

UNIVERSIDADE FEDERAL DE ITAJUBÁ

GUILHERME FERNANDES DE OLIVEIRA

PEDRO COELHO TAGLIAFERRO

TIAGO MOURA MONTEIRO

**DETECÇÃO DE QUEDAS DE PESSOAS UTILIZANDO TÉCNICAS DE MACHINE
LEARNING PARA DISPOSITIVOS IOT EMBARCADOS**

ITAJUBÁ

2024

GUILHERME FERNANDES DE OLIVEIRA - 2021005067

PEDRO COELHO TAGLIAFERRO - 2021007929

TIAGO MOURA MONTEIRO - 2022015569

**DETECÇÃO DE QUEDAS DE PESSOAS UTILIZANDO TÉCNICAS DE MACHINE
LEARNING PARA DISPOSITIVOS IOT EMBARCADOS**

Relatório submetido ao Professor Marcelo José
Rovai como requisito parcial para aprovação na
disciplina IESTI01 – TINYML –
APRENDIZADO DE MÁQUINA APLICADO
PARA DISPOSITIVOS IOT EMBARCADOS,
do curso de graduação em Engenharia
Eletrônica da Universidade Federal de Itajubá.

ITAJUBÁ

2024

SUMÁRIO

| | | |
|------------|--------------------------------|-----------|
| 1 | INTRODUÇÃO | 4 |
| 2 | OBJETIVO | 5 |
| 3 | HARDWARE | 6 |
| 4 | SOFTWARE | 7 |
| 5 | COLETA DE DADOS..... | 8 |
| 6 | PRÉ-PROCESSAMENTO | 9 |
| 7 | DESIGN DO MODELO..... | 11 |
| 7.1 | Primeiro Modelo | 11 |
| 7.2 | Segundo Modelo | 12 |
| 7.3 | Terceiro Modelo | 14 |
| 8 | RESULTADOS | 15 |
| 8.1 | Primeiro Modelo | 15 |
| 8.2 | Segundo Modelo | 15 |
| 8.3 | Terceiro Modelo | 16 |
| 9 | DEPLOY | 17 |
| 10 | SUGESTÕES FUTURAS | 19 |
| 11 | CONCLUSÃO..... | 20 |

1 INTRODUÇÃO

Quase metade dos idosos com 80 anos ou mais sofrem quedas, e isso tem sido uma crescente na última década. Esse aumento não se deve apenas ao maior acesso à saúde pública, mas também ao envelhecimento populacional, algo já previsto pela teoria da transição demográfica, que impacta as políticas públicas em diversos países.

Um estudo de caso realizado entre 2005 e 2010 evidenciou os altos custos associados a esses eventos, com gastos superiores a milhões de reais. Essa realidade destaca a necessidade de prevenir e controlar quedas entre idosos.

Investimentos em pesquisa têm sido direcionados para o desenvolvimento de dispositivos de detecção de quedas. Por exemplo, em 2017, o Google financiou uma pesquisa da USP e da UFSCar para evitar quedas de idosos utilizando tecnologias avançadas. Em 2022, um pesquisador da Universidade Federal do Rio Grande do Sul criou um dispositivo usando essas mesmas tecnologias. Ambos os projetos, separados por cinco anos, demonstram a relevância e o contínuo desenvolvimento nesta área.

2 OBJETIVO

Detectar quedas de pessoas e disparar um alarme sonoro, como também classificar movimentos de sentar-se, andar e parar. Utilizando-se do Kit Tiny Machine Learning.

3 HARDWARE

O Hardware utilizado foi o Kit de desenvolvimento de TINYML, consistindo em um Arduino nano 33 BLE Sense, um TINY Machine Learning Shield, um cabo USB tipo A e um módulo de câmera OV7675 que estava disponível, porém não se faz necessário o uso para o projeto. Sendo assim, utilizamos o acelerômetro propriamente da placa Arduino nano 33 BLE Sense para capturar os dados de movimentação.



Figura 1: Kit TINYML

4 SOFTWARE

O Software utilizado para a criação do modelo, captura de dados e treinamento, foi o Edge Impulse com a implementação sendo feita via biblioteca Arduino através do Software Arduino IDE 2.3.2.

Para coletar os dados e integrá-los ao Edge Impulse, foi necessário seguir as etapas de instalação no próprio site da Edge Impulse, no qual solicita a instalação do Arduino CLI.

Vale salientar que para utilizar o Arduino IDE nesta aplicação também se fez necessário instalar a biblioteca da placa (Arduino nano 33 BLE).

5 COLETA DE DADOS

Para o dataset foram coletadas 79 amostras de treinamento e 27 amostras de teste, totalizando 75% de treinamento e 25% para teste. Assim dentro das 79 amostras de treinamento, temos 19 para a classe andando, 16 para a classe parado, 28 para a classe queda e 16 para a classe sentando-se. Já para as 27 amostras de testes, temos 6 para andando, 6 para parado, 9 para queda e 6 para sentando-se.

Os dados foram coletados pelo sensor acelerômetro do Arduino via Edge Impulse. Com uma janela de 4000ms a uma frequência de 50Hz. Tendo as seguintes características do sinal.



Figura 2: Sinal acelerômetro parado



Figura 3: Sinal acelerômetro andando

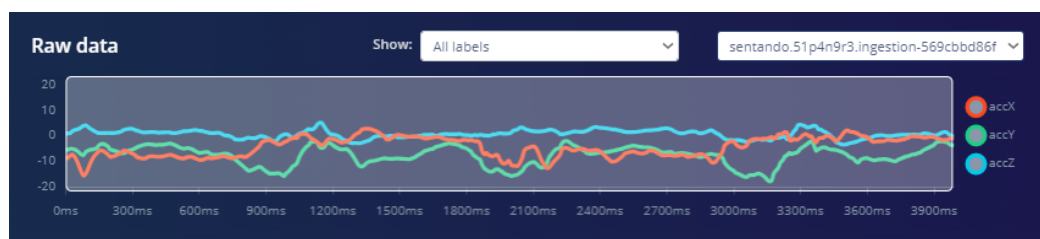


Figura 4: Sinal acelerômetro sentando-se



Figura 5: Sinal acelerômetro queda

6 PRÉ-PROCESSAMENTO

O pré-processamento utilizado em todos os modelos com objetivo de filtrar os dados do acelerômetro foi a análise espectral. As configurações utilizadas em cada modelo estão presentes nas imagens a seguir, visto que foram necessários 3 modelos para se obter uma melhor separação de dados e se aproximar do objetivo do projeto.

Parameters Autotune parameters

Filter

Scale axes ⓘ 1

Input decimation ratio ⓘ 1

Type ⓘ none

Analysis

Type ⓘ FFT

FFT length ⓘ 16

Take log of spectrum? ⓘ ☒

Overlap FFT frames? ⓘ ☒

Improve low frequency resolution? ⓘ ☐

Figura 6: Parâmetros para o 1ª modelo

Parameters Autotune parameters

Filter

Scale axes ⓘ 1

Input decimation ratio ⓘ 1

Type ⓘ none

Analysis

Type ⓘ FFT

FFT length ⓘ 64

Take log of spectrum? ⓘ ☒

Overlap FFT frames? ⓘ ☒

Improve low frequency resolution? ⓘ ☐

Figura 7: Parâmetros para o 2ª Modelo

Parameters

Autotune parameters

Filter

Scale axes ⓘ

2

Input decimation ratio ⓘ

1

Type ⓘ

low

Cut-off frequency ⓘ

8

Order ⓘ

6

Analysis

Type ⓘ

FFT

FFT length ⓘ

16

Take log of spectrum? ⓘ

☒

Overlap FFT frames? ⓘ

☒

Improve low frequency resolution? ⓘ

☒

Figura 8: Parâmetros para o 3ª Modelo

7 DESIGN DO MODELO

7.1 Primeiro Modelo

O primeiro modelo implementado tinha como objetivo analisar o comportamento inicial do sistema. Os dados foram coletados com o próprio kit de desenvolvimento, sem muita atenção à qualidade das amostras. Nesta etapa também definimos os labels que seriam usadas em outros modelos.

Todos os parâmetros utilizados foram o padrão do Edge Impulse a fim de observar a resposta sem nenhum tipo de ajuste. Isto resultou em uma certa confusão de dados, principalmente na confusão dos estados “andando” e “sentando”.

As imagens abaixo mostram os parâmetros usados no modelo:

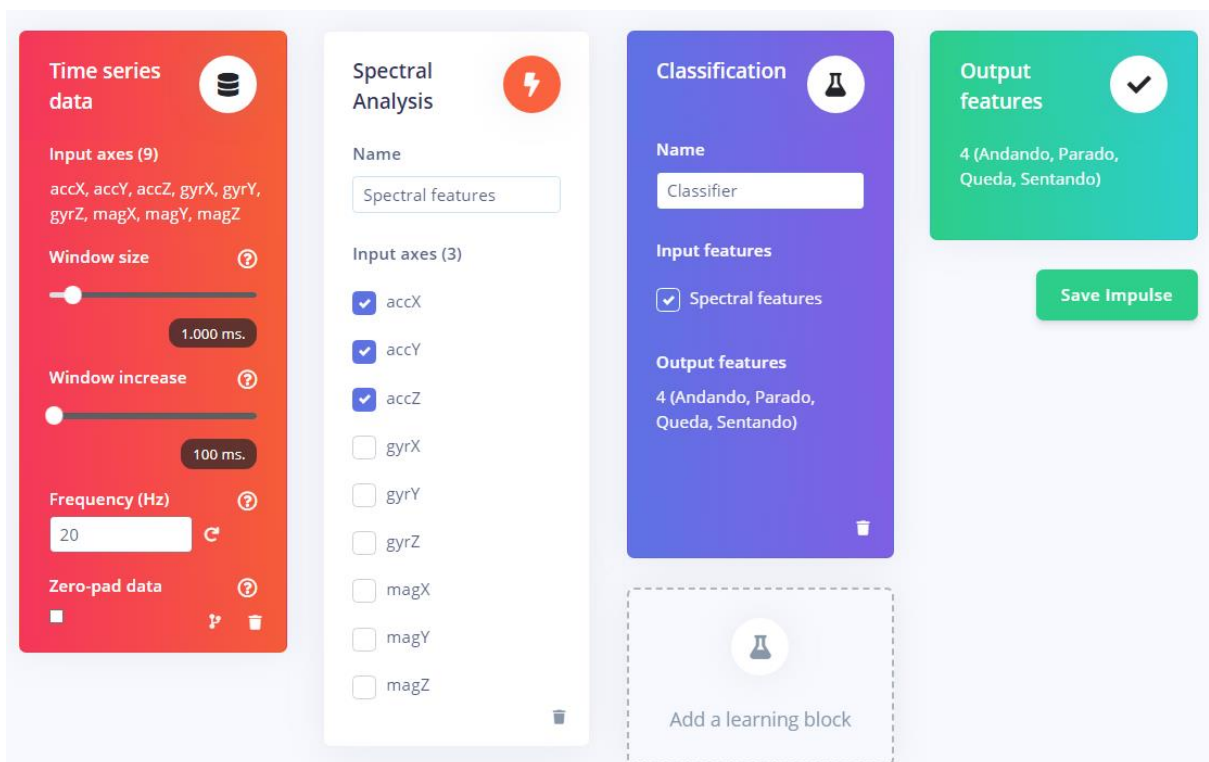


Figura 9: Criação do impulso modelo 1

Neural Network settings

Training settings

Number of training cycles ⓘ30

Use learned optimizer ⓘ

Learning rate ⓘ0.0005

Training processor ⓘCPU

Advanced training settings

Neural network architecture

Input layer (111 features)

Dense layer (20 neurons)

Dense layer (10 neurons)

Add an extra layer

Output layer (4 classes)

Figura 10: Rede neural modelo 1

7.2 Segundo Modelo

Pensando na fase de processamento de dados, surgiu o segundo modelo, nele alteramos os parâmetros da análise espectral para ver o comportamento, além de uma mudança na janela de captura dos dados.

O problema relacionado à janela de captura foi notado ao observarmos que alguns dados adquiridos tinham uma duração maior do que a janela de captura, o que pode gerar confusão, então colocamos 2000ms de janela (o dobro do modelo padrão testado anteriormente).

A outra mudança foi quanto à análise espectral, aumentamos a quantidade de pontos da FFT para 64 (o padrão é 16).

As imagens abaixo mostram os parâmetros usados no modelo:

The interface for creating the second model impulse consists of four main panels:

- Time series data:** Includes input axes (accX, accY, accZ, gyrX, gyrY, gyrZ, magX, magY, magZ), window size (2.000 ms), window increase (200 ms), frequency (20 Hz), and zero-pad data options.
- Spectral Analysis:** Includes a name field (Spectral features) and input axes (accX, accY, accZ, gyrX, gyrY, gyrZ, magX, magY, magZ).
- Classification:** Includes a name field (Classifier), input features (Spectral features), and output features (4 (Andando, Parado, Queda, Sentando)).
- Output features:** Includes a list of output features (4 (Andando, Parado, Queda, Sentando)) and a 'Save Impulse' button.

A dashed box at the bottom right indicates where to 'Add a learning block'.

Figura 11: Criação do impulso modelo 2

The 'Neural Network settings' interface is divided into several sections:

- Training settings:**
 - Number of training cycles: 30
 - Use learned optimizer: ☐
 - Learning rate: 0.0005
 - Training processor: CPU
- Advanced training settings:** (Collapsed)
- Neural network architecture:**
 - Input layer (111 features)
 - Dense layer (20 neurons)
 - Dense layer (10 neurons)
 - Add an extra layer
 - Output layer (4 classes)

Figura 12: Rede neural modelo 2

7.3 Terceiro Modelo

O segundo modelo foi implementado e ele nos indicou que nosso dataset precisava de refinamento, para se aproximar da movimentação cotidiana. No terceiro modelo alteramos novamente os dados espectrais e melhoramos os dados do dataset.

Primeiramente, retiramos os dados inconsistentes e com menos de 4 segundos, adicionando outros no lugar, fazendo assim uma base mais confiável para montar o modelo.

Também aumentamos ainda mais a janela de dados, indo para 4000 ms.

Além disso, percebemos que o comprimento da análise espectral não foi um fator decisivo com este conjunto de dados, mas um filtro passa-baixa de sexta ordem em 8 Hz foi fundamental para melhorar a classificação.

As imagens abaixo mostram os parâmetros usados no modelo

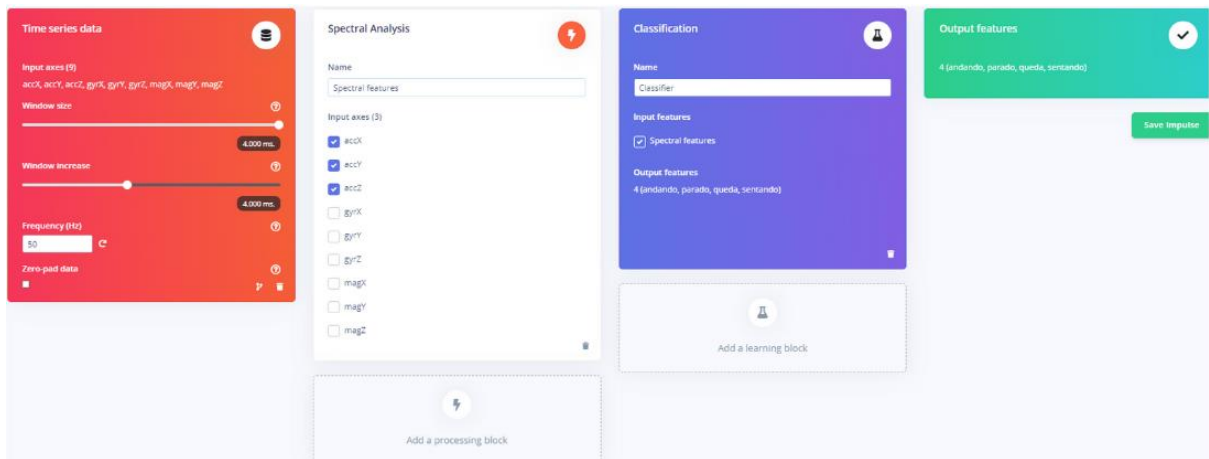


Figura 13: Criação do impulso modelo 3

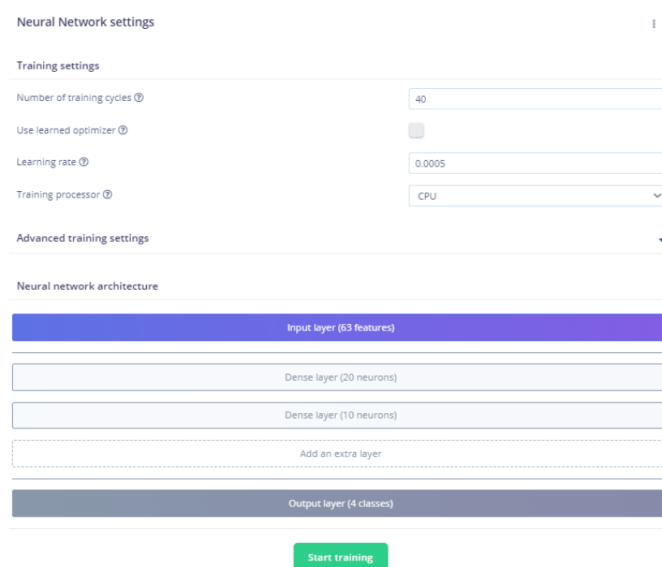


Figura 14: Rede neural modelo 3

8 RESULTADOS

8.1 Primeiro Modelo

Quanto à performance, o primeiro modelo performou de forma relativamente satisfatória (77% de acurácia), no entanto, havia muita margem para melhoria, principalmente na parte de processamento de dados e aquisição. Como queríamos observar o que poderíamos melhorar, este modelo não foi implementado na placa de desenvolvimento.



Figura 15: Matriz de confusão do modelo 1

8.2 Segundo Modelo

As mudanças realizadas no segundo modelo surtiram efeito positivo, apesar de ainda existir confusão, este segundo modelo atingiu uma performance superior em relação ao anterior (87% de acurácia), com isso resolvemos fazer a implementação de fato. Ao realizar este primeiro teste, foi possível perceber que a utilização não se mostrou prática, com o resultado sendo incerto na maior parte do tempo.



Figura 16: Matriz de confusão do modelo 2

8.3 Terceiro Modelo

O terceiro modelo nos apresentou um resultado excelente após o treinamento e classificação (93% de acurácia). Seu *deploy* apresentou o resultado que esperávamos, com a única ressalva sendo a inferência feita em intervalos e não em tempo contínuo.

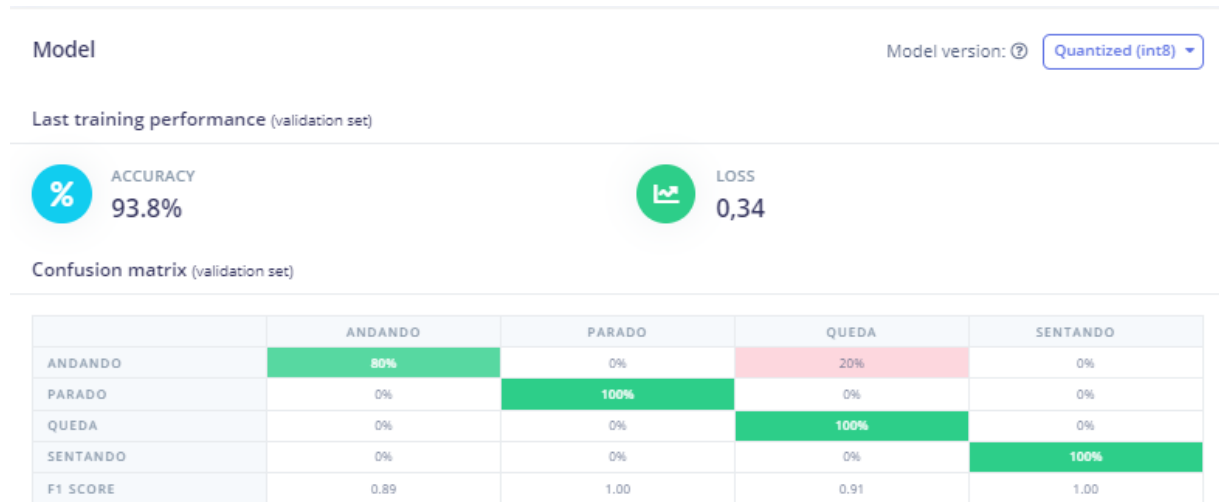



Figura 17: Matriz de confusão do modelo 3

9 DEPLOY

Para realizar a inferência no mundo real, foi utilizado a biblioteca que o Edge Impulse gera automático para deploy no Arduino Nano 33 BLE Sense, convertendo o projeto do Edge Impulse em Arduino. Assim instalamos a biblioteca e obtemos o código para realizar os testes.

Foi adicionado uma função na qual a partir do resultado da classificação é tomado uma ação. Sendo assim para a classe “andando” escolheu-se acender o led verde, para a classe “parado” escolheu-se acender o led azul e builtin, para a classe “queda” escolheu-se disparar um alarme sonoro por meio de um buzzer externo, ligado ao pino D11 e para a classe “sentando” escolheu-se acender o led vermelho.

O resultado obtido do deploy é mostrado pelo Edge Impulse, como mostra a imagem a seguir:



EON™ Compiler
Same accuracy, 54% less RAM, 57% less ROM.

▼

Quantized (int8)

Selected ✓

| | SPECTRAL FEATURES | CLASSIFIER | TOTAL |
|----------|-------------------|------------|--------|
| LATENCY | 49 ms. | 1 ms. | 50 ms. |
| RAM | 3,6K | 1,4K | 3,6K |
| FLASH | - | 16,4K | - |
| ACCURACY | | | - |

Figura 18: Resultado do Deploy

Para inferência, a placa foi presa como um relógio, como mostrar a imagem a seguir:

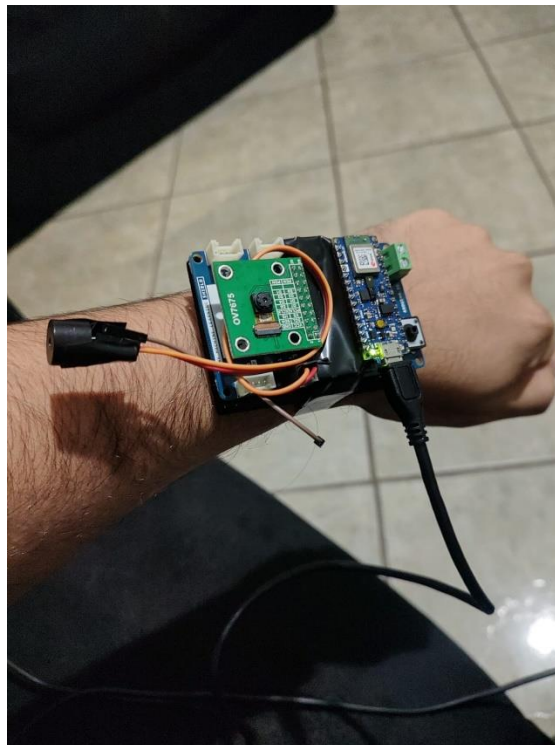


Figura 19: Montagem da inferência

Também foi possível obter o resultado de cada inferência via terminal

```
Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 113 ms., Classification: 0 ms., Anomaly: 0 ms.):
    andando: 0.83203
    parado: 0.13672
    queda: 0.02344
    sentando: 0.01172
Prediction: andando with probability 0.83
```

Figura 20: Usuário andando

```
Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 113 ms., Classification: 0 ms., Anomaly: 0 ms.):
    andando: 0.03906
    parado: 0.00781
    queda: 0.23828
    sentando: 0.71484
Prediction: sentando with probability 0.71
```

Figura 21: Usuário sentando-se

```
Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 113 ms., Classification: 0 ms., Anomaly: 0 ms.):
    andando: 0.00000
    parado: 0.98828
    queda: 0.01172
    sentando: 0.00000
Prediction: parado with probability 0.99
```

Figura 22: Usuário parado

```
Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 113 ms., Classification: 0 ms., Anomaly: 0 ms.):
    andando: 0.00781
    parado: 0.01172
    queda: 0.94141
    sentando: 0.03906
Prediction: queda with probability 0.94
```

Figura 23: Usuário caiu

Por fim, foi feita uma simulação da inferência, na qual se encontra no anexo 2 do documento.

10 SUGESTÕES FUTURAS

- Mais dados para compor o dataset, na qual resultaria em maior precisão na classificação;
- Melhorar a qualidade do dataset, utilizar outros acelerômetros para entender como se comporta o sinal ou realizar eventos de forma mais precisa, a fim de analisar características específicas de cada evento;
- Classificação contínua, dessa forma deve ser possível captar os eventos independente do tempo da janela de classificação.
- Substituir alarme por uma funcionalidade de maior impacto, no projeto foi utilizado um Buzzer apenas para validação, porém pode se implementar uma ligação a alguém ou envio das coordenadas.
- Adicionar detecção de anomalias, vale levantar a possibilidade de apenas detectar a queda do usuário, independente dos movimentos realizados e evitar a classificação de parado, andando e sentando-se.

11 CONCLUSÃO

A crescente população de idosos e os altos custos públicos relacionados à quedas neste grupo populacional nos mostra a necessidade de uma solução para remediar os efeitos de tais eventos. O projeto atingiu o objetivo de detectar quedas e emitir um sinal sonoro, com a classificação de movimento utilizando o kit Arduino de TINYML. Com a coleta de dados, pré-processamento e refinamento do modelo atingimos uma eficácia significativa em sua função. Esse desenvolvimento revela um potencial na área de TINYML e detecção de quedas, com possibilidades de expansão e aprimoramento do projeto.

REFERÊNCIAS

1. EDGE IMPULSE. Disponível em: <https://edgeimpulse.com>. Acesso em: 10 jul. 2024.
2. BRASIL. Ministério da Saúde. Todos os anos, 40% dos idosos com 80 anos ou mais sofrem quedas. Disponível em: <https://www.gov.br/saude/pt-br/assuntos/noticias/2022/outubro/todos-os-anos-40-dos-idosos-com-80-anos-ou-mais-sofrem-quedas>. Acesso em: 10 jul. 2024.
3. CNN BRASIL. Brasil vive transição demográfica que impactará políticas públicas, diz professor. Disponível em: <https://www.cnnbrasil.com.br/nacional/brasil-vive-transicao-demografica-que-impactara-politicas-publicas-diz-professor/>. Acesso em: 10 jul. 2024.
4. G1 SÃO CARLOS E REGIÃO. Google financia pesquisa da USP e da UFSCar para evitar quedas de idosos. Disponível em: <https://g1.globo.com/sp/sao-carlos-regiao/noticia/google-financia-pesquisa-da-usp-e-da-ufscar-para-evitar-quedas-de-idosos.ghtml>. Acesso em: 10 jul. 2024.
5. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL (UFRGS). Pesquisador da UFRGS cria dispositivo vestível que detecta quedas. Disponível em: <https://www.ufrgs.br/ciencia/pesquisador-da-ufrgs-cria-dispositivo-vestivel-que-detecta-quedas/>. Acesso em: 10 jul. 2024.

ANEXOS

1. Código: <https://drive.google.com/file/d/1L-NdfwyaIOV3rS47JPt6asWv7SIoQgXw/view>
2. Inferência: https://drive.google.com/file/d/1McgvoFDAXB-4iBKco39dEh0kJk-OtvLC/view?usp=drive_link
3. Apresentação: <https://www.youtube.com/watch?v=7BsHzaYMCd8>