraídas de cada movimento.

Posteriormente, outro código foi utilizado para consolidar todas essas features num único ficheiro denominado `allfeatures.csv`, agregando os dados das 168 aquisições num formato estruturado e unificado.

3.2 Dataset de Treino e Escolha do Classificador

Para a criação do dataset de treino, foi gerado um ficheiro CSV contendo as features de 40 amostras de cada movimento. Este ficheiro foi carregado no software *Orange*, seguindo o esquema apresentado na Figura 3.1.

Após a análise das features no *Orange*, foi realizado um ranking das melhores, resultando na seleção das 10 mais relevantes. Esas features são importantes porque refletem padrões estatísticos e de frequência que são cruciais para identificar a diferença entre os movimentos. Por exemplo a

5_Autocorrelation é importante porque movimentos repetitivos, como "andar", tendem a ter uma alta autocorrelação, porque o padrão do acelerometro e do giroscópio repete-se ao longo do tempo. Enquanto que, o "sentar" e "levantar" podem ter padrões menos repetitivos. Outra feature a realçar é a *5_Median frequency*, que destaca o ponto no qual a potência espectral total é dividida igualmente, ou seja, metade do sinal está a baixo e a outra metade está a cima desta referência. Por exemplo, o movimento de "andar", à partida, tem uma frequência mediana mais baixa e contínua, enquanto que o de "levantar" e "sentar" apresentam frequências mais altas, com maior número de variações. Figura 3.2

Com as melhores features definidas, o próximo passo foi determinar o classificador mais adequado para o modelo. Para isso, foi realizada uma cross validation, considerando a *accuracy* como métrica principal. Os resultados obtidos são apresentados na Figura 3.3, onde o *SVM* foi identificado como o classificador mais eficiente para o nosso conjunto de treino.

De seguida fez-se a matriz de confusão para avaliar o desempenho do modelo de classificação, especialmente por ser uma classificação de multi-classes. Obtendo apenas três valores classificados incorretamente.

Contudo, quando aplicámos o classificador SVM no nosso codigo python para fazer o predict da class, obtivemos demasiados erros inesperados, o movimento de sentar estava a ser maioritariamente assumido como levantar. O que nos levou a usar o classificador Random Forest, obtendo uma proporção de movimentos classificados corretamente muito maior. Figura 3.4

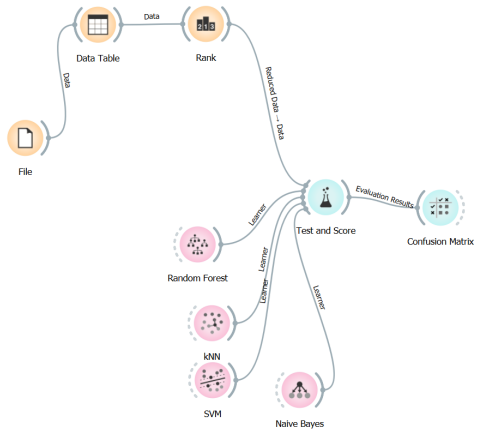


Figure 3.1: Esquema de processamento no *Orange* para preparação do dataset.

		#	Info. gain	Gain ratio	Gini	ANOVA	χ^2	ReliefF	FCBF
1	1_ECDF Percentile_1		0.967	0.488	0.414	138.578	66.528	0.155	1.183
2	1_Centroid		0.939	0.470	0.397	147.800	72.100	0.200	1.101
3	1_Slope		0.895	0.447	0.387	133.185	67.600	0.199	0.000
4	5_Autocorrelation		0.872	0.447	0.333	112.740	73.755	0.198	0.972
5	0_Power bandwidth		0.815	0.417	0.313	656.175	47.156	0.338	0.000
6	5_Median frequency		0.806	0.410	0.319	226.206	57.634	0.231	0.000
7	0_Human range energy		0.805	0.402	0.323	127.089	67.233	0.386	0.000
8	5_Median absolute diff		0.803	0.401	0.318	278.912	69.033	0.277	0.811
9	1_Median absolute diff		0.795	0.424	0.301	197.455	53.556	0.259	0.850
10	0_Mean diff		0.784	0.392	0.336	48.501	46.033	0.152	0.778

Figure 3.2: As 10 melhores features selecionadas através do ranking no *Orange*.

Model	AUC	CA	F1	Prec	Recall	MCC
SVM	0.991	0.975	0.975	0.975	0.975	0.963
Naive Bayes	0.990	0.975	0.975	0.975	0.975	0.963
Random Forest	0.982	0.967	0.967	0.967	0.967	0.950
kNN	0.880	0.792	0.790	0.796	0.792	0.691

Figure 3.3: Comparação dos classificadores com base na *accuracy*, destacando o *SVM* como o melhor.

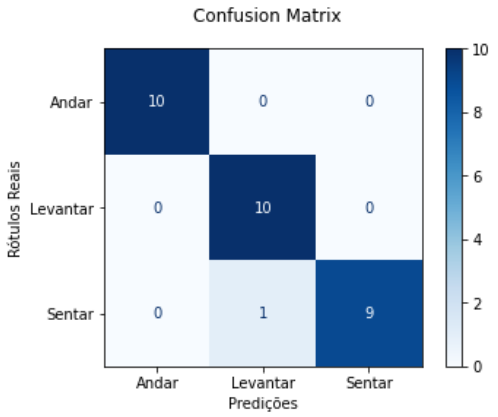


Figure 3.4: Matriz confusão

4 Interface do Utilizador

4.1 Streamlit

A interface do utilizador foi desenvolvida com o objetivo de oferecer uma experiência intuitiva e eficiente para monitorizar e classificar movimentos humanos em tempo real. Construída em Streamlit, a aplicação organiza as suas funcionalidades em diferentes secções, apresentadas de forma clara e acessível ao utilizador. Figura 4.1

Na página inicial, é apresentado o título do projeto, juntamente com uma breve descrição dos seus objetivos. Esta introdução explica que o sistema foi concebido para identificar automaticamente três tipos de movimentos —andar, sentar e levantar— com base nos dados recolhidos por um sensor M5 Stick posicionado junto ao peito do utilizador. Adicionalmente, o utilizador é orientado sobre como utilizar o dispositivo e interagir com a aplicação.

Figure 4.1: Páginal Inicial da Interface

Uma das funcionalidades centrais da interface é o botão **"Iniciar Leitura de Dados no M5Stick"**, que permite ao utilizador começar a recolher dados diretamente do sensor. Este botão ativa um script responsável pela aquisição em tempo real e pelo armazenamento dos dados capturados num ficheiro CSV, o qual será posteriormente utilizado para análise e classificação.

Após a recolha de dados, o utilizador pode aceder à funcionalidade de **"Classificação de Movimentos"**. Aqui, um botão dedicado permite carregar o último ficheiro CSV gerado e submetê-lo ao modelo de Machine Learning previamente treinado. O modelo analisa as medições do acelerómetro e giroscópio (gyroX, gyroY, gyroZ, accX, accY e accZ) para prever o tipo de movimento realizado. Após a classificação, o sistema exibe a previsão diretamente na interface, complementada por uma representação visual em forma de GIF animado, que ilustra o movimento identificado. Figura 4.2

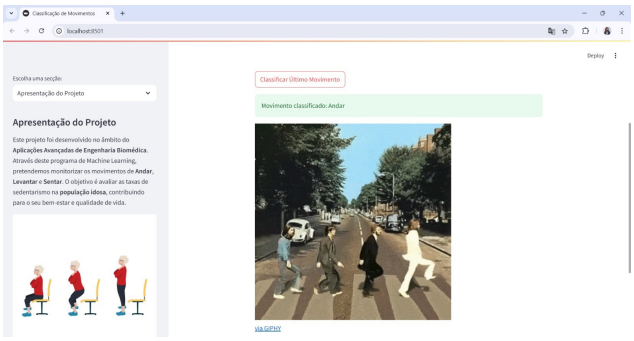
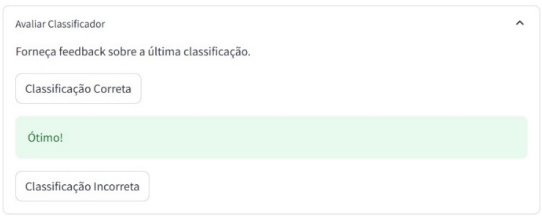
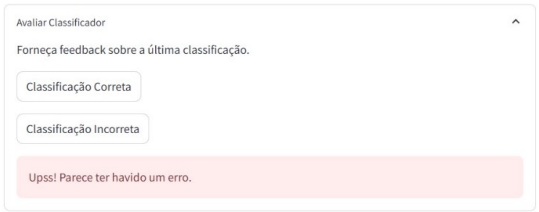


Figure 4.2: Interface após classificação do último movimento

Uma funcionalidade adicional importante é a capacidade do utilizador fornecer feedback sobre as predições. Após visualizar os resultados, o utilizador pode indicar se a classificação foi correta ou incorreta, utilizando um botão apropriado. Este feedback é registado no sistema, podendo ser utilizado para futuras análises e ajustes do modelo, melhorando assim o desempenho geral. Figura 4.3



(a) Classificação Correta



(b) Classificação Incorreta

Figure 4.3: Comparação entre classificação correta e incorreta.

Para promover maior transparência e utilidade, a interface também apresenta um histórico de

classificações, onde o utilizador pode consultar os movimentos registados ao longo do tempo. Este histórico é exibido numa tabela interativa que inclui informações como o nome do ficheiro analisado, o movimento previsto, o horário da classificação e o feedback associado. Figura 4.4

Além disso, uma funcionalidade específica calcula o tempo acumulado que o utilizador passou em cada tipo de movimento. Os resultados são apresentados em gráficos de barras, proporcionando uma visão clara e detalhada da distribuição de atividades. Figura 4.5

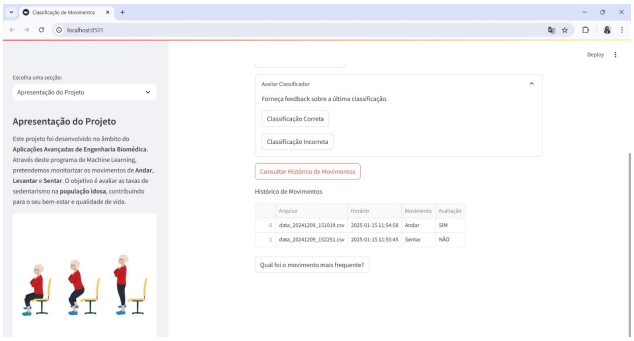
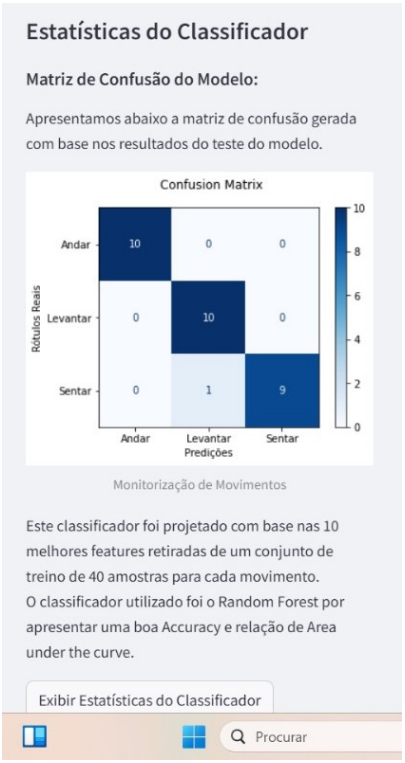


Figure 4.4: Histórico de movimentos

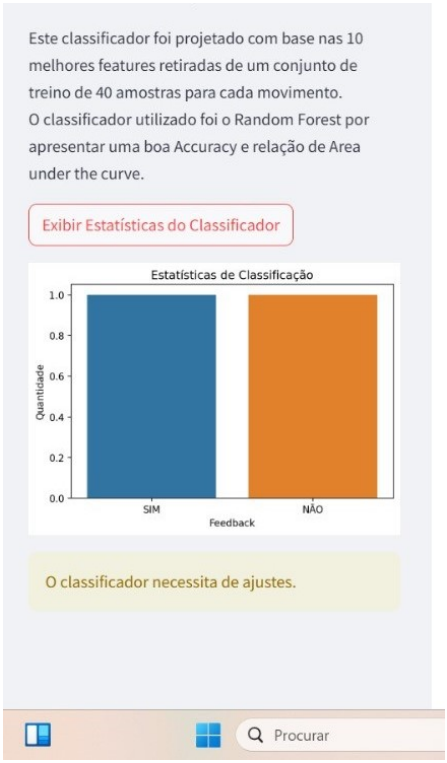
Figure 4.5: Tempo por movimento com indicação do movimento mais frequente

No que respeita às estatísticas do sistema, a interface inclui várias ferramentas visuais para monitorizar o desempenho do modelo. Um gráfico de barras exibe o número de previsões corretas e incorretas com base no feedback do utilizador, enquanto que uma matriz de confusão, gerada durante os testes do modelo, é apresentada na barra lateral oferecendo uma análise detalhada do desempenho em cada classe, evidenciando padrões de erro e pontos de melhoria. O código está formatado para que se o número de classificações incorretas for superior ou igual ao número de

classificações corretas, uma mensagem apareça a recomendar o ajuste do classificador. Figura 4.6



(a) Matriz de Confusão



(b) Precisão do Classificador

Figure 4.6: Representação da matriz de confusão e do estado do classificador.

Também na barra lateral, estão disponíveis secções informativas, como uma breve descrição do projeto, com a explicação do seu propósito e aplicação prática, e uma área dedicada aos créditos, onde se destacam as alunas responsáveis pelo desenvolvimento e os professores orientadores.

O sistema foi projetado para suportar a comunicação através do protocolo MQTT, permitindo a sua integração com dispositivos externos e a transmissão de dados para monitorização remota, garantindo a flexibilidade e a aplicação prática do sistema em diferentes cenários.

5 Discussão

Este trabalho apresenta um sistema para monitorização e classificação de movimentos utilizando o protocolo MQTT, o sensor M5Stick e o microcomputador Raspberry Pi. Embora os resultados tenham sido satisfatórios, algumas limitações foram identificadas, e várias melhorias podem ser implementadas no futuro para tornar o sistema mais robusto, eficiente e aplicável a diferentes cenários.

Uma das principais limitações do projeto reside nos dados utilizados para treinar o modelo de machine learning. Durante o processo de treino, os dados foram adquiridos exclusivamente por uma única pessoa, com o sensor M5Stick sempre na mesma posição e no mesmo ambiente. Além disso, os movimentos de sentar e levantar foram executados de forma padronizada e repetitiva. Este cenário restringiu a diversidade do conjunto de treino, o que pode comprometer a capacidade do modelo de generalizar para novos utilizadores ou para condições diferentes das apresentadas no treino. Por exemplo, pessoas com diferentes características físicas ou formas de executar os movimentos podem não ser corretamente classificadas pelo sistema.

Outro ponto a destacar é a ausência de uma classe de rejeição no modelo de machine learning. Atualmente, o sistema está configurado para classificar todos os movimentos em uma das três categorias predefinidas: "Andar", "Sentar" ou "Levantar". Contudo, na prática, o utilizador pode realizar movimentos que não se enquadram nessas categorias, como inclinar-se ou realizar gestos com os braços. Nesses casos, o modelo pode produzir uma classificação errada, reduzindo a confiabilidade do sistema. A introdução de uma classe de rejeição permitiria ao sistema identificar movimentos desconhecidos, aumentando a precisão em situações reais.

Por fim, o sistema atualmente permite ao utilizador fornecer feedback sobre a precisão das classificações, mas este feedback não é utilizado para melhorar o modelo de forma dinâmica. A ausência de aprendizagem contínua limita a capacidade do sistema de se adaptar e melhorar ao longo do tempo com base nos erros identificados durante a utilização prática. Contudo isto pode ser encarado como uma prespetiva futura para criar um modelo melhor.

6 Conclusão

O presente trabalho resultou no desenvolvimento de um sistema inovador para a monitorização e classificação de movimentos humanos, combinando tecnologias de IoT, machine learning e comunicação MQTT. Este sistema utiliza o sensor M5Stick para capturar dados inerciais do utilizador, o Raspberry Pi como unidade central de processamento e uma interface em Streamlit para exibição dos resultados ao utilizador de forma intuitiva e acessível.

O protocolo MQTT desempenhou um papel fundamental na comunicação em tempo real entre os dispositivos, assegurando que as previsões realizadas pelo Raspberry Pi fossem transmitidas eficientemente para a interface de utilizador. Adicionalmente, a implementação do modelo de Random Forest, treinado com as features mais relevantes extraídas por meio da biblioteca TSFEL, permitiu uma classificação precisa dos movimentos "Andar", "Sentar" e "Levantar".

Apesar do sucesso alcançado, algumas limitações foram identificadas, como a pouca diversidade nos dados de treino, que pode comprometer a generalização do modelo para diferentes utilizadores e cenários. Além disso, a ausência de uma classe de rejeição para movimentos desconhecidos limita a robustez do sistema em contextos reais. Contudo, foram implementadas funcionalidades extra consideradas relevantes para o propósito do nosso trabalho, pelo que apresentamos este projeto com bastante satisfação e sensação de trabalho bem conseguido, sendo que foi a primeira vez que trabalhamos com o sistema integrado neste projeto.

Em suma, este trabalho demonstra a viabilidade de um sistema integrado, com combinação de sensores compactos, processamento eficiente e tecnologias de comunicação em tempo real para monitorização de movimentos, com potencial de aplicação em contextos como a saúde e o bem-estar de populações idosas, podendo ser alargada a outras aplicações e contextos.