

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

José Luiz Poli Miola

**Desempenho de Modelos Inteligentes para
Classificação de Intenção de Movimento de Mão
a partir de sinais de sEMG no microcontrolador
ESP32**

Porto Alegre

2024

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

José Luiz Poli Miola

**Desempenho de Modelos Inteligentes para Classificação
de Intenção de Movimento de Mão a partir de sinais de
sEMG no microcontrolador ESP32**

Projeto de Diplomação II, apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para a obtenção do grau de Engenheiro Eletricista

UFRGS

Orientador: Prof. Dr. Tiago Weber

Porto Alegre

2024

José Luiz Poli Miola

Desempenho de Modelos Inteligentes para Classificação de Intenção de Movimento de Mão a partir de sinais de sEMG no microcontrolador ESP32

Projeto de Diplomação II, apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para a obtenção do grau de Engenheiro Eletricista

BANCA EXAMINADORA

Prof. Dr. Raphael Martins Brum
UFRGS

Prof. Dr. Maurício Cagliari Tosin
UFRGS

Prof. Dr. Tiago Weber
Orientador - UFRGS

Resumo

Próteses de mão controladas por sEMG são soluções não invasivas para amputados terem novamente a funcionalidade de seus membros, melhorando sua qualidade de vida. Controlar estas próteses exige a criação de um classificador capaz de não somente classificar estes movimentos, mas que seja possível ser embarcado. Este trabalho testou diferentes configurações de redes neurais, regressão logística e florestas aleatórias, variando hiperparâmetros, número de entradas e características para avaliar seus desempenhos e os testando em um microcontrolador Esp32. Foi utilizada a base de dados NinaPro para avaliação dos modelos, utilizando 18 movimentos discretos. Foi encontrado que o modelo com maior taxa de acerto foi o de floresta aleatória, utilizando todos os 12 canais com 78,0% de taxa de acerto. Os modelos com maiores taxas de acerto para redes neurais e regressões logísticas encontraram ambos 74,3%. Foi encontrado que as latências dos modelos aumentam conforme aumentam o número de entradas, com latências variado de 4897-5785 μs (redes neurais), 16-1127 μs (regressão logística) e 2-313 μs (florestas aleatórias).

Palavras-chave: sEMG; Microcontrolador; Movimentos;

Abstract

sEMG-controlled hand prostheses are non-invasive solutions for amputees to regain the functionality of their limbs, improving their quality of life. Controlling these prostheses requires the creation of a classifier capable of not only classifying these movements, but also being embeddable. This work tested different configurations of neural networks, logistic regression, and random forests, varying hyperparameters, the number of inputs, and features to evaluate their performances and testing them on an Esp32 microcontroller. The NinaPro database was used to evaluate the models, using 18 discrete movements. It was found that the model with the highest accuracy rate was the random forest, using all 12 channels with 78,0% accuracy rate. The models with the highest accuracy rates for neural networks and logistic regressions both found 74.3%. It was found that the latencies of the models increase as the number of inputs increases, with latencies ranging from 4897-5785 μs (neural networks), 16-1127 μs (logistic regression), and 2-313 μs (random forests).

Keywords: sEMG; Microcontroller; Movements;.

Lista de Figuras

Figura 1 – Eletrodos Delsys Trigno.	10
Figura 2 – Um contra todos para Regressão Logística	12
Figura 3 – Exemplo de uma árvore de decisões.	13
Figura 4 – Exemplo floresta aleatória.	14
Figura 5 – Neurônio.	15
Figura 6 – Rede Neural.	16
Figura 7 – Esquema da metodologia.	17
Figura 8 – Microcontrolador utilizado.	18
Figura 9 – Protocolo de Aquisição	19
Figura 10 – Movimentos feitos	20
Figura 11 – Janelamento do sinal de sEMG.	21
Figura 12 – Etapas da seleção de parâmetros e características.	23
Figura 13 – Representação da seleção sequencial de características.	24
Figura 14 – Representação do ordenamento dos canais.	25
Figura 15 – Taxa de acerto por número de características	29
Figura 16 – Posição dos eletrodos	30
Figura 17 – Taxa de acerto redes neurais por número de canais, avaliada para 40 indivíduos.	31
Figura 18 – Taxa de acerto florestas aleatórias por número de canais, avaliada para 40 indivíduos.	32
Figura 19 – Taxa de acerto regressão logística por número de canais, avaliada para 40 indivíduos.	32
Figura 20 – Matriz de confusão para o modelo com 12 canais de Rede Neural.	37
Figura 21 – Matriz de confusão para o modelo com 12 canais de Floresta Aleatória.	38
Figura 22 – Matriz de confusão para o modelo com 12 canais de Regressão Logística.	39
Figura 23 – Comparação entre modelos de floresta aleatória embarcados e modelos no computador	40
Figura 24 – Latência média dos modelos de Regressão Logística	41
Figura 25 – Latência média dos modelos de Florestas Aleatórias	41
Figura 26 – Latência média do cálculo do valor RMS por número de canais	42
Figura 27 – Latência média do cálculo do desvio padrão por número de canais	42
Figura 28 – Latência média do cálculo da variância por número de canais	43
Figura 29 – Latência média total das regressões logísticas	43
Figura 30 – Latência média total das redes neurais	44
Figura 31 – Latência média total das florestas aleatórias	44
Figura 32 – Memória ocupada pelos modelos	45

Lista de Tabelas

Tabela 1 – Versões de softwares utilizados	18
Tabela 2 – Hiperparâmetros Testados	23
Tabela 3 – Resultados da primeira otimização de hiperparâmetros	28
Tabela 4 – Características testadas	29
Tabela 5 – Características selecionadas	29
Tabela 6 – Resultados da segunda otimização de hiperparâmetros	30
Tabela 7 – Resultados do ordenamento de importância dos canais	30
Tabela 8 – Dados por classe para o modelo com 12 canais de Rede Neural	34
Tabela 9 – Dados por classe para o modelo com 12 canais de Floresta Aleatória	35
Tabela 10 – Dados por classe para o modelo com 12 canais de Regressão Logística	36
Tabela 11 – Taxas de acerto em outros trabalhos	46
Tabela 12 – Melhores taxas de acerto encontradas por modelo	46

Sumário

1	INTRODUÇÃO	8
1.1	Objetivo Geral	9
1.2	Objetivos específicos	9
2	FUNDAMENTAÇÃO TEÓRICA	10
2.1	Eletromiografia	10
2.2	Aprendizado de Máquina	10
2.2.1	Regressão Logística	11
2.2.2	Arvore de decisões e Floresta Aleatória	12
2.2.3	Redes Neurais	15
2.3	Aprendizado de Máquina em Sistemas Embarcados	16
3	METODOLOGIA	17
3.1	Materiais utilizados	17
3.2	Base de dados	18
3.3	Janelamento	20
3.4	Características	21
3.5	Seleção de parâmetros e características	22
3.5.1	Otimização dos hiperparâmetros	23
3.5.2	Seleção de Características	24
3.5.3	Avaliação da influência dos canais	24
3.6	Implementações dos Modelos em Sistema Embarcado	25
3.7	Avaliação dos modelos finais	26
3.7.1	Taxa de acerto	26
3.7.2	Latência	26
3.7.3	Ocupação de memória	27
4	RESULTADOS	28
4.1	Seleção de parâmetros e características	28
4.1.1	Otimização de hiperparâmetros	28
4.1.2	Seleção de características	28
4.1.3	Otimização de hiperparâmetros	30
4.1.4	Avaliação da influência dos canais	30
4.2	Avaliação dos modelos finais	31
4.2.1	Taxa de acerto	31
4.2.2	Latência	40
4.2.3	Ocupação de memória	44
4.3	Comparação com trabalhos da literatura	45
5	CONCLUSÕES	47
5.1	Trabalhos futuros	47
	REFERÊNCIAS BIBLIOGRÁFICAS	48

1 Introdução

Próteses de mão controladas por sEMG (Eletromiografia de superfície) são soluções não invasivas para amputados terem novamente a funcionalidade de seus membros, melhorando sua qualidade de vida. Muitas delas, porém, ainda tem sistemas de controle rudimentares, que exigem um longo treinamento do usuário (ATZORI *et al.*, 2014).

Atualmente a busca de formas de controle naturais são um tópico emergente (LIU, 2010), sendo uma de suas principais aplicações o controle de próteses para pacientes amputados. O uso de sinais de sEMG é uma alternativa não invasiva para uma interface entre pessoas e máquinas. Esta técnica tem se intensificado por trazer informações importantes sobre o movimento de membros superiores (TOSIN; MACHADO; BALBINOT, 2022).

Para o bom funcionamento das próteses é preciso desenvolver um método de classificação de movimentos adequado ao problema. Sinais de sEMG possuem forte natureza estocástica, o que torna necessário o desenvolvimento de um modelo inteligente para processá-los. A literatura mostra que já foram testadas diversas abordagens e modelos, com diferentes graus de sucesso, porém ainda há margem para melhoria.

O bom funcionamento da prótese não depende apenas da taxa de acerto do modelo. É muito importante, também, avaliar outras métricas de desempenho, como latência. Como o controlador da prótese precisa ser portátil, implementações em microcontroladores podem ser úteis ao unir desempenho e portabilidade.

O sinal de sEMG não pode ser usado diretamente por conta de sua natureza fortemente estocástica e de ser fruto da sinergia de diversos músculos (ANAM *et al.*, 2019). Uma forma tradicional de tratar o sinal é dividi-lo em janelas e a partir delas extrair características. Apesar de janelas maiores poderem aumentar a taxa de acerto do classificador, aplicações reais exigem um tempo de resposta de no máximo 300ms para serem fluidas (OSKOEI; HU, 2008).

Dentro de cada janela, várias características podem ser obtidas, tanto no domínio do tempo quanto no da frequência. Estudos já obtiveram bons resultados usando valor RMS (CRISWELL, 2011), Variância, Kurtosis, Frequência Mediana (CENE; BALBINOT, 2015) e Valor Médio Absoluto (OSKOEI; HU, 2008). Apenas aumentar o número de características, entretanto, não implica necessariamente um aumento na performance. Além de aumentar a carga computacional, características com alto correlação entre si pode ser consideradas redundantes e elimina-las pode aumentar a performance do modelo (ORTIZ-CATALAN; BRÅNEMARK; HÅKANSSON, 2013).

A grande maioria dos classificadores pesquisados atualmente dividem os movimentos em uma série de gestos discretos (TOSIN; MACHADO; BALBINOT, 2022). Já foram testados diversos modelos inteligentes com diferentes graus de sucesso, como SVM (BOSCHMANN *et al.*, 2009), Redes Neurais (ARIYANTO *et al.*, 2015), Regressão Logística (CENE; BALBINOT, 2015) e Árvores de decisão (JOSE *et al.*, 2017).

Com o avanço na área de aprendizagem de máquina, surgem novas ferramentas para trazer estes modelos para sistemas embarcados, como microcontroladores (SAHA; SANDHA; SRIVASTAVA, 2022). No cenário de uma prótese mecânica, é importante não somente uma boa taxa de acerto, mas também que sistema seja portátil o suficiente

para não atrapalhar o usuário (CENE; BALBINOT, 2019). Muitos dos estudos feitos levam em consideração que os classificadores operam em computadores com alto poder de processamento, que podem ser inconvenientes pela falta de portabilidade (QIN *et al.*, 2021).

Em (CENE; BALBINOT, 2019) foi utilizado uma *Extreme learning machine* embarcada em um Raspberry Pi para fazer classificação de movimentos, com acurácias variando de 90.9% a 77.2% para voluntários não amputados e 63.1% a 55.3% para voluntários amputados. Já (QIN *et al.*, 2021) embarcaram duas diferentes arquiteturas de Redes neurais convolucionais em um *Arduino Nano*, obtendo 89.4% e 98.9% de acurácia. (TAM *et al.*, 2020) também propõe uma Rede neural convolucional embarcada, com 98.9% de taxa de acerto.

1.1 Objetivo Geral

O objetivo deste trabalho é avaliar o desempenho de diferentes classificadores, implementados em microcontrolador para a classificação de movimentos de mão a partir de sinais de sEMG.

1.2 Objetivos específicos

- Criar software para treinar modelos e avaliar desempenho;
- Selecionar atributos para o modelo;
- Avaliar influencia dos canais de entrada, avaliando a possibilidade de reduzi-los sem perder taxa de acerto;
- Criar firmwares a serem usados no ESP32 para modelos treinados e avaliar;

2 Fundamentação Teórica

2.1 Eletromiografia

Os músculos humanos produzem sinais elétricos quando acionados. Estes sinais, chamados de sinais mioelétricos, são produzidos por pequenas correntes geradas por trocas de íons entre as membranas musculares (JAMAL, 2012). Estes sinais historicamente já foram utilizados para diversas áreas da medicina e biomecânica, como para o diagnóstico de doenças neuromusculares (PULLMAN *et al.*, 2000), detecção de movimentos e medida de força muscular (BENEDETTI *et al.*, 2001).

A eletromiografia se divide inicialmente com relação aos tipos de eletrodos utilizados. Existem os invasivos, como agulhas ou fios finos, que podem causar desconforto e lesões nos tecidos, e os de superfície. Os sinais intramusculares coletados por eletrodos invasivos geralmente apresentam melhor qualidade, enquanto os sinais mioelétricos coletados na superfície tendem a ter menor qualidade. No entanto, as vantagens práticas e clínicas dos eletrodos de superfície os tornam o foco principal de pesquisa e a escolha para este estudo (JAMAL, 2012).

Quando coletados por eletrodos de na superfície da pele os sinais são denominados de sEMG (Eletromiografia de superfície). Estes sinais detectados são da ordem de 10mV (-5mv a 5mv) com frequência de 5-450Hz. A Figura 1 mostra eletrodos do tipo Delsys Trigno.



Figura 1 – Eletrodos Delsys Trigno.
Fonte: (PIZZOLATO *et al.*, 2017).

2.2 Aprendizado de Máquina

Aprendizado de máquina é um ramo da inteligência artificial no qual sistemas são desenvolvidos para aprender e melhorar a partir de exemplos fornecidos. Isso é alcançado

por meio de uma variedade de algoritmos capazes de fazer previsões. Não há um algoritmo único capaz de resolver todos os problemas (MAHESH, 2020); a escolha do modelo a ser utilizado depende dos requisitos específicos do problema em questão.

A aprendizagem de máquina pode ser categorizada em três tipos principais: aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço. No aprendizado não supervisionado, o algoritmo é treinado com dados não rotulados, ou seja, sem uma classificação prévia. O modelo busca então identificar agrupamentos ou padrões nesses dados (MONARD; BARANAUSKAS, 2003).

Em contraste, no aprendizado por reforço, o algoritmo aprende a tomar decisões por meio de tentativa e erro. Ele interage com um sistema, recebendo como único *feedback* recompensas ou punições por suas ações, com o objetivo de maximizar uma recompensa cumulativa (WIERING; OTTERLO, 2012). Esse tipo de aprendizado é frequentemente empregado em cenários nos quais o algoritmo precisa tomar uma série de decisões interdependentes.

Aprendizado supervisionado utiliza dados rotulados para treinar modelos. Nesse tipo de aprendizado, o objetivo é desenvolver uma função que associa uma entrada a uma saída, utilizando pares de exemplos de entrada e saída (JAMES DANIELA WITTEN, 2013). Posteriormente, essa função é capaz de prever uma saída para novas entradas não presentes no conjunto de treinamento.

Modelos de aprendizado supervisionado podem também ser subdivididos em problemas de regressão e de classificação (GÉRON, 2022). Nos problemas de regressão, o objetivo é prever uma variável de saída contínua. Isso significa que a saída é um valor numérico que pode assumir qualquer valor dentro de um intervalo. Nos problemas de classificação, o objetivo é prever a categoria ou classe a qual um dado pertence. A variável de saída é categórica, o que significa que ela pertence a uma das classes pré-definidas.

Neste trabalho será utilizado modelos de aprendizado supervisionado, para resolver um problema de classificação. As próximas seções apresentam os modelos utilizados.

2.2.1 Regressão Logística

A Regressão Logística é um modelo estatístico que tem a capacidade de prever a probabilidade de ocorrência de um determinado evento. Este modelo utiliza a equação logística, conforme definida pela Equação 1, para calcular essa probabilidade. O modelo prevê a probabilidade P de um evento binário ocorrer, dado as entradas X , parametrizado pelos coeficientes β .

$$P(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n)}} \quad (1)$$

Existem $n + 1$ coeficientes β na equação, sendo n o número de entradas. A equação representa a probabilidade de uma entrada pertencer a uma classe.

O treinamento deste classificador consiste em encontrar os melhores valores para β , para que resulte na melhor classificação dado um conjunto de dados de entrada. Isso é feito, geralmente, utilizando algum método de otimização, como o gradiente descendente.

Como o modelo inicialmente é usado somente para casos binários, pode ser utilizada a técnica de *um contra todos* para realizar a classificação para k classes. Este método

consiste em treinar k diferentes classificadores, sendo cada um responsável por uma classe diferente. Isso é feito considerando, em cada modelo, uma classe como "1" e todas as outras como "0". A classe escolhida como a classificação final é a que resulta na maior probabilidade dentre as diferentes equações. Será preciso otimizar, portanto $(n + 1)(k)$ parâmetros β . A figura 2 ilustra o uso de *um contra todos*.

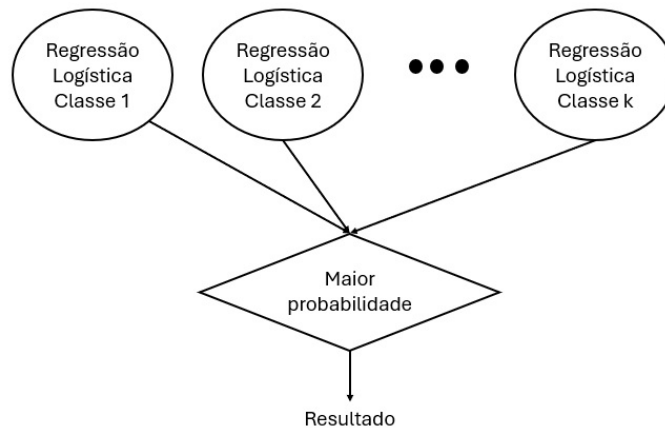


Figura 2 – Um contra todos para Regressão Logística

Fonte: O autor.

2.2.2 Árvore de decisões e Floresta Aleatória

A árvore de decisões é um algoritmo de aprendizagem de máquina supervisionado, não linear comumente utilizado para classificação. Este algoritmo classifica as entradas fazendo uma série de perguntas sobre as entradas. Cada pergunta está em um nó, e cada possível resposta aponta para um nó filho. A árvore é, assim, construída de forma hierárquica. Cada item é classificado a partir do nó mais acima, seguindo um caminho até chegar a um nó sem filhos, chamado de folha. Esta folha contém a classificação do item (KINGSFORD; SALZBERG, 2008).

A Figura 3 ilustra uma árvore de decisões hipotética. Nela as folhas estão em verde, mostrando as possíveis classificações em duas classes. Duas variáveis de entrada são levadas em conta, A e B, para realizar a classificação.

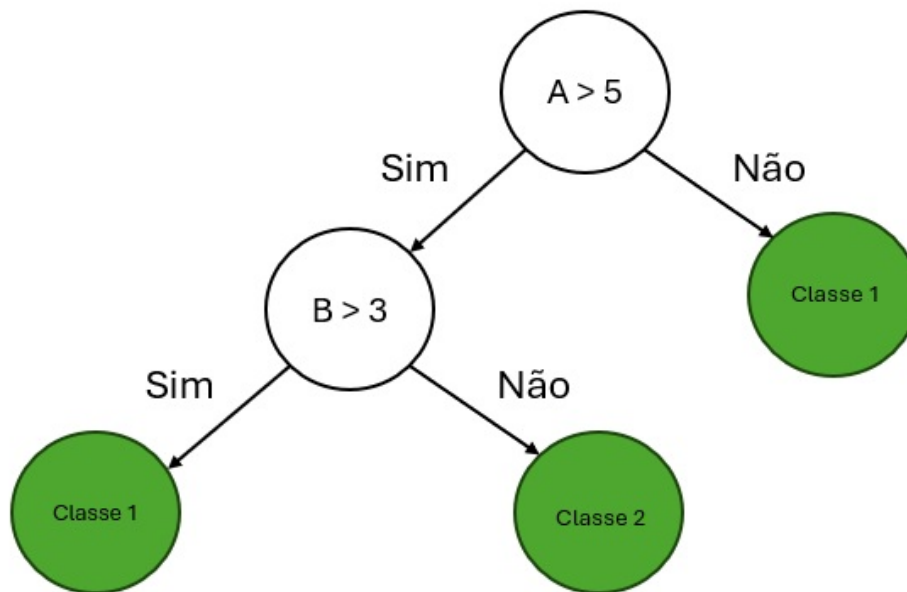


Figura 3 – Exemplo de uma árvore de decisões.

Fonte: O autor.

A árvore é construída de cima para baixo, dividindo sempre em dois o espaço das entradas. Este processo é repetido recursivamente para as duas novas subárvores que surgem, até que algum critério de parada seja atingido. Cada divisão é feita de forma que minimize uma equação escolhida como critério. As duas mais comuns são o índice Gini (Equação 2) e entropia (Equação 3) (JAMES DANIELA WITTEN, 2013).

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}) \quad (2)$$

$$D = - \sum_{k=1}^K \hat{p}_{mk} \log(\hat{p}_{mk}) \quad (3)$$

Em ambas as equações \hat{p}_{mk} representa a proporção das amostras de treinamento pertencentes a classe k na região m (região das possíveis amostras da subárvore em questão). Ambas as equações apresentam resultados similares, podendo ser interpretadas como *homogeneidade* das amostras do nó em questão.

Os critérios de parada para o crescimento da árvore, assim como equação escolhida pra guiar a divisão, são hiperparâmetros do modelo, definidos antes do início do treinamento. Alguns critérios de parada comum são (ROKACH; MAIMON, 2005):

- profundidade máxima da árvore foi atingida
- todas as amostras de um nó são de uma mesma categoria
- a função usada como critério para divisão não pode ser reduzida além de um valor pré-definido

- o número de amostras em um nó atinge um valor mínimo

Árvores de decisão são métodos que possuem, de forma intrínseca, uma alta variância. Isso implica que, ao dividir um conjunto de treinamento em diversas partes, cada subdivisão pode levar a resultados significativamente distintos. Uma estratégia eficaz para reduzir essa variância e aprimorar a acurácia consiste em construir várias árvores de decisão, ao invés de uma única. O método conhecido como Floresta Aleatória é um exemplo típico dessa abordagem.

Para desenvolver diversas árvores de decisão distintas, geramos subconjuntos de treinamento variados a partir do conjunto de dados original. Essa variedade é alcançada por meio da duplicação de certas amostras e da exclusão de outras em cada subconjunto, garantindo uma composição única para cada um. Posteriormente, cada árvore é treinada utilizando um subconjunto específico.

Para ampliar a diversidade entre as árvores de decisão, cada divisão dentro de uma árvore considera apenas um subconjunto limitado de características. Esta estratégia é adotada para evitar que características dominantes no processo de classificação monopolizem a maioria das divisões. Caso contrário, as árvores podem se tornar excessivamente similares, frustrando o objetivo de reduzir a variância. Em cada ponto de divisão, um conjunto aleatório de \sqrt{p} características é selecionado para análise, onde p representa o número total de características disponíveis.

Após serem geradas as diferentes árvores, a classificação é feita levando em consideração o resultado de todas. Em um problema de classificação, como é o caso deste trabalho, é feita uma votação majoritária entre elas para definir a classe escolhida. A Figura 4 mostra um exemplo de floresta aleatória, com três árvores de decisão.

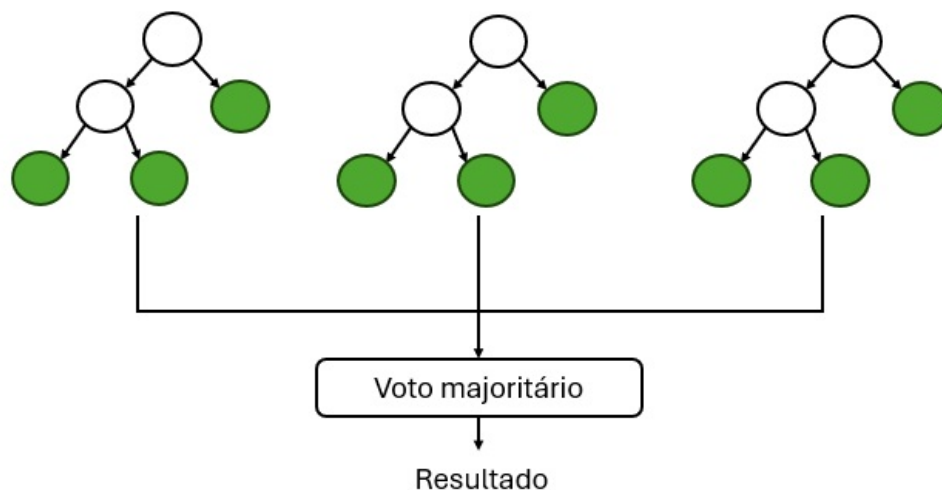


Figura 4 – Exemplo floresta aleatória.

Fonte: O autor.

2.2.3 Redes Neurais

Uma Rede Neural Artificial é um modelo de aprendizagem de máquina inspirado no cérebro humano. Ela foi desenvolvida a partir de estudos sobre o comportamento de neurônios, suas conexões e seu processo de aprendizagem.

O elemento fundamental de uma rede neural é o neurônio. Assim como seu paralelo biológico, um neurônio recebe, processa e transmite uma informação para outros neurônios. Um neurônio pode ter diversas entradas, sendo elas vindo de outros neurônios ou algum dado externo. Cada entrada é associada a um peso, um valor multiplicativo que sinaliza a importância desta entrada. Quanto maior o peso maior será a importância da entrada.

Estas entradas são agregadas no neurônio, geralmente as somando. O processamento deste é feito pela função de ativação. O resultado encontrado desta operação é o que será usado como saída do neurônio. Funções de ativação típicas são: sigmoid, tangente hiperbólica, softmax e ReLU (SHARMA; SHARMA; ATHAIYA, 2017). A Figura 5 é uma representação típica de um neurônio, mostrando todas as etapas discutidas anteriormente.

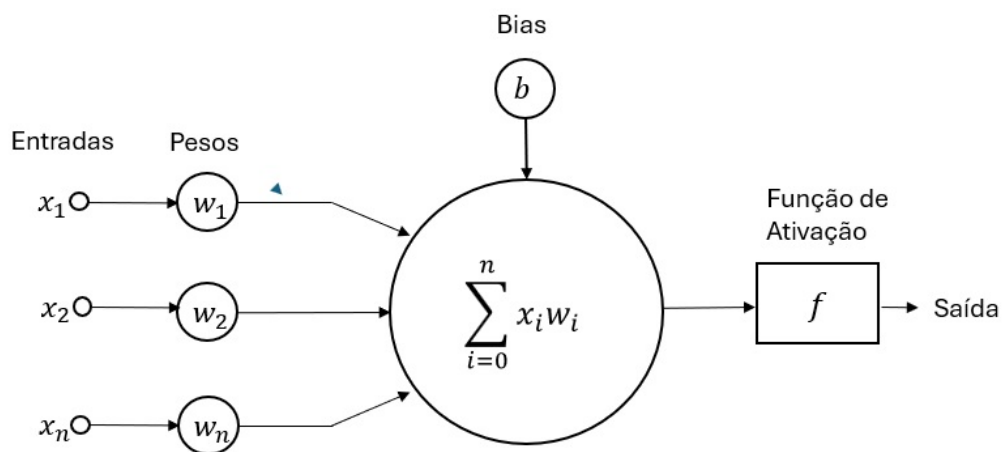


Figura 5 – Neurônio.

Fonte: O autor.

Uma rede neural é, portanto, um agregamento destes neurônios. Uma rede neural típica é dividida em camadas, onde um neurônio de uma camada alimenta os neurônios de uma camada seguinte. Numa rede neural existem a camada de entrada, a camada de saída e as camadas ocultas. Elas funcionam da seguinte maneira:

- Camada de entrada: Recebe dados externos como entrada. Cada neurônio nesta camada representa uma característica de entrada.
- Camadas ocultas: Camadas intermediárias que fazem o processamento dos dados. Quanto mais camadas e neurônios, mais complexa se torna a rede, aumentando o tempo de treinamento e de processamento dos dados.
- Camada de saída: Gera o resultado final da rede.

A imagem 6 ilustra uma rede neural com duas entradas, duas camadas ocultas com 5 neurônios cada e duas saídas.

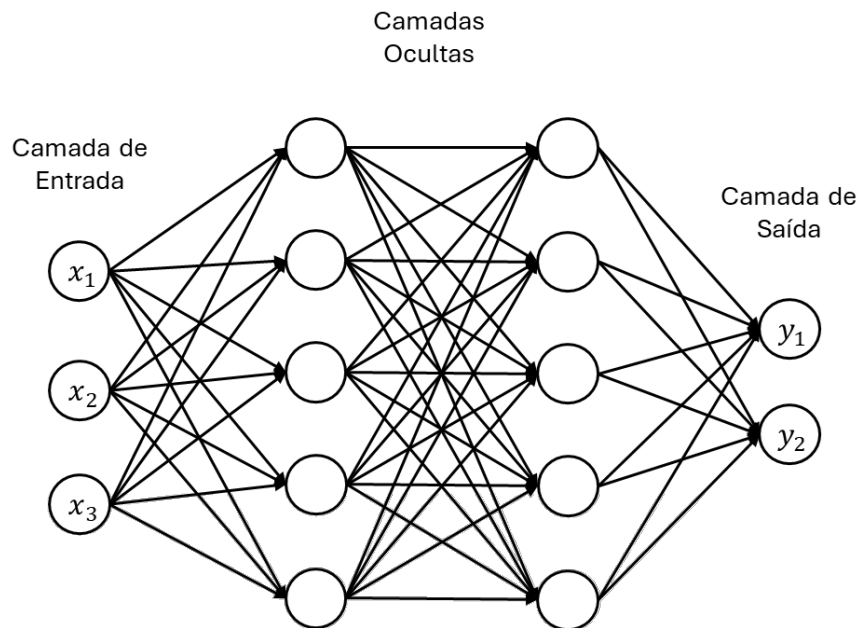


Figura 6 – Rede Neural.

Fonte: O autor.

2.3 Aprendizado de Máquina em Sistemas Embarcados

Um aspecto importante dos modelos de aprendizado de máquina embarcados é otimizá-los para hardware limitado. Uma forma de fazer é utilizando conceitos como quantização.

Quantização, no contexto de aprendizado de máquina, se refere a redução da resolução dos pesos de um modelo. O objetivo é representar estes pesos com uma menor quantidade de bits, sem comprometer, ou comprometendo o menos possível, o desempenho. Dentre os benefícios deste processo está não somente a redução da memória necessária para armazenar os pesos do modelo, mas também um processamento mais rápido a partir de operações menos custosas (ROTH *et al.*, 2020).

Pesos em uma rede neural, por exemplo, são comumente representados em ponto flutuante com 64 ou 32 bits de precisão (BETHGE *et al.*, 2021). Após o processo de quantização, estes mesmos pesos podem ser representados em uma definição menor em ponto flutuante ou até mesmo por inteiros de 16 bits ou 8 bits.

Existem dois tipos principais de quantização, treinamento consciente de quantização e quantização pós treinamento. No primeiro método a quantização é realizada já durante o treinamento, já o segundo o modelo é treinado normalmente e a quantização funciona como um pós processamento.

3 Metodologia

Este capítulo descreve as etapas para a construção do classificador de movimentos. A Figura 7 mostra as etapas da metodologia, mostrando suas principais etapas.

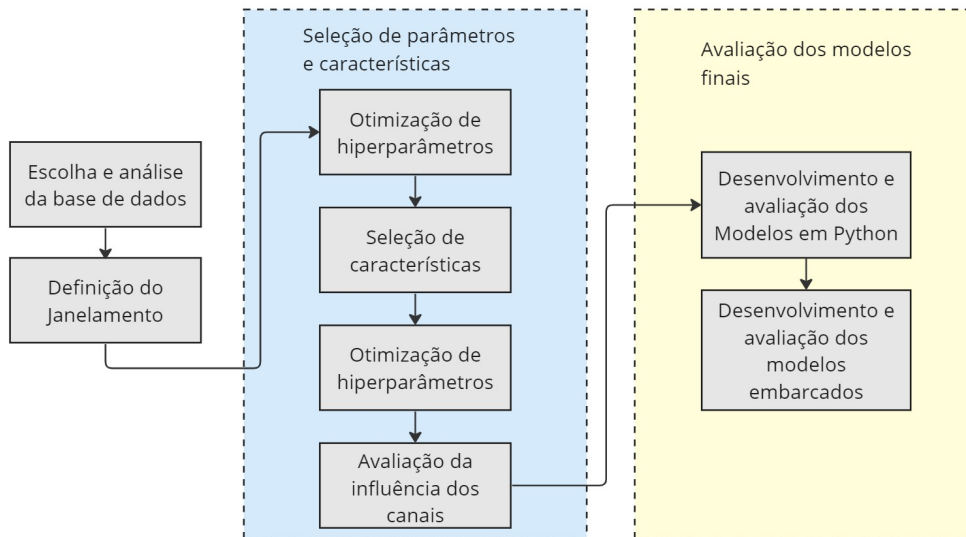


Figura 7 – Esquema da metodologia.

Fonte: O autor.

Primeiramente no trabalho foi feita a escolha da base de dados com a qual os modelos serão treinados. Foi definido, também, como será feito o janelamento dos dados, de onde serão extraídas as características.

Na seção "Seleção de parâmetros e características", em azul na Figura 7, mostra como os modelos foram otimizados, levando em conta hiperparâmetros e características. Nesta parte também foram ordenados, em função da influência da taxa de acerto dos modelos, os canais de aquisição de sEMG.

A seção "Avaliação dos modelos finais" apresenta o treinamento e a obtenção das métricas dos modelos otimizados na etapa anterior. Esta seção foi dividida em duas partes: a primeira avaliando os modelos em Python e a segunda avaliando os modelos no microcontrolador.

3.1 Materiais utilizados

O treinamento dos modelos foi feito utilizando python em conjunto com bibliotecas. A Tabela 1 mostra as bibliotecas versões utilizadas no trabalho.

Ferramenta	Versão
Python	3.10.12
Scikit-learn	1.3.2
Tensorflow	2.10
Arduino IDE	2.2.1
MicroML Generator	1.1
EverywhereML	0.2.37

Tabela 1 – Versões de softwares utilizados

Para embarcar o modelo foi utilizado a placa de desenvolvimento Esp32 DevKit v1. Esta placa conta com o microprocessador Xtensa Dual-Core 32-bit LX6, memória flash integrada de 4MiB e um clock de 240MHz. A Figura 8 mostra o microcontrolador utilizado conectado no computador.



Figura 8 – Microcontrolador utilizado.

Fonte: O autor.

3.2 Base de dados

A base de dados NinaPro é empregada como referência em estudos que exploram a correlação entre o movimento de braço e mão e os sinais de sEMG, visando o desenvolvimento de próteses para amputados (ATZORI *et al.*, 2014). Sua escolha se fundamenta por ser muito utilizada em trabalhos de classificação de movimentos, simplificando a comparação entre diferentes abordagens e resultados.

A aquisição dos dados realizada por (ATZORI *et al.*, 2014) utilizou 12 eletrodos *Delsys Trigno Wireless*, com uma taxa de aquisição de 2kHz, posicionados 2 no braço

e 10 antebraço de cada voluntário. Cada voluntário executou uma série de movimentos, intercalados uma posição de descanso. Cada movimento foi repetido sequencialmente 6 vezes, sem aleatorização, conforme apresentado em uma tela. A Figura 9 ilustra o protocolo de aquisição.

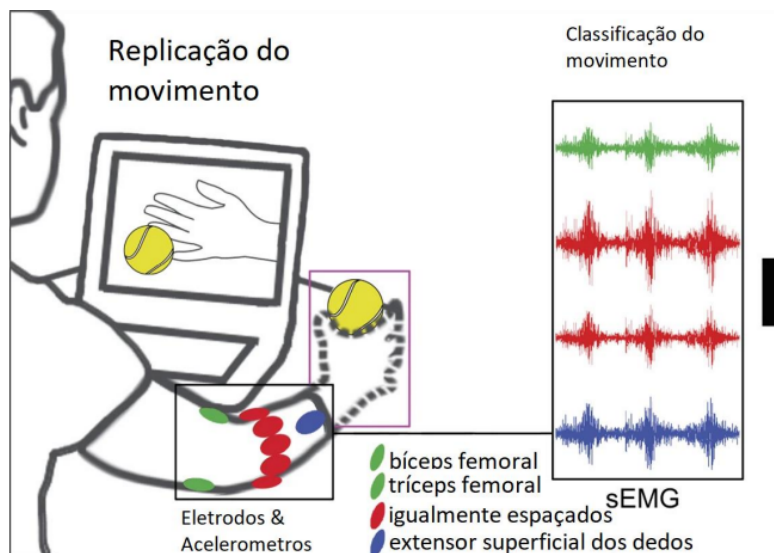


Figura 9 – Protocolo de Aquisição
Fonte: (ATZORI *et al.*, 2014), adaptado

Em seguida, os dados passaram por uma etapa de pós-processamento, que incluiu a correção dos rótulos de cada movimento, a fim de corrigir erros experimentais e de tempo de reação humano. Os rótulos corrigidos foram utilizados para o trabalho.

Foi utilizada a base de dados 2 do NinaPro para a realização deste trabalho. Essa base de dados é composta por dados de 40 voluntários, todos não amputados. Os voluntários executaram três exercícios com sequências de movimentos diferentes. Para este estudo, apenas os dados do primeiro exercício foram considerados, que inclui 17 movimentos distintos, além do descanso. A Figura 10 apresenta os movimentos realizados durante esse exercício.

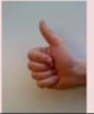


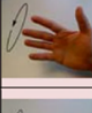





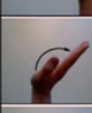





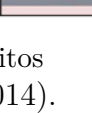

Exercise B					
1	Thumb up		9	Wrist supination (axis: middle finger)	
2	Extension of index and middle, flexion of the others		10	Wrist pronation (axis: middle finger)	
3	Flexion of ring and little finger, extension of the others		11	Wrist supination (axis: little finger)	
4	Thumb opposing base of little finger		12	Wrist pronation (axis: little finger)	
5	Abduction of all fingers		13	Wrist flexion	
6	Fingers flexed together in fist		14	Wrist extension	
7	Pointing index		15	Wrist radial deviation	
8	Adduction of extended fingers		16	Wrist ulnar deviation	
			17	Wrist extension with closed hand	

Figura 10 – Movimentos feitos
Fonte: (ATZORI *et al.*, 2014).

3.3 Janelamento

Uma janela de dados é um intervalo em que as características do sinal foram calculadas. Se o tamanho da janela for pequeno, o intervalo produzirá um tempo de resposta mais curto para o usuário, mas também aumentará a variância dos dados. Por outro lado, se a janela for muito grande, ela pode ter o efeito contrário, melhorando a taxa de acerto do classificador e aumentando o tempo de resposta.

Para garantir um movimento fluido, o tempo entre a ação e a resposta deve ser inferior a 300ms (OSKOEI; HU, 2008). Assim, foi definida uma janela de 300ms com 75ms de sobreposição. Esses parâmetros permitem que a aquisição e o processamento estejam dentro do critério mínimo para a fluidez do movimento. A Figura 11 ilustra o janelamento para um dos canais.

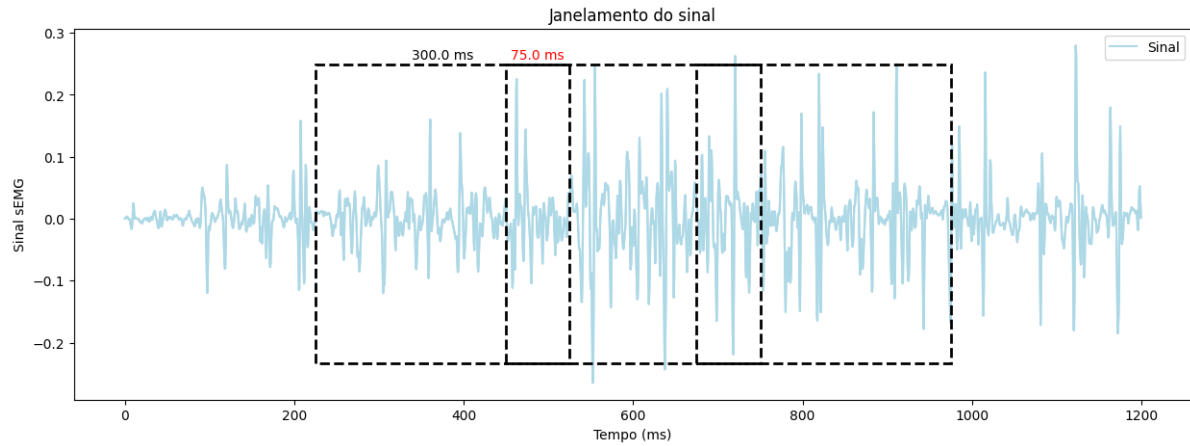


Figura 11 – Janelamento do sinal de sEMG.

Fonte: O autor.

Cada dado adquirido contém dois rótulos de movimento: um que corresponde ao movimento exibido na tela quando o movimento foi feito e outro corrigido posteriormente, levando em conta fatores como o tempo de reação de cada voluntário. Foi considerado o segundo rótulo para a classificação. O rótulo atribuído a cada janela é aquele que ocorre com mais frequência nos dados contidos nela.

3.4 Características

Foram pré-escolhidas 6 características a serem extraídas dos canais. São elas:

- RMS

O valor RMS (Root Mean Square) é uma medida matemática que representa a magnitude efetiva de uma quantidade variável. Ele é calculado ao elevar ao quadrado os valores do sinal, tirar a média desses quadrados e, por fim, extrair a raiz quadrada dessa média.

$$RMS = \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2} \quad (4)$$

- Média Aritmética

A média aritmética calcula o valor central de um conjunto de dados, somando todos os valores dividindo pela quantidade de números.

$$Média = \frac{1}{N} \sum_{i=1}^N x_i \quad (5)$$

- Desvio Padrão

O desvio padrão é a medida estatística que indica a quantidade de variação esperada dos valores sobre sua média. Um desvio padrão baixo indica que os valores tendem a estarem mais próximos da média, já um desvio padrão alto indica que estão mais afastados.

$$DesvioPadrão = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2} \quad (6)$$

- Variância

O quadrado do desvio padrão

$$\text{Variância} = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2 \quad (7)$$

- Cruzamento por zero

Indica quantas vezes os valores passaram por zero, ou seja, quantas vezes trocaram de sinal.

- Curtose

A curtose é uma medida estatística que descreve o grau de achatamento ou alongamento da distribuição de um conjunto de dados, comparada com a distribuição normal.

$$\text{Curtose} = \frac{1}{N} \sum_{i=1}^N \left(\frac{x_i - \mu}{\sigma} \right)^4 - 3 \quad (8)$$

Essas características passarão por uma etapa de seleção individual para cada tipo de modelo, como será descrito a seção seguinte.

3.5 Seleção de parâmetros e características

Foram criados três conjuntos de modelos: baseado em redes neurais, em regressão logística e em florestas aleatórias. Foram analisadas diferentes configurações para os métodos, a fim de investigar sua influência nas métricas avaliadas. Os modelos de Floresta Aleatória e Regressão Logística foram treinados utilizando *scikit-learn* e as redes neurais utilizando *Tensorflow* e *Keras*.

Os modelos foram treinados individualmente para os dados de cada voluntário. A taxa de acerto de uma configuração foi considerada, portanto, como a média da taxa de acerto de todos os voluntários. Nesta etapa de seleção de parâmetros e características foram utilizados os dados de 12 dos 40 voluntários, a fim de diminuir o tempo de processamento. Posteriormente, foram avaliados os dados com todos os voluntários. Em todas as etapas de treinamento foi utilizada bases de treinamento e teste, divididas aleatoriamente, na proporção de 80% e 20% respectivamente.

A Figura 12 ilustra os passos realizadas nesta etapa. As etapas de otimização de hiperparâmetros foram desconsideradas para Regressão Logística.

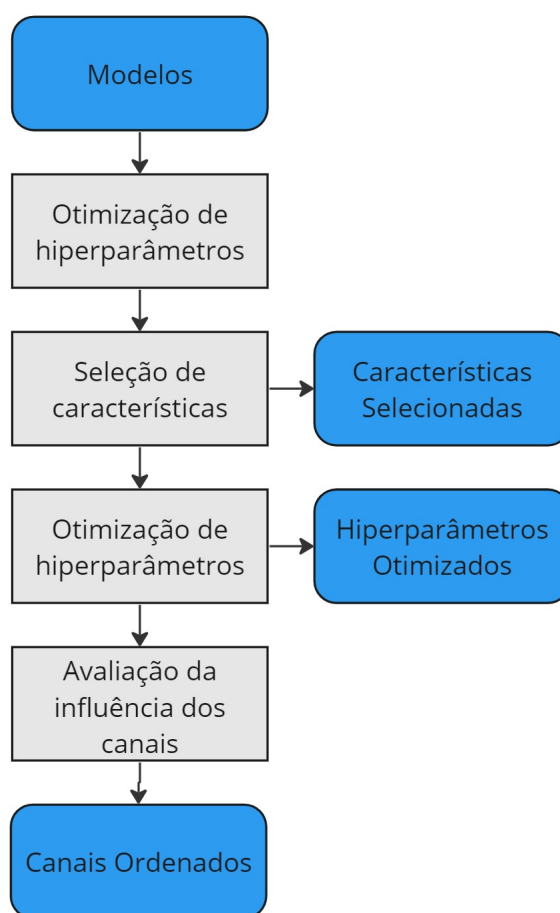


Figura 12 – Etapas da seleção de parâmetros e características.

Fonte: O autor.

3.5.1 Otimização dos hiperparâmetros

Foram feitas pesquisas em grade para otimizar os hiperparâmetros dos modelos. A pesquisa foi feita inicialmente antes da seleção de características, utilizando como entrada a totalidade das características que foram escolhidas da literatura. Esta etapa será repetida, mas desta vez apenas com as características selecionadas, utilizando os mesmos parâmetros.

A Tabela 2 mostra os hiperparâmetros variados durante os testes. A faixa de variação dos parâmetros foram definidas a partir de testes preliminares. Como Regressão Logística não apresenta muitas possibilidades de variação, não foi feita pesquisa em grade com ela.

Modelo	Hiperparâmetro	Valores
Rede Neural	Camadas Ocultas	2 a 4, passo 1
	Neurônios por camada	25 a 100, passo 25
	Função de ativação	Seigmoid e Relu
Floresta Aleatória	Profundidade Máxima	2 a 10, passo 2
	Número de Árvores	2 a 20, passo 2

Tabela 2 – Hiperparâmetros Testados

A métrica escolhida para selecionar o melhor modelo foi a taxa de acerto média entre os voluntários para cada modelo.

3.5.2 Seleção de Características

Com os modelos otimizados, foram escolhidas as características a partir de técnica de seleção sequencial. Os modelos foram treinados primeiramente com apenas uma característica, avaliando o taxa de acerto com todas as possíveis escolhas. Em seguida, mantendo fixa a melhor característica individual, foi avaliada as melhores duplas de características adicionando as outras restantes. O processo segue da mesma forma até que seja testado o modelo com todas as características. É escolhido então, dentre os conjuntos testados, o com maior taxa de acerto média.

A Figura 13 ilustra o processo de seleção de características. Nela é mostrado as duas primeiras etapas, onde a primeira característica é selecionada e é buscada a segunda.

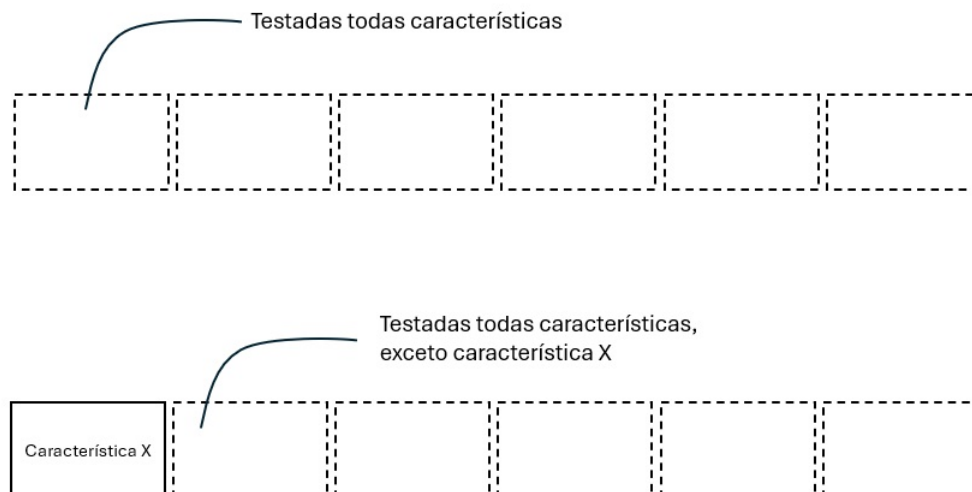


Figura 13 – Representação da seleção sequencial de características.

Fonte: O autor.

Esta etapa foi feita de forma individual para as Florestas Aleatórias, Rede Neural e Regressão Logística. As características escolhidas, portanto, podem ser diferentes para os três modelos.

3.5.3 Avaliação da influência dos canais

Para cada tipo de classificador foram criados modelos aumentando de forma sequencial o número de canais de entrada. Cada canal representa os dados vindos de um eletrodo, com todas suas características, definidas na etapa anterior e iguais para todos os canais. Foi feita uma nova otimização de hiperparâmetros, da mesma forma que feita anteriormente, mas utilizando somente as características selecionadas para cada modelo.

Para avaliar qual o melhor canal para ser adicionado em cada etapa, são avaliadas todas as possibilidades de incremento e escolhida aquela que representa o maior aumento na

taxa de acerto. Antes desta etapa, foi realizada uma nova otimização de hiperparâmetros, utilizando os modelos com as características selecionadas.

Para a escolha do primeiro canal, por exemplo, foram treinados modelos de um canal para todas as 12 alternativas e escolhido a melhor. Com o primeiro já escolhido e avaliado é feito a mesma coisa, mas para o segundo canal, avaliando todas as outras 11 possibilidades. A escolha das entradas dos próximos classificadores é feita da mesma forma, até chegar no classificador com 12 canais de entrada. O processo é similar ao de seleção de características. Os hiperparâmetros são mantidos os mesmos, independentemente do numero de canais.

A Figura 14 ilustra o processo. Os primeiros canais (a esquerda na figura), são os considerados mais impactantes para a taxa de acerto.

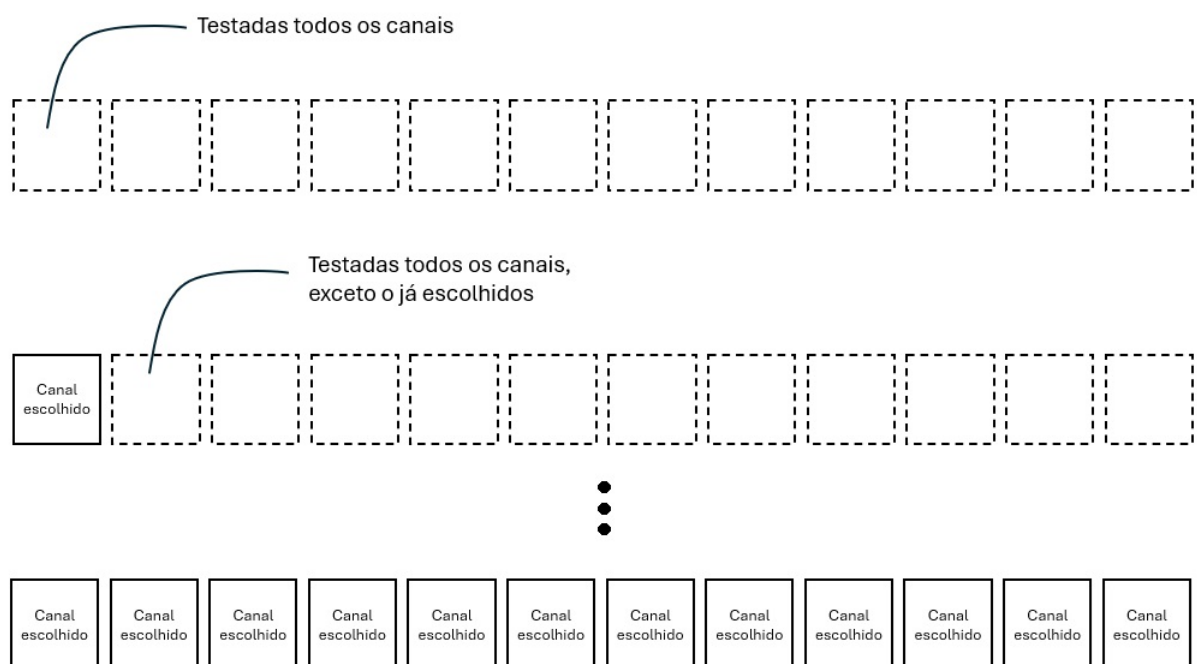


Figura 14 – Representação do ordenamento dos canais.

Fonte: O autor.

Avaliar estas diferentes possibilidades teve como objetivo mostrar o impacto do número de entradas nas métricas de desempenho. Ao aumentar o número de entradas é esperado que a taxa de acerto melhore, porém a latência também pode variar. Além disso pode-se avaliar quais canais de entrada são de menor e maior impacto. Através destas medidas, torna-se viável efetuar uma tomada de decisão informada sobre o projeto, ponderando o equilíbrio entre essas duas métricas.

3.6 Implementações dos Modelos em Sistema Embarcado

Para embarcar o modelo no microprocessador foi preciso converter o código dos modelos para a linguagem C. No caso dos modelos feitos com o *scikit-learn* foi utilizada a biblioteca *MicroML Generator* para converter os modelos diretamente para C. O conversor transforma dados em um código estruturado como uma biblioteca, que pode ser

posteriormente chamado para realizar as predições. Os parâmetros dos modelos já foram treinados com floats de 32 bits.

As redes neurais feitas com *Tensorflow* foram convertidas para sua versão em *Tensorflow Lite*, que é compatível com microcontroladores. Para gerar o código em C que é usado diretamente no Esp32 foi utilizado a biblioteca *EverywhereML*. Os parâmetros das redes neurais são de floats de 64 bits.

3.7 Avaliação dos modelos finais

Os modelos finais foram implementados tanto no microcontrolador quanto no computador. Foram treinados os modelos dos 40 voluntários para cada configuração avaliado no computador e 5 voluntários no microcontrolador. As métricas avaliadas no microcontrolador foram: taxa de acerto, latência e ocupação de memória; já no computador foi avaliada somente a taxa de acerto.

3.7.1 Taxa de acerto

A taxa de acerto pode ser obtida antes do modelo ser embarcado. Foram treinados para todos os 40 voluntários os modelos de 1-12 canais com as configurações definidas anteriormente. Os modelos de Florestas Aleatórias e Regressão Logística foram treinados para parâmetros de 64 bits e Redes Neurais de 64 bits. Os modelos desenvolvidos tiveram a taxa de acerto aferida a partir da base de testes.

A melhor configuração em modelos de canais também passou por uma análise mais aprofundado, sendo calculadas a precisão, sensibilidade e f1-score por classe. A definição dessas métricas são apresentadas nas Equações 9, 10, 11.

$$\text{Precisão} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Positivos}} \quad (9)$$

$$\text{Sensibilidade} = \frac{\text{Verdadeiros Positivos}}{\text{Verdadeiros Positivos} + \text{Falsos Negativos}} \quad (10)$$

$$\text{F1-Score} = 2 \times \frac{\text{Precisão} \times \text{Sensibilidade}}{\text{Precisão} + \text{Sensibilidade}} \quad (11)$$

Todos os modelos foram então reavaliados, desta vez no ESP32, utilizando um grupo de 5 voluntários nos quais os modelos foram embarcados. Para a avaliação, foi utilizada a comunicação serial no microprocessador para enviar cada vetor da base de testes e receber o resultado da classificação. Estes resultados foram então comparados com aqueles obtidos no software, com o objetivo de verificar se o modelo embarcado está produzindo resultados equivalentes aos obtidos no computador.

3.7.2 Latência

Para aferir a latência de cada modelo foram separadas as duas etapas do processamento, a extração de características e o processamento do modelo em si.

Foi medido o tempo de processamento de cada um dos modelos descritos na seção anterior, a partir das entradas já pré-processadas. Para fazer esta medida no microcontrolador, foi novamente utilizado os dados de 5 voluntários escolhidos aleatoriamente. Esses dados foram submetidos a um pré-processamento em software antes de ser testado no microcontrolador.

Posteriormente foi medido tempo de processamento de cada característica, junto com a etapa de normalização. Foram escolhidos aleatoriamente 10 amostras da base de dados para efetuar o teste. Cada amostra é uma janela com dados de um certo período, conforme descrito na seção sobre janelamento. Foi medido o tempo necessário para a normalização destes dados e a extração de cada característica. Foi, então, feita média destas 10 medidas.

Como o processamento no microcontrolador foi feito de forma inteiramente sequencial, é possível determinar o tempo total de processamento para cada uma das configurações somado os resultados obtidos em cada etapa.

3.7.3 Ocupação de memória

A ocupação de memória foi medida considerando o espaço que o programa compilado ocupa no microcontrolador. A ocupação considerada foi apenas a do modelo, sem o código para extração de características. Foi considerada a média dos resultados obtidos dos 5 voluntários com seus modelos embarcados para cada configuração.

4 Resultados

Este capítulo apresentará os resultados obtidos a partir dos testes realizados, bem como a análise destes.

4.1 Seleção de parâmetros e características

Esta seção trata da obtenção de parâmetros e características que serão usadas para o treinamento dos modelos finais. Para todas as etapas desta seção foram usados os mesmos 12 voluntários para a obtenção das métricas apresentadas. Os voluntários escolhidos foram os de número 1 a 12 da base de dados.

4.1.1 Otimização de hiperparâmetros

Foi realizado, como descrito na seção 3.5.1, uma pesquisa em grade para obter hiperparâmetros otimizados para o caso com todas as características e 12 canais de entrada para os modelos.

Modelo	Hiperparâmetro	Valores Possíveis	Valor Escolhido
Rede Neural	Camadas Ocultas	2 a 4, passo 1	2
	Neurônios por camada	25 a 100, passo 25	100
	Função de ativação	Seigmoid e Relu	Relu
Floresta Aleatória	Profundidade Máxima	2 a 10, passo 2	10
	Número de Árvores	2 a 20, passo 2	18

Tabela 3 – Resultados da primeira otimização de hiperparâmetros

Foi considerado como critério da pesquisa a taxa de acerto médio dos 12 voluntários treinados para cada configuração. A Tabela 3 mostra os resultados obtidos.

4.1.2 Seleção de características

Para definir o melhor conjunto de características foi realizada uma seleção sequencial, sendo a taxa de acerto média o critério para a seleção. Os hiperparâmetros utilizados para este teste foram os obtidos na seção anterior, que constam na Tabela 3.

A Tabela 4 mostra as características que participaram da seleção, sendo elas iguais para todos os modelos. A Tabela 5 mostra quais foram as escolhidas em cada caso.

Características testadas

RMS
 Média Aritimética
 Desvio Padrão
 Variância
 Cruzamento por zero
 Curtose

Tabela 4 – Características testadas

Modelo	Características selecionadas
Rede Neural	Desvio Padrão
Floresta Aleatória	RMS
Regressão Logística	RMS e Variância

Tabela 5 – Características selecionadas

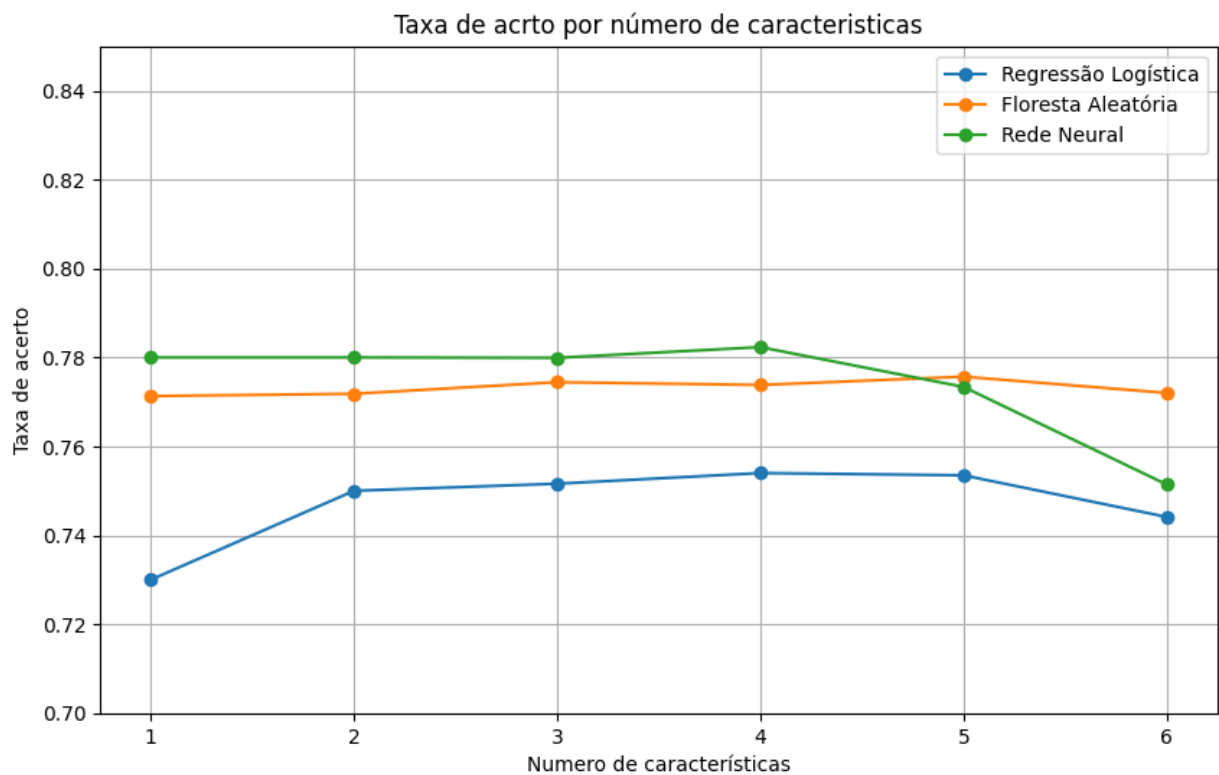


Figura 15 – Taxa de acerto por número de características

Fonte: O autor.

A Figura 15 mostra as taxas de acerto conforme a adição sequencial de características, sendo sempre adicionada a característica que mais melhore a taxa de acerto.

Não houve aumento da taxa de acerto com o acréscimo de muitas características para o sistema. Algumas características tiveram baixa taxa de acerto quando estiveram sozinhas. Curtose e cruzamento por zero, por exemplo, tiveram uma taxa de acerto média entre os 3 modelos, quando sozinhas, de 52% e 47% respectivamente

4.1.3 Otimização de hiperparâmetros

Utilizando as características selecionadas os hiperparâmetros foram novamente otimizados. Foi realizada a mesma pesquisa em grade feita na seção 4.1.1, com a diferença que os modelos usam como entrada somente as características selecionadas. A Tabela 6 mostra os resultados obtidos.

Modelo	Hiperparâmetro	Valores Possíveis	Valor Escolhido
Rede Neural	Camadas Ocultas	2 a 4, passo 1	2
	Neurônios por camada	25 a 100, passo 25	100
	Função de ativação	Seigmoid e Relu	Relu
Floresta Aleatória	Profundidade Máxima	2 a 10, passo 2	10
	Número de Árvores	2 a 20, passo 2	18

Tabela 6 – Resultados da segunda otimização de hiperparâmetros

Apesar da mudança das características, os hiperparâmetros obtidos na pesquisa em grade foram os mesmo que os obtidos anteriormente.

4.1.4 Avaliação da influência dos canais

Os canais foram ordenados de forma a descobrir a melhor combinação para cada número de canais utilizados, conforme descrito na seção 3.5.3. Com estes resultados será possível definir quais os canais mais importantes para a classificação dos movimentos. A Tabela 7 mostra os ordenamentos obtidos para cada modelo.

Modelo	Ordenamento dos canais (do mais ao menos importante)
Rede Neural	9, 10, 12, 2, 11, 8, 1, 3, 7, 4, 5, 6
Floresta Aleatória	3, 10, 12, 9, 2, 11, 6, 1, 7, 8, 5, 4
Regressão Logística	3, 10, 12, 9, 2, 11, 6, 1, 7, 8, 5, 4

Tabela 7 – Resultados do ordenamento de importância dos canais

A Figura 16 mostra qual a posição de cada eletrodo referente a cada canal. Os canais 1-8 estão posicionados igualmente espaçados no antebraço. Por sua vez, os canais 9 e 10 situam-se nos músculos flexores superficial dos dedos e extensor superficial dos dedos, respectivamente. Já os canais 11 e 12 estão localizados no bíceps e no tríceps, respectivamente.

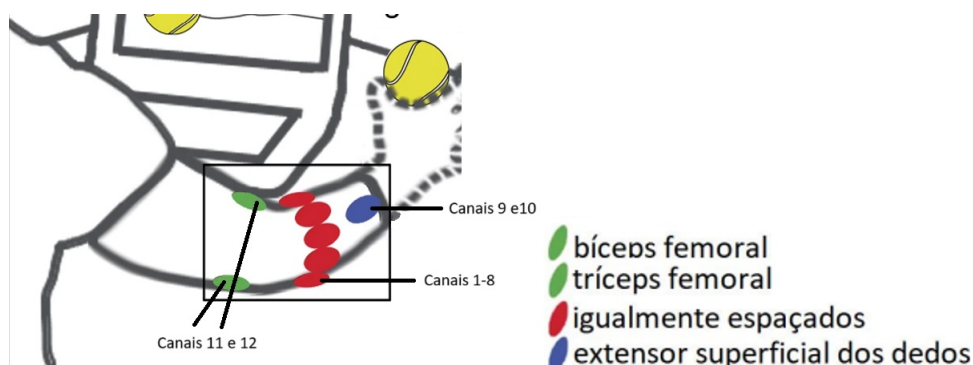


Figura 16 – Posição dos eletrodos

Fonte: (ATZORI *et al.*, 2014), adaptado

Alguns canais apareceram mais recorrentemente entre os mais importantes. Os canais de 9-10, posicionados mais próximos a mão, estão mais vezes entre os mais importantes, assim como os canais 11 e 12, posicionados no bíceps e tríceps.

4.2 Avaliação dos modelos finais

4.2.1 Taxa de acerto

Foram treinados os modelos de acrescentando canais de forma sequencial, acrescentando do mais importante ao menos. Nesta etapa foram treinados os modelos para todos os 40 voluntários, um modelo para cada, em todas as configurações. Os hiperparâmetros utilizados foram os da segunda otimização, apresentados na Tabela 6.

As figuras 17, 18 e 19 mostram as taxas de acerto encontradas para os modelos baseados em redes neurais, florestas aleatórias e regressão logística, respectivamente. As taxas de acertos apresentadas foram encontradas rodados os modelos em Python, no computador.

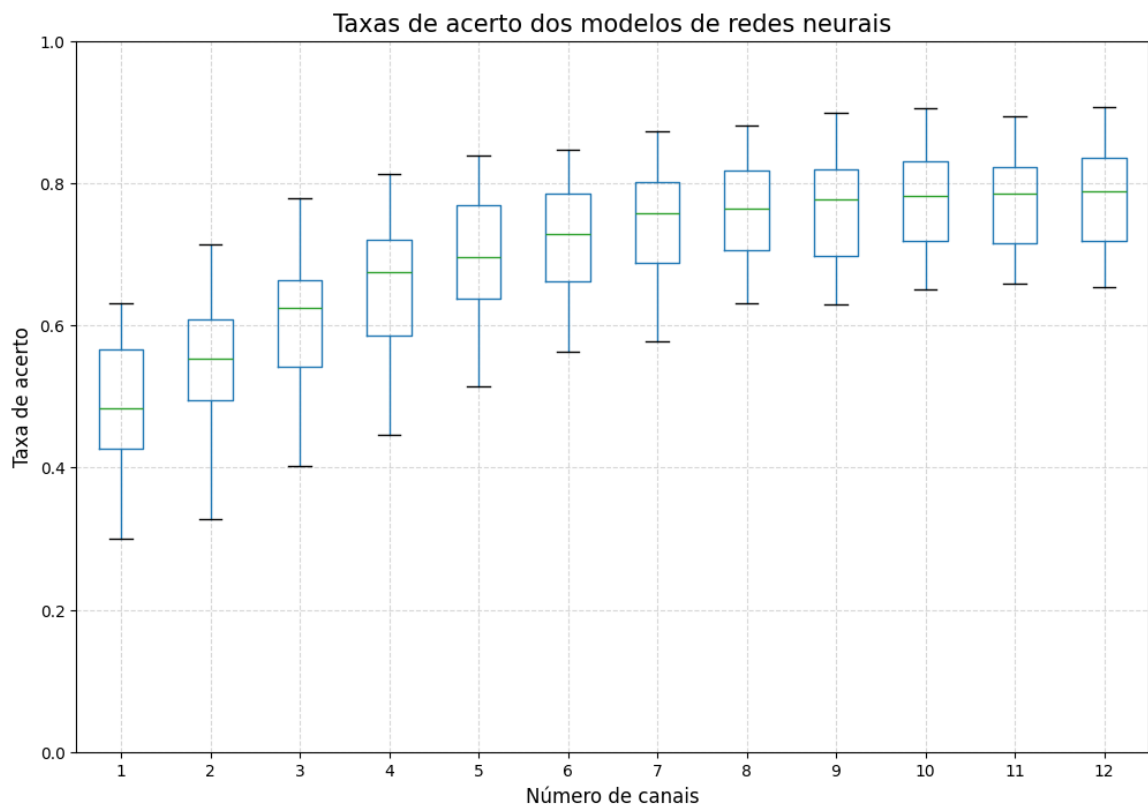


Figura 17 – Taxa de acerto redes neurais por número de canais, avaliada para 40 indivíduos.

Fonte: O autor.

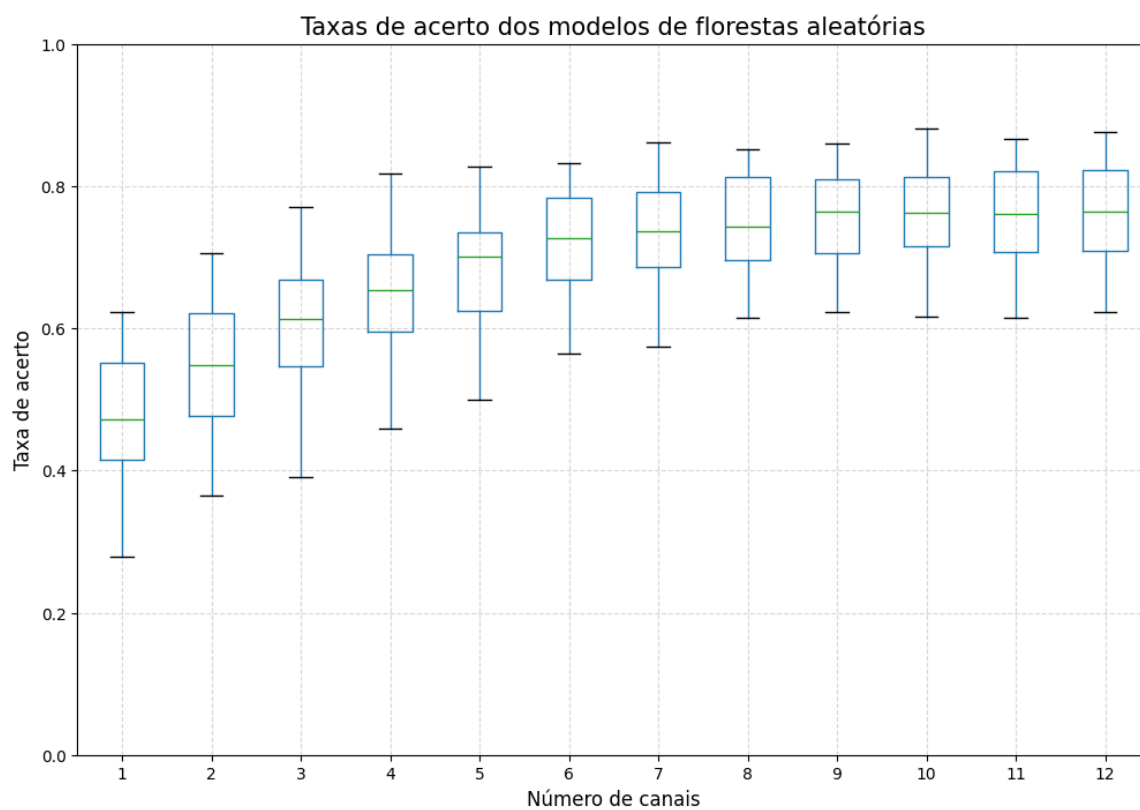


Figura 18 – Taxa de acerto florestas aleatórias por número de canais, avaliada para 40 indivíduos.

Fonte: O autor.

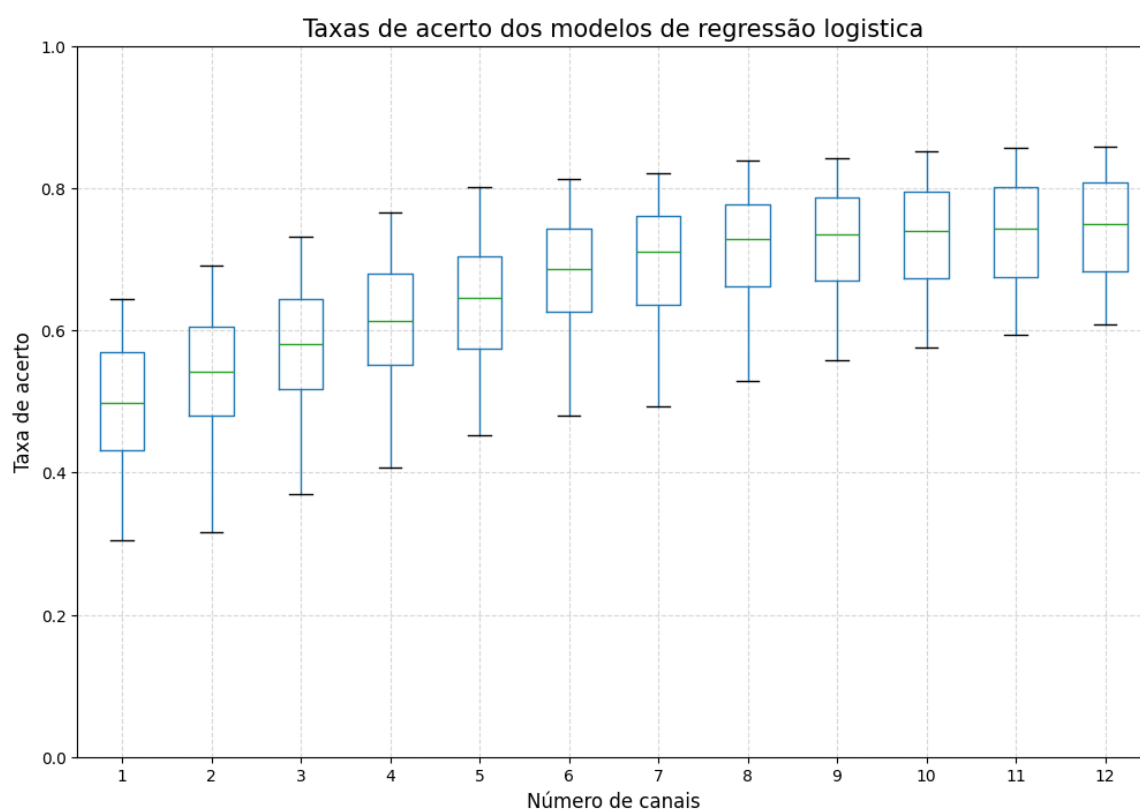


Figura 19 – Taxa de acerto regressão logística por número de canais, avaliada para 40 indivíduos.

Fonte: O autor.

Como esperado os modelos aumentam a taxa de acerto com o aumento do número de canais. O aumento, entretanto, diminui à medida em que aumentam os canais. Pode ser considerada uma escolha de projeto, portanto, utilizar menos que doze canais, podendo reduzir a latência e o número de eletrodos necessários sem prejudicar muito a taxa de acerto.

Foram analisados mais profundamente os modelos com maior taxa de acerto, que para os três tipos de classificadores foram os com 12 canais. As Tabelas 8, 9 e 10 exibem a precisão, sensibilidade e f1-score para cada classe desses modelos. Além disso, são apresentadas a média das métricas entre todas as classes, sem considerar o número de amostras, e a média ponderada, que utiliza o número de amostras como peso. As classes representam os movimentos apresentados na Figura 10 e 0 a posição de repouso.

Observa-se que, devido ao repouso ser a classe mais frequente - cada movimento é seguido por um período de repouso - sua sensibilidade é significativamente maior. Esse fenômeno contribui para que a média ponderada da sensibilidade seja superior à média calculada entre todas as classes. Da mesma forma, o f1-score associado ao repouso é maior em comparação com as demais classes, refletindo essa frequência elevada de ocorrência. A precisão desta classe, porém, se mantém na ordem que das outras.

Além disso foi plotada as matrizes de confusão para os mesmos modelos nas Figuras 20, 21 e 22 . Os valores de cada linha foram normalizados. É possível perceber que a maior parte dos erros das classes de movimento (1-17) é classificando erroneamente como repouso (0), por conta do desbalanceamento da base.

Classe	Precisão	Sensibilidade	F1-Score	Amostras
0	0,79	0,93	0,85	14717
1	0,84	0,73	0,78	1719
2	0,78	0,64	0,70	1352
3	0,77	0,73	0,75	734
4	0,79	0,68	0,73	1045
5	0,77	0,60	0,68	1047
6	0,80	0,66	0,72	771
7	0,80	0,62	0,70	1306
8	0,75	0,60	0,67	929
9	0,75	0,65	0,70	1031
10	0,76	0,66	0,71	976
11	0,70	0,57	0,63	1195
12	0,69	0,57	0,62	864
13	0,76	0,70	0,73	656
14	0,79	0,72	0,76	806
15	0,81	0,61	0,70	941
16	0,70	0,67	0,69	911
17	0,79	0,69	0,73	835
Média	0,77	0,67	0,71	31835
Média Ponderada	0,78	0,78	0,77	31835

Tabela 8 – Dados por classe para o modelo com 12 canais de Rede Neural

Classe	Precisão	Sensibilidade	F1-Score	Amostras
0	0,77	0,96	0,85	14717
1	0,85	0,71	0,77	1719
2	0,77	0,59	0,67	1352
3	0,75	0,67	0,71	734
4	0,77	0,59	0,67	1045
5	0,73	0,53	0,61	1047
6	0,84	0,60	0,70	771
7	0,81	0,59	0,68	1306
8	0,66	0,54	0,59	929
9	0,75	0,63	0,69	1031
10	0,74	0,62	0,67	976
11	0,72	0,48	0,57	1195
12	0,61	0,54	0,58	864
13	0,79	0,64	0,70	656
14	0,80	0,60	0,69	806
15	0,85	0,57	0,68	941
16	0,68	0,59	0,63	911
17	0,80	0,63	0,71	835
Média	0,76	0,61	0,68	31835
Média Ponderada	0,76	0,76	0,75	31835

Tabela 9 – Dados por classe para o modelo com 12 canais de Floresta Aleatória

Classe	Precisão	Sensibilidade	F1-Score	Amostras
0	0,75	0,95	0,84	14717
1	0,83	0,65	0,73	1719
2	0,74	0,53	0,62	1352
3	0,77	0,66	0,71	734
4	0,74	0,59	0,66	1045
5	0,74	0,48	0,58	1047
6	0,83	0,57	0,68	771
7	0,79	0,52	0,63	1306
8	0,66	0,51	0,58	929
9	0,71	0,59	0,65	1031
10	0,71	0,61	0,66	976
11	0,68	0,43	0,53	1195
12	0,61	0,51	0,55	864
13	0,74	0,62	0,67	656
14	0,77	0,60	0,67	806
15	0,76	0,53	0,63	941
16	0,68	0,58	0,63	911
17	0,80	0,63	0,71	835
Média	0,74	0,59	0,65	31835
Média Ponderada	0,74	0,74	0,73	31835

Tabela 10 – Dados por classe para o modelo com 12 canais de Regressão Logística

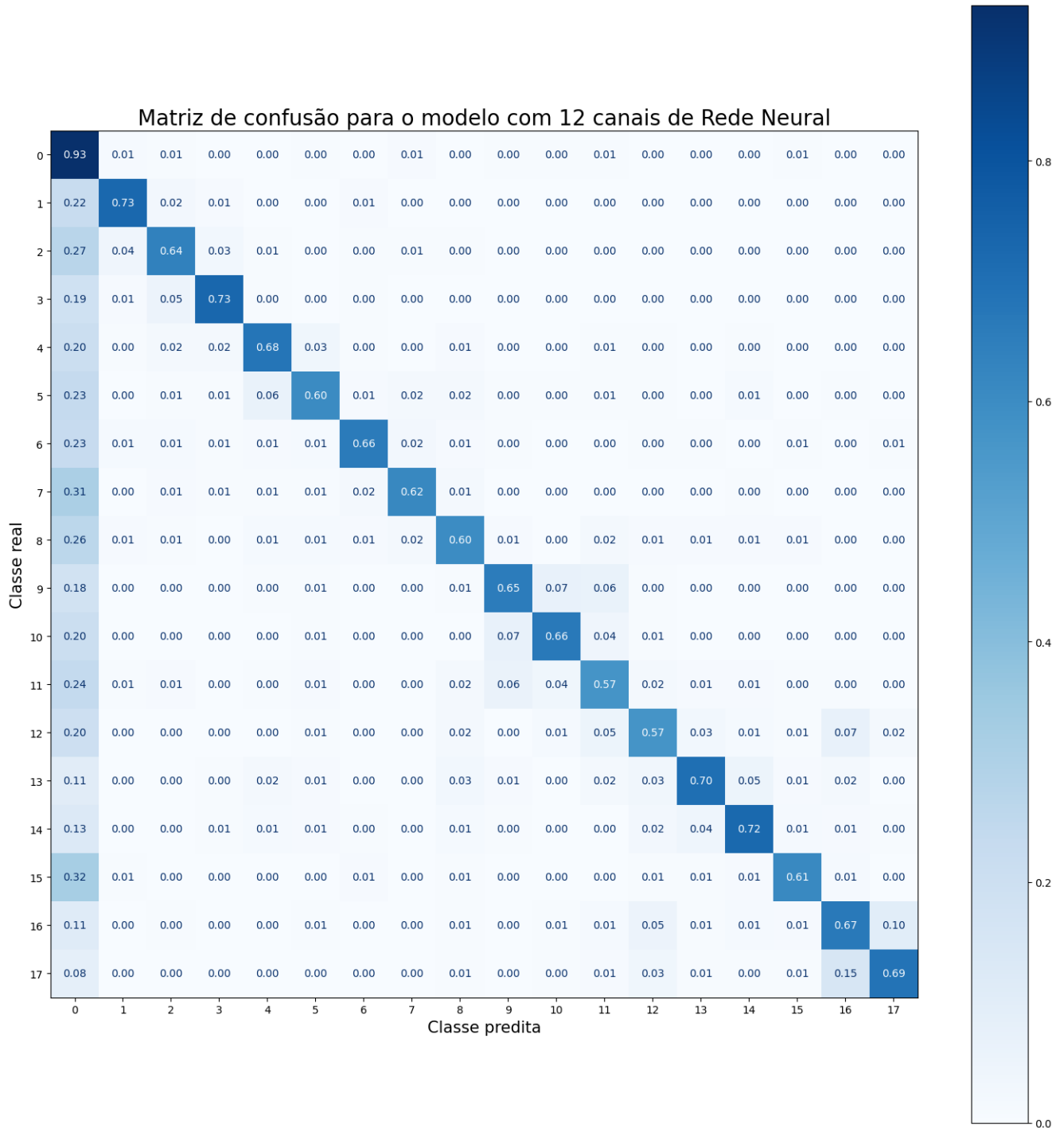


Figura 20 – Matriz de confusão para o modelo com 12 canais de Rede Neural.

Fonte: O autor.

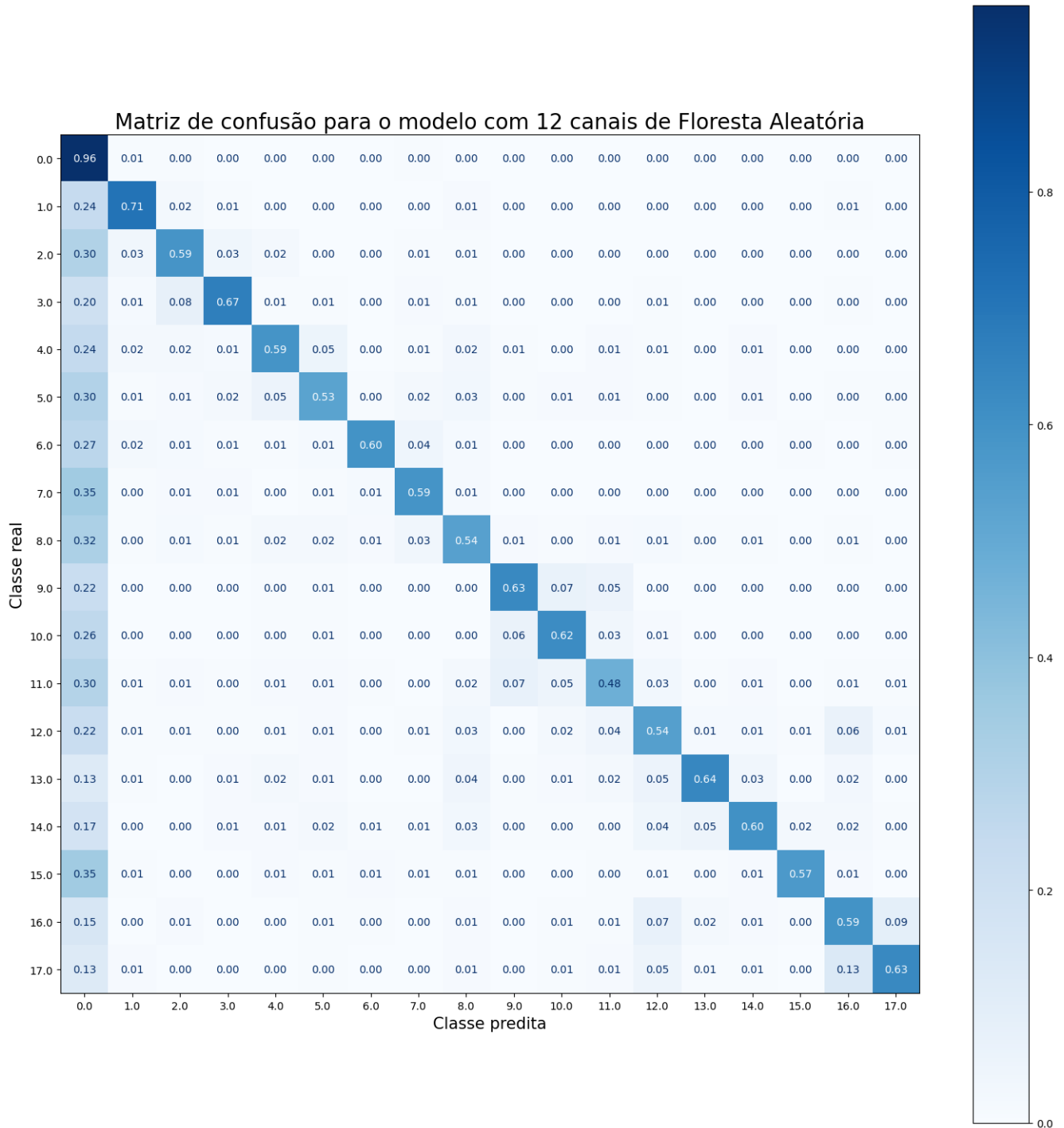


Figura 21 – Matriz de confusão para o modelo com 12 canais de Floresta Aleatória.
Fonte: O autor.

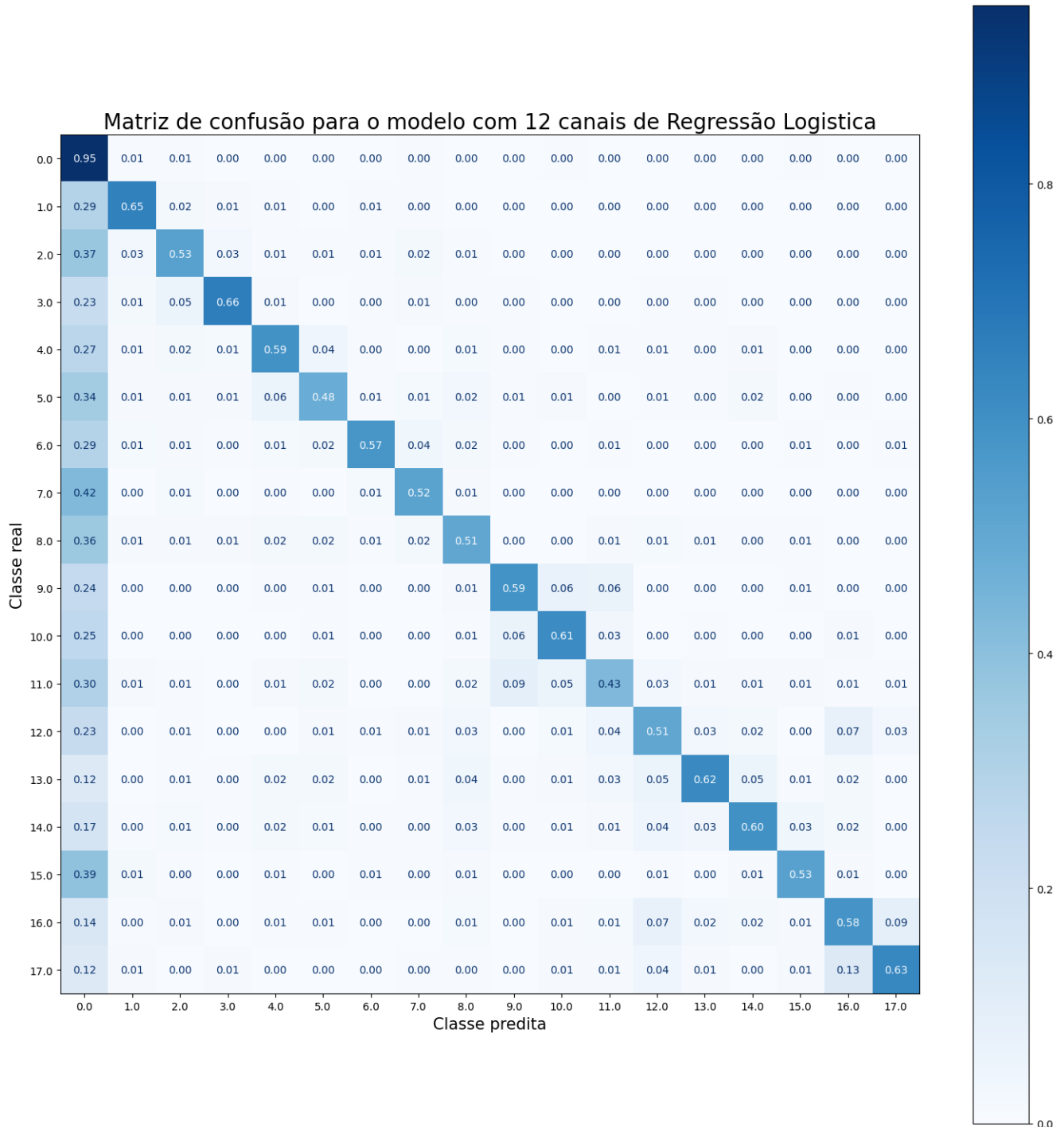


Figura 22 – Matriz de confusão para o modelo com 12 canais de Regressão Logística.

Fonte: O autor.

Posteriormente foram escolhidos aleatoriamente 5 dos 40 voluntários (1,2,33,37 e 40) para ter sua classificação verificada no microcontrolador. Para as redes neurais e regressões logísticas foram encontradas as mesmas classificações tanto no microcontrolador quanto no computador. Já para as florestas aleatórias, houve diferenças nos resultados.

Para analisar as diferenças entre os resultados obtidos, foi calculada a taxa de acerto média dos 5 voluntários no microcontrolador e em Python. Foi calculado também o percentual médio de amostras, por voluntário, que retornam um valor diferente nos dois meios. Esta métrica é calculada, para cada voluntário, pelo número de amostras com

resultados diferentes dividido pelo número total de amostras. A Figura 23 mostra essas três medidas.

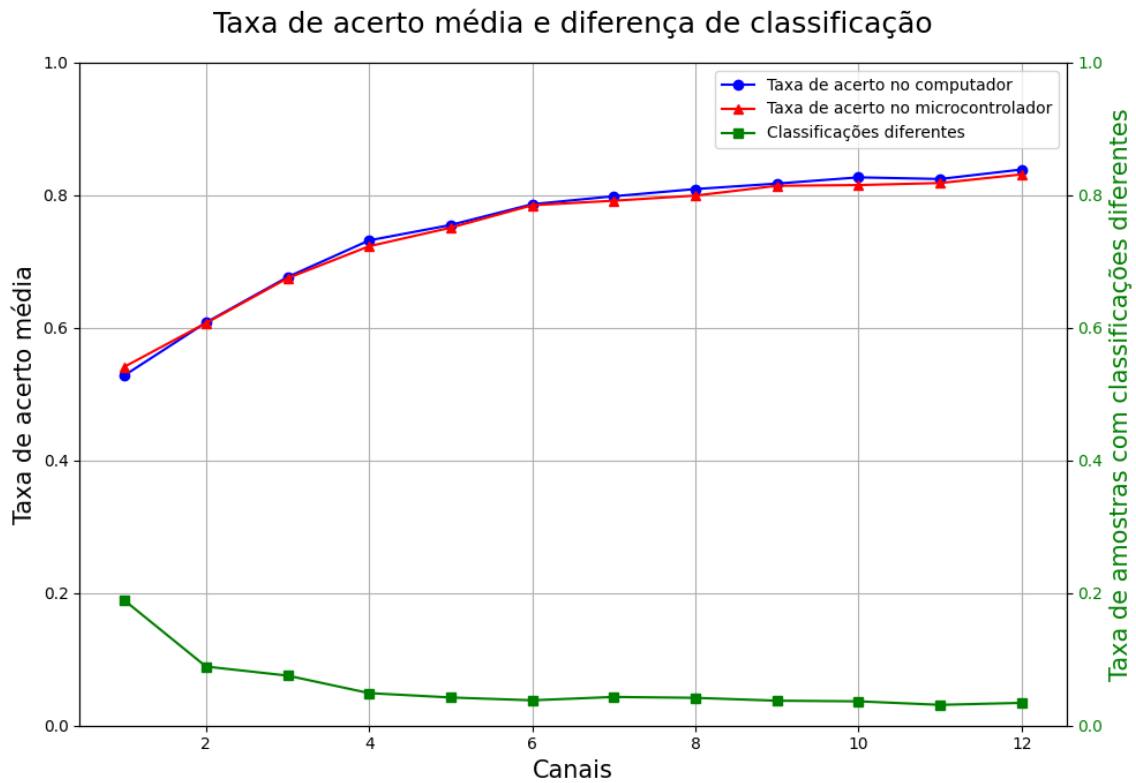


Figura 23 – Comparação entre modelos de floresta aleatória embarcados e modelos no computador
Fonte: O autor.

Ao analisar o gráfico, é possível perceber que há mais amostras divergentes quanto menos entradas há no sistema. As taxas de acerto, entretanto, não apresentam grande variação em nenhum ponto do gráfico.

4.2.2 Latência

Para avaliar a latência, o procedimento foi dividido em duas fases: extração de características e processamento do modelo. A latência dos modelos foi quantificada medindo-se o intervalo desde o momento em que os dados pré-processados são introduzidos no modelo até o instante em que a classificação é fornecida. Foram usados os dados de 5 voluntários, escolhidos aleatoriamente, com o resultado final sendo a média dos valores obtidos.

As figuras 24, e 25 mostram os resultados obtidos. No caso das florestas aleatórias a latência obtida foi inferior a um microssegundo para todas as configurações.

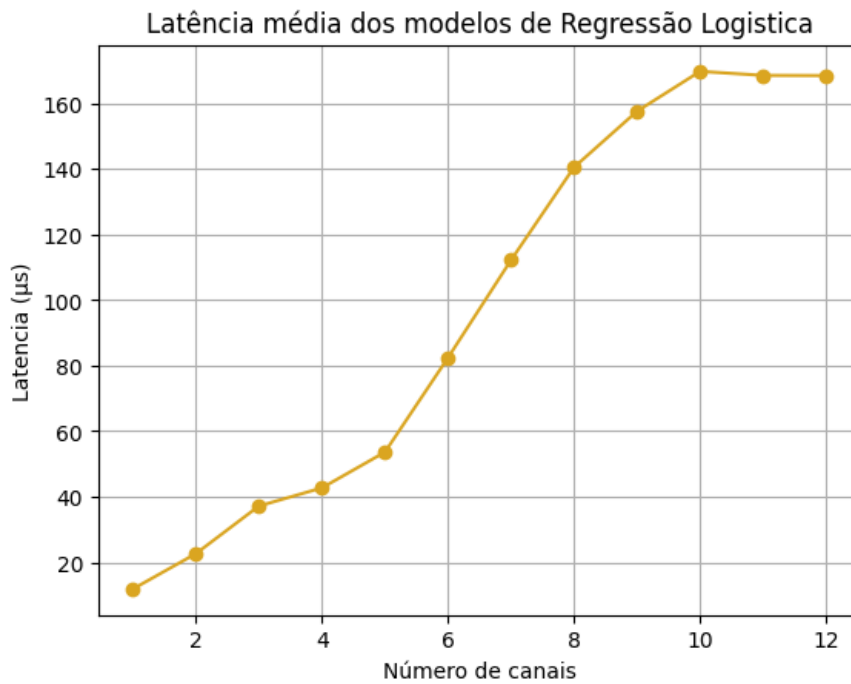


Figura 24 – Latência média dos modelos de Regressão Logística

Fonte: O autor.

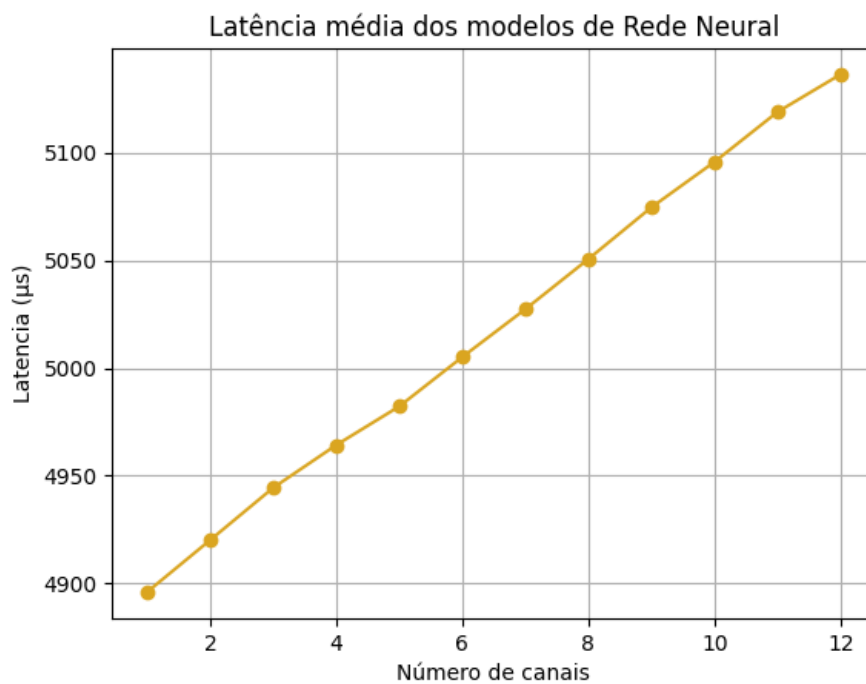


Figura 25 – Latência média dos modelos de Florestas Aleatórias

Fonte: O autor.

A latência da extração das características e normalização foram medidas usando dados de 10 janelas. Foi medido o tempo necessário para extrair as características de 1 a 12 canais e fazer sua normalização. A latência da extração de cada uma das três características utilizadas nos modelos finais, com a normalização já embutida, é apresentada nas figuras 26, 27 e 28.

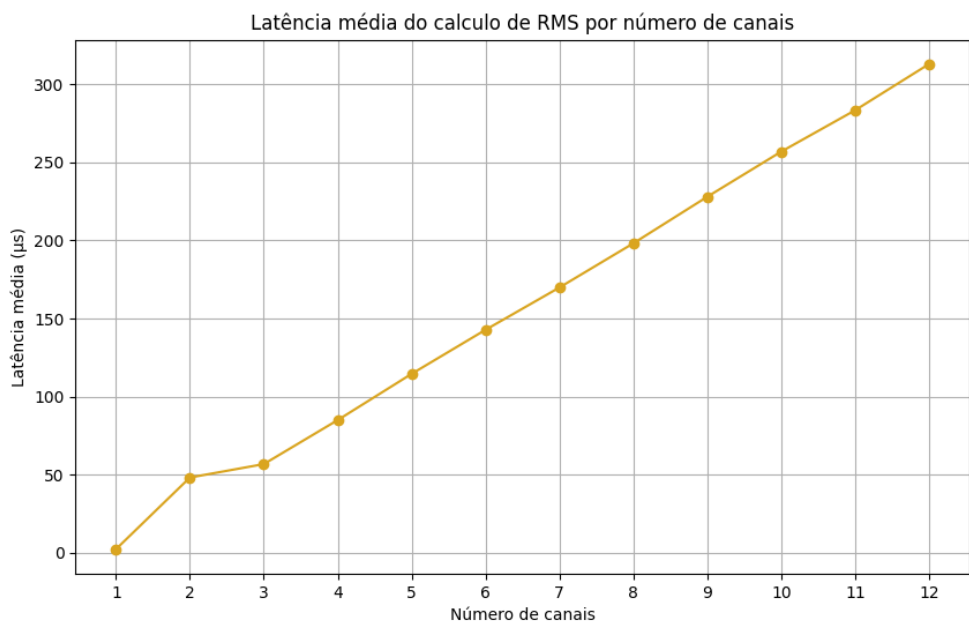


Figura 26 – Latência média do cálculo do valor RMS por número de canais
Fonte: O autor.

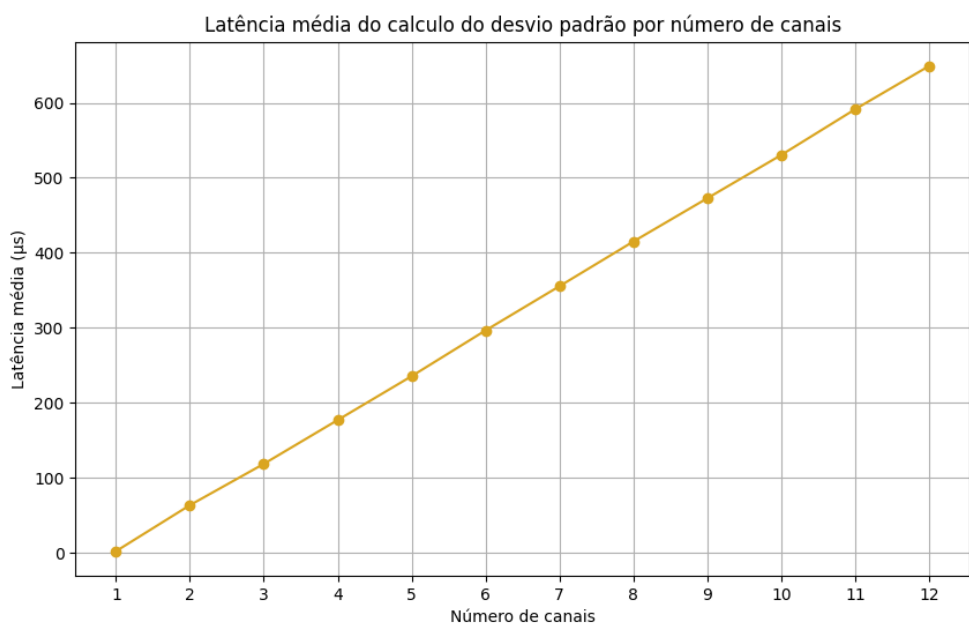


Figura 27 – Latência média do cálculo do desvio padrão por número de canais
Fonte: O autor.

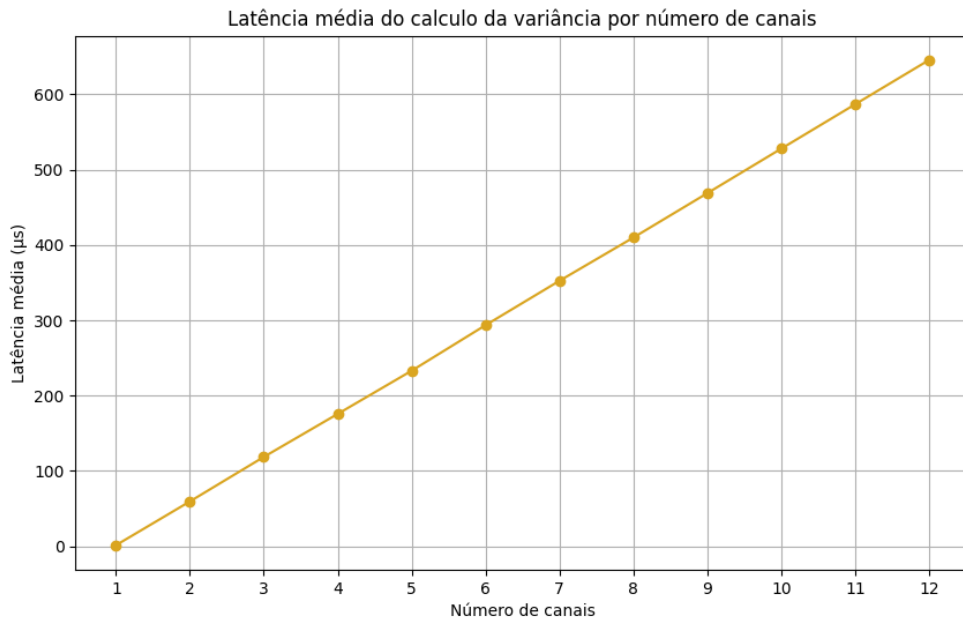


Figura 28 – Latência média do cálculo da variância por número de canais

Fonte: O autor.

A latência do cálculo do RMS foi a menor entre as características chegando a 313 μs para 12 canais. Desvio padrão e variância tiveram uma latência maior, chegando a 649 μs e 645 μs para 12 canais, respectivamente.

Combinando, portanto, a latência dos modelos com suas respectivas características utilizadas tem-se os seguintes resultados, mostrados nas figuras 29, 30 e 31.

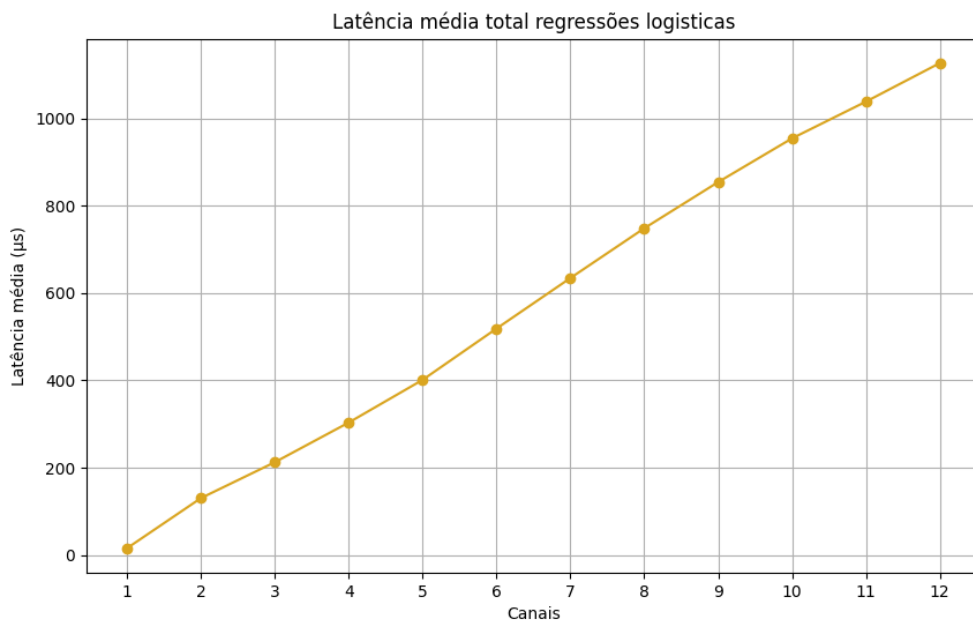


Figura 29 – Latência média total das regressões logísticas

Fonte: O autor.

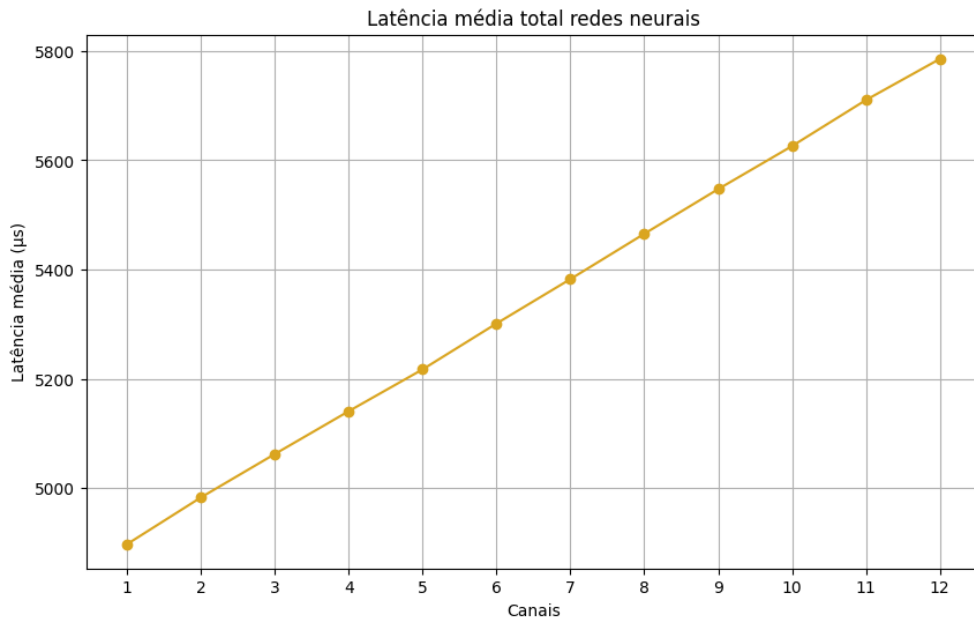


Figura 30 – Latência média total das redes neurais

Fonte: O autor.

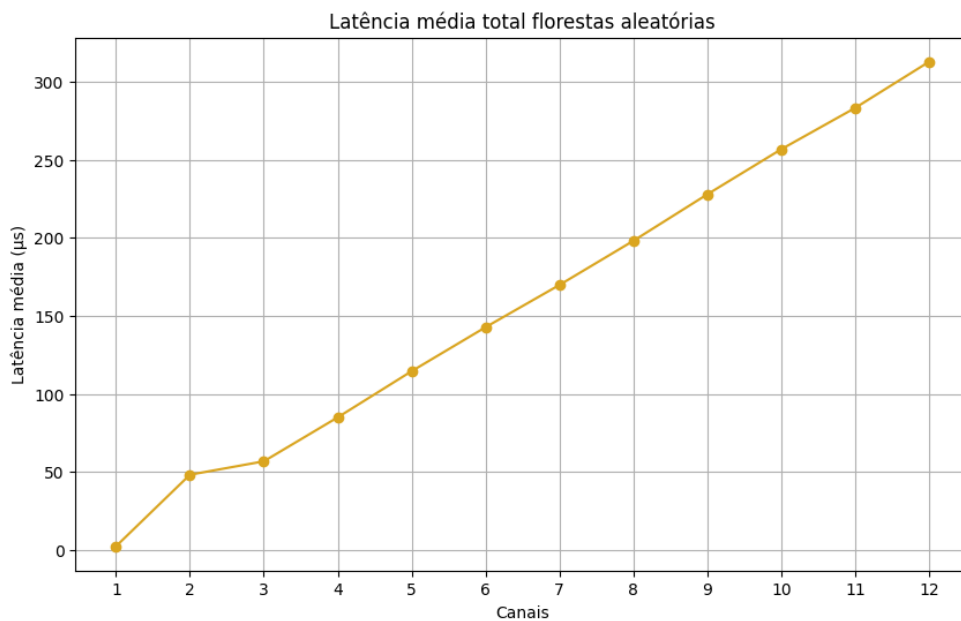


Figura 31 – Latência média total das florestas aleatórias

Fonte: O autor.

Os modelos de florestas aleatórias foram os que tiveram menor latência e as redes neurais as maiores.

4.2.3 Ocupação de memória

Foi verificado o quanto cada modelo ocupa da memória do microcontrolador. Foram considerados os dados de 5 voluntários, escolhidos aleatoriamente, para cada configuração. Foi considerado como ocupação de memória do modelo somente o modelo em si, sem o

código referente a extração de características. As médias de cada resultado são apresentados na Figura 32.

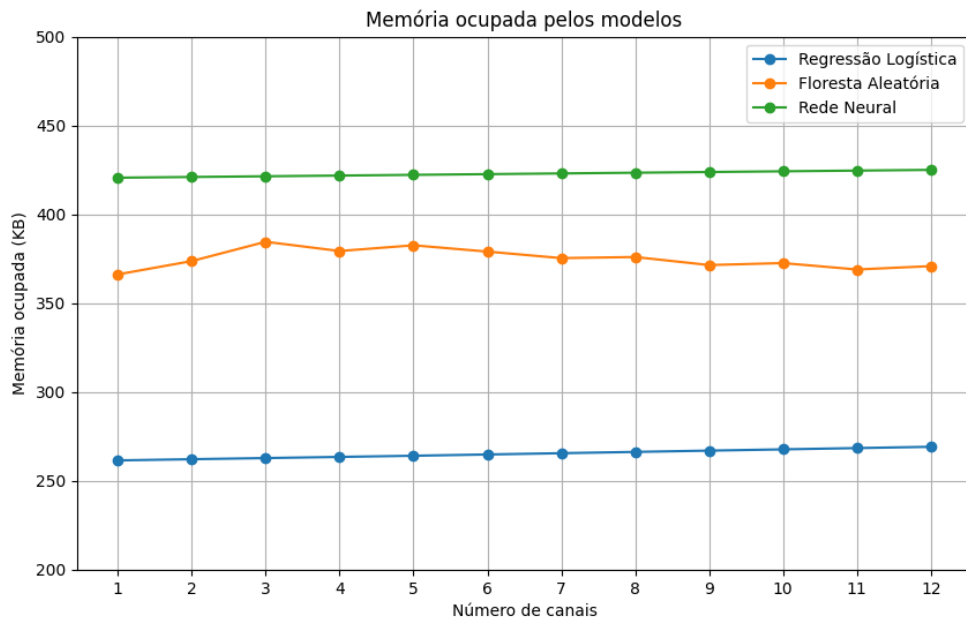


Figura 32 – Memória ocupada pelos modelos

Fonte: O autor.

A ocupação dos modelos de regressão logística foram os que menos ocuparam memória, por conter menos parâmetros, seguido das florestas aleatórias e redes neurais. Também como os hiperparâmetros foram mantidos constantes independentemente do número de canais houve pouca variação da ocupação de memória dentro de cada classe de modelo.

4.3 Comparação com trabalhos da literatura

Pode ser difícil fazer uma comparação direta entre resultados obtidos em trabalhos relacionados. A base de dados utilizada e movimentos utilizados para a classificação podem afetar diretamente no resultado obtido.

Considerando todos os movimentos da base de dados 2 do Nina Pro, com 50 movimentos, (ATZORI *et al.*, 2014) encontrou 75,27% de taxa de acerto, (ZHAI *et al.*, 2016) 77,41% e (CENE *et al.*, 2019) 79,77%, todos considerando o melhor caso encontrado. (GIJSBERTS *et al.*, 2014) utilizando 41 movimentos da base de dados encontrou 77,48% de taxa de acerto.

Neste trabalho foi utilizado somente o exercício A da base de dados 2, com 18 movimentos diferentes (incluindo descanso). O melhor classificador encontrado neste trabalho foi o baseado Floresta Aleatória, com 12 canais, com 78% de taxa de acerto média, considerando 40 voluntários.

A Tabela 11 mostra um compilado dos resultados encontrados. A Tabela 11 mostra as melhores taxas de acerto para cada tipo de modelo, indicando para quantos canais o resultado foi obtido.

Trabalho	Taxa de acerto
(ATZORI <i>et al.</i> , 2014)	75,27%
(ZHAI <i>et al.</i> , 2016)	77.41%
(CENE <i>et al.</i> , 2019)	79.77%
(GIJSBERTS <i>et al.</i> , 2014)	77.48%

Tabela 11 – Taxas de acerto em outros trabalhos

Modelo	Número de canais	Taxa de acerto
Floresta Aleatória	12	78,0%
Regressão Logística	12	74,3%
Rede Neural	12	74,3%

Tabela 12 – Melhores taxas de acerto encontradas por modelo

Em (QIN *et al.*, 2021) foi treinado CNNs (Rede Neural Convolutacional), de 1D e 2D, para realizar a classificação dos movimentos, que foram embarcados em um Arduino Nano. As redes tiveram uma latências de 79-85ms e 132-135ms respectivamente. Os modelos obtidos neste trabalho todos estiveram abaixo destes valores.

5 Conclusões

O objetivo deste trabalho foi avaliar diferentes modelos para a classificação da intenção de movimentos da mão a partir de sinais de sEMG. Os modelos incluíram redes neurais, regressão logística e florestas aleatórias, sendo testados com diversas configurações tanto em computador quanto em um sistema embarcado.

Foi feita seleção de características para os três tipos de modelos, por seleção sequencial e otimização de hiperparâmetros, por uma pesquisa em grade. Os modelos otimizados foram testados variando a quantidade de canais de entrada, sendo adicionados sequencialmente a partir de um ordenamento de prioridade feito previamente. Os modelos foram então embarcados, para validar a taxa de acerto e medir a latência e memória ocupada.

Para cada grupo de modelos os com maior taxa de acerto foram os com o uso de todos os 12 canais, obtendo uma taxa de acerto média de 74,3% (redes neurais), 74,3% (regressão logística) e 78,0% (Florestas Aleatórias). O aumento da taxa de acerto, porém, foi diminuindo ao longo que os canais foram sendo adicionados, podendo indicar que algum valor intermediário possa ser ideal. As menores latências também foram encontradas nos modelos de florestas aleatórias, variando de 2 μs a 312 μs , em função do número de canais.

A partir dos resultados obtidos, é possível concluir que o aumento do número de canais também ocasiona no aumento da latência. Apesar disso, as latências encontradas não são impeditivas para uma implementação em uma prótese mecânica, visto que são significativamente menores que o janelamento de 300ms utilizado.

O sistema proposto conseguiu taxas de acerto próximas a modelos que usam a mesma base de dados, apesar de muitos estudos utilizarem um conjunto maior de movimentos. Os modelos propostos se comportaram de forma similar quando foram embarcados, mostrando ser possível realizar a classificação nesses sistemas.

5.1 Trabalhos futuros

O estudo de classificadores de movimentos a partir de sinais sEMG que possam ser embarcados é fundamental para o avanço da área, permitindo sua implementação em contextos reais que exigem portabilidade.

Outros modelos, maiores e mais robustos, como redes neurais profundas, podem ser estudados. Além disso, microcontroladores com maior desempenho podem ser testados com estes modelos maiores. Estes novos modelos, além dos já estudados neste trabalho, podem ser estudados com outras bases de dados.

O objetivo final destes trabalhos seria a criação de um sistema completo. O sistema completo de uma prótese controlada por sEMG consistiria desde a aquisição dos dados a classificação dos movimentos e criação de um atuador.

Referências Bibliográficas

- ANAM, K.; SWASONO, D. I.; MUTTAQIN, A. Z.; HANGGARA, F. S. Finger movement regression with myoelectric signal and deep neural network. In: *2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE)*. [S.l.: s.n.], 2019. p. 187–191.
- ARIYANTO, M.; CAESARENDRA, W.; MUSTAQIM, K. A.; IRFAN, M.; PAKPAHAN, J. A.; SETIAWAN, J. D.; WINOTO, A. R. Finger movement pattern recognition method using artificial neural network based on electromyography (emg) sensor. In: *2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT)*. [S.l.: s.n.], 2015. p. 12–17.
- ATZORI, M.; GIJSBERTS, A.; CASTELLINI, C.; CAPUTO, B.; HAGER, A.-G. M.; ELSIG, S.; GIATSIDIS, G.; BASSETTO, F.; MÜLLER, H. Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Scientific Data*, v. 1, n. 1, p. 140053, Dec 2014. ISSN 2052-4463. Disponível em: <<https://doi.org/10.1038/sdata.2014.53>>.
- BENEDETTI, M.; EBENBICHLER, G.; LOISEL, P.; ROY, S. Clinician's view: dynamic emg. *IEEE Engineering in Medicine and Biology Magazine*, v. 20, n. 6, p. 33–37, 2001.
- BETHGE, J.; BARTZ, C.; YANG, H.; CHEN, Y.; MEINEL, C. Meliusnet: An improved network architecture for binary neural networks. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. [S.l.: s.n.], 2021. p. 1439–1448.
- BOSCHMANN, A.; KAUFMANN, P.; PLATZNER, M.; WINKLER, M. Towards multi-movement hand prostheses: Combining adaptive classification with high precision sockets. In: *Proceedings of the 2nd European Conference Technically Assisted Rehabilitation*. [S.l.: s.n.], 2009.
- CENE, V. H.; BALBINOT, A. Upper-limb movement classification through logistic regression semg signal processing. In: *2015 Latin America Congress on Computational Intelligence (LA-CCI)*. [S.l.: s.n.], 2015. p. 1–5.
- CENE, V. H.; BALBINOT, A. Enhancing the classification of hand movements through semg signal and non-iterative methods. *Health and Technology*, Springer Science and Business Media LLC, v. 9, n. 4, p. 561–577, abr. 2019. ISSN 2190-7196. Disponível em: <<http://dx.doi.org/10.1007/s12553-019-00315-6>>.
- CENE, V. H.; TOSIN, M.; MACHADO, J.; BALBINOT, A. Open database for accurate upper-limb intent detection using electromyography and reliable extreme learning machines. *Sensors*, v. 19, n. 8, 2019. ISSN 1424-8220. Disponível em: <<https://www.mdpi.com/1424-8220/19/8/1864>>.
- CRISWELL, E. Cram jr. *Cram's introduction to surface electromyography*, p. 340–376, 2011.
- GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. [S.l.]: "O'Reilly Media, Inc.", 2022.

GIJSBERTS, A.; ATZORI, M.; CASTELLINI, C.; MULLER, H.; CAPUTO, B. Movement error rate for evaluation of machine learning methods for sEMG-based hand movement classification. *IEEE Trans. Neural Syst. Rehabil. Eng.*, Institute of Electrical and Electronics Engineers (IEEE), v. 22, n. 4, p. 735–744, jul. 2014.

JAMAL, M. Z. Signal acquisition using surface emg and circuit design considerations for robotic prosthesis. In: NAIK, G. R. (Ed.). *Computational Intelligence in Electromyography Analysis*. Rijeka: IntechOpen, 2012. cap. 18. Disponível em: <<https://doi.org/10.5772/52556>>.

JAMES DANIELA WITTEN, T. H. R. T. M. *An Introduction to Statistical Learning*. [S.l.]: Springer, 2013.

JOSE, N.; RAJ, R.; ADITHYA, P. K.; SIVANADAN, K. S. Classification of forearm movements from semg time domain features using machine learning algorithms. In: *TENCON 2017 - 2017 IEEE Region 10 Conference*. [S.l.: s.n.], 2017. p. 1624–1628.

KINGSFORD, C.; SALZBERG, S. L. What are decision trees? *Nature Biotechnology*, Springer Science and Business Media LLC, v. 26, n. 9, p. 1011–1013, set. 2008. ISSN 1546-1696. Disponível em: <<http://dx.doi.org/10.1038/nbt0908-1011>>.

LIU, W. Natural user interface- next mainstream product user interface. In: *2010 IEEE 11th International Conference on Computer-Aided Industrial Design Conceptual Design 1*. [S.l.: s.n.], 2010. v. 1, p. 203–205.

MAHESH, B. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR)*. [Internet], v. 9, n. 1, p. 381–386, 2020.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, v. 1, n. 1, p. 32, 2003.

ORTIZ-CATALAN, M.; BRÅNEMARK, R.; HÅKANSSON, B. BioPatRec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms. *Source Code for Biology and Medicine*, Springer Science and Business Media LLC, v. 8, n. 1, abr. 2013. Disponível em: <<https://doi.org/10.1186/1751-0473-8-11>>.

OSKOEI, M. A.; HU, H. Support vector machine-based classification scheme for myoelectric control applied to upper limb. *IEEE Transactions on Biomedical Engineering*, v. 55, n. 8, p. 1956–1965, 2008.

PIZZOLATO, S.; TAGLIAPIETRA, L.; COGNOLATO, M.; REGGIANI, M.; MÜLLER, H.; ATZORI, M. Comparison of six electromyography acquisition setups on hand movement classification tasks. *PloS one*, Public Library of Science San Francisco, CA USA, v. 12, n. 10, p. e0186132, 2017.

PULLMAN, S.; GOODIN, D.; MARQUINEZ, A.; TABBAL, S.; RUBIN, M. utility of surface emg: Report of the therapeutics and technology assessment subcommittee of the american academy of neurology. *Neurology*, v. 55, n. 2, p. 177, 2000.

QIN, S.; ZHANG, J.; SHEN, H.; WANG, Y. Arm movements recognition by implementing cnn on microcontrollers. In: *2021 9th International Conference on Control, Mechatronics and Automation (ICCA)*. [S.l.: s.n.], 2021. p. 171–176.

- ROKACH, L.; MAIMON, O. Decision trees. In: _____. *Data Mining and Knowledge Discovery Handbook*. Boston, MA: Springer US, 2005. p. 165–192. ISBN 978-0-387-25465-4. Disponível em: <https://doi.org/10.1007/0-387-25465-X_9>.
- ROTH, W.; SCHINDLER, G.; KLEIN, B.; PEHARZ, R.; TSCHIATSCHEK, S.; FRÖNING, H.; PERNKOPF, F.; GHAHRAMANI, Z. *Resource-Efficient Neural Networks for Embedded Systems*. arXiv, 2020. Disponível em: <<https://arxiv.org/abs/2001.03048>>.
- SAHA, S. S.; SANDHA, S. S.; SRIVASTAVA, M. Machine learning for microcontroller-class hardware: A review. *IEEE Sensors Journal*, v. 22, n. 22, p. 21362–21390, 2022.
- SHARMA, S.; SHARMA, S.; ATHAIYA, A. Activation functions in neural networks. *Towards Data Sci*, v. 6, n. 12, p. 310–316, 2017.
- TAM, S.; BOUKADOUM, M.; CAMPEAU-LECOURS, A.; GOSSELIN, B. A fully embedded adaptive real-time hand gesture classifier leveraging hd-semg and deep learning. *IEEE Transactions on Biomedical Circuits and Systems*, v. 14, n. 2, p. 232–243, 2020.
- TOSIN, M. C.; MACHADO, J. C.; BALBINOT, A. sEMG-based upper limb movement classifier: Current scenario and upcoming challenges. *Journal of Artificial Intelligence Research*, AI Access Foundation, v. 75, p. 83–127, set. 2022. Disponível em: <<https://doi.org/10.1613/jair.1.13999>>.
- WIERING, M. A.; OTTERLO, M. V. Reinforcement learning. *Adaptation, learning, and optimization*, Springer, v. 12, n. 3, p. 729, 2012.
- ZHAI, X.; JELFS, B.; CHAN, R. H. M.; TIN, C. Short latency hand movement classification based on surface emg spectrogram with pca. In: *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. [S.l.: s.n.], 2016. p. 327–330.