UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL – UFRGS

FACULDADE DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

JOÃO PEDRO JACQUES HOSS

Comparação de técnicas de aprendizado de máquina para a classificação de gestos para um sistema de interface homem-máquina sem contato

PORTO ALEGRE

OUTUBRO DE 2022

JOÃO PEDRO JACQUES HOSS

Comparação de técnicas de aprendizado de máquina para a classificação de gestos para um sistema de interface homem-máquina sem contato

Projeto de diplomação apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica, pelo Curso de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul – UFRGS

Orientador: Prof. Dr. Tiago Oliveira Weber

PORTO ALEGRE

OUTUBRO DE 2022

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL – UFRGS

FACULDADE DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Comparação de técnicas de aprendizado de máquina para a classificação de gestos para um sistema de interface homem-máquina sem contato

Banca Examinadora:

_____

Prof. Dr. Tiago Oliveira Weber
Orientador
Universidade Federal do Rio Grande do Sul

_____

Prof. Dr. Alexandre Balbinot
Universidade Federal do Rio Grande do Sul

_____

Prof. Dr. Altamiro Amadeu Susin
Universidade Federal do Rio Grande do Sul

“Não acho que quem ganhar ou quem perder,
nem quem ganhar ou perder, vai ganhar ou perder.
Vai todo mundo perder.”
- ROUSSEFF, Dilma.

Resumo

O reconhecimento de gestos de mão é um aspecto importante na comunicação interpessoal e com avanços tecnológicos na área de interação homem-máquina, tornou-se possível utilizar esses gestos como uma maneira do usuário interagir e controlar dispositivos. Nos últimos anos, diversos ramos da indústria veem desenvolvendo aplicações que utilizam o reconhecimento de gestos, como o ramo automotivo, ramo de robótica e ramo de realidade virtual, por exemplo. Levando em consideração estes últimos avanços importantes, esse trabalho explora a coleta de bases de dados, a análise dos dados coletados, o treinamento de diversos modelos de aprendizado de máquina sobre os dados até o desenvolvimento de uma aplicação simples utilizando um sistema de interface homem-máquina controlada por gestos. Para isso, duas bases de dados foram coletadas onde cada amostra representa o esqueleto da mão do voluntário, obtido utilizando a ferramenta *MediaPipe*, formado por 21 pontos. Além disso, para ambas as bases de dados, foram realizadas coletadas de amostras a uma distância de 50 cm, 75 cm e 100 cm da *webcam*. Uma base de dados segue um projeto de experimento com aleatorização na coleta de dados tendo a participação de 5 voluntários para coletar um total de 450 amostras para cada um dos 10 gestos propostos. A outra não seguiu um projeto com aleatorização na coleta e contou com a participação de um único voluntário para coletar 4.500 amostras por gesto. Sobre essas bases de dados, foram criados modelos inteligentes capazes de classificar os gestos utilizando oito técnicas de aprendizado de máquina, sendo elas regressão logística, classificador *Bayes* ingênuo, máquina de suporte de vetor, árvore de decisão, floresta aleatória, *XGBoost*, rede neural simples e uma *MLP*. Sobre cada técnica foi realiza um processo de otimização de hiper parâmetros. Dessa forma, a técnica que resultou nos modelos com o melhor desempenho foi a técnica *XGBoost*. O modelo *XGBoost* treinado sobre a base de dados aleatorizada obteve uma taxa de acerto de 99,425 ± 0,135% e um valor *F1-score* de 99,375 ± 0,175%, e o modelo treinado sobre os dados não aleatorizado obteve 99,400 ± 0,069% e 99,355 ± 0,075% respectivamente como taxa de acerto e *F1-score*. Por fim, foi desenvolvido uma simples aplicação que utilize esse modelo onde o usuário consegue realizar comandos de controle em um computador.

Palavras chaves: Otimização de hiper parâmetros; aprendizado de máquina; *XGBoost;* interface homem-máquina.

Abstract

Hand gestures recognition is an important aspect in interpersonal communication and with technological advances in the area of human-machine interaction, it has become possible to use these gestures as a way for the user to interact and control devices. In recent years, several branches of the industry have been developing applications that use gesture recognition, such as the automotive branch, robotics and virtual reality branch, for example. Considering these last important advances, this work explores the collection of databases, the analysis of the collected data, the training of different machine learning models on the data until the development of a simple application using a gesture-controlled human-machine interface system. For this, two databases were collected where each sample represents the skeleton of the volunteer's hand, obtained using the MediaPipe tool, formed by 21 points. In addition, for both databases, samples were collected at a distance of 50 cm, 75 cm and 100 cm from the webcam. A database follows an experiment design with randomization in data collection with the participation of 5 volunteers to collect a total of 450 samples for each of the 10 proposed gestures. The other did not follow a project with randomization in the collection and had the participation of a single volunteer to collect 4,500 samples per gesture. On these databases, intelligent models were created capable of classifying gestures using eight machine learning techniques, namely logistic regression, naive Bayes classifier, support vector machine, decision tree, random forest, *XGBoost*, simple neural network and a *MLP*. A hyperparameter optimization process was performed on each technique. Thus, the technique that resulted in the models with the best performance was the *XGBoost* technique. The *XGBoost* model trained on the randomized database obtained a hit rate of 99.425 ± 0.135% and an F1-score of 99.375 ± 0.175%, and the model trained on the non-randomized data obtained 99.400 ± 0.069% and 99.355 ± 0.075 % respectively as hit rate and F1-score. Finally, a simple application was developed that uses this model where the user can perform control commands on a computer.

Keywords: Hyper parameters optimization; machine learning; *XGBoost*; man-machine interface.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

fps	*Frames per Second*
MLP	*Multilayer Perceptron*
CNNs	*Convolutional Neural Networks*
GPU	*Graphics Processing Unit*
GB	*GigaByte*
API	*Application Programming Interface*
RNAs	*Redes Neurais Artificiais*

Sumário

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

A interação entre homem e máquina é um tópico que vem sendo estudado cada vez mais ao longo dos últimos anos. Apesar da utilização do teclado e *mouse* serem amplamente adotados, diversos outros tipos de interfaces vêm sendo propostos e alguns recebem um grande grau de adoção. O maior exemplo deste tipo foi a adoção em massa de celulares que utilizam tecnologia por toque. Esta foi uma proposta que revolucionou totalmente indústrias inteiras e, também, a maneira com que as pessoas interagem com *smartphones* e dispositivos móveis. O interessante desta interface é o quanto ela foi capaz de ampliar o leque de possibilidades de ações que o usuário tem a disposição. Ao invés de tentar substituir o propósito de um teclado e *mouse*, a tecnologia por toque focou em fornecer comandos que não eram possíveis, ou fáceis, utilizando o teclado e *mouse*.

Ao longo dos anos, uma área que vem ganhando relevância é a que estuda interfaces homem-máquina sem que o usuário interaja diretamente com um dispositivo, como o caso do *mouse*, teclado e tela *touchscreen* (DEVINEAU *et al*, 2018; RAFID *et al*, 2022; PASQUALE *et al*, 2016; GUO *et al*, 2021). Há muitas tentativas em reproduzir a revolução como foi a da tecnologia por toque para os dispositivos móveis nesta área. Uma interface que vem se tornou amplamente popular é a interface utilizando comandos de voz do usuário, com aplicações como o serviço *Alexa* da *Amazon* (LOPATOVSKA *et al*. 2018).

Além da interface por comandos de voz, outra que vem recebendo destaque é a interface por gestos, onde o usuário realiza gestos com suas mãos para executar ações num sistema operacional. Além de ser uma interface que sempre teve muita visibilidade em filmes e obras de ficção científica, ela tem uma vantagem de poder ser intuitiva e fácil de ser adotada. A sua popularidade na comunidade científica vem crescendo juntamente com os avanços feitas em técnicas de visão computacional e processamento de imagens para identificar objetos, padrões e movimentos.

Diversas aplicações vêm introduzindo essa interface de gestos nos mais diversos ramos. Há trabalhos que desenvolvem aplicações na área de telemedicina, possibilitando o médico a realizar cirurgias remotas controlando equipamentos utilizando apenas seus gestos (QI *et al*. 2021). Também, esse tipo de interface está ganhando espaço, também, no setor automotivo, com o desenvolvimento de paneis de controle controlado por gestos (ZENGELER *et al*. 2018), assim como em aplicações que contemple realidade virtual e realidade aumentada (SAGAYAM, K. M.; HEMANTH, D. J. 2017).

1.2 OBJETIVO GERAL

Visando colaborar com a comunidade científica no desenvolvimento de novas tecnologias, este projeto de diplomação tem como objetivo implementar e comparar o desempenho de algumas técnicas de aprendizado de máquina populares na tarefa de reconhecimento de gestos de mão para suportar o uso de uma interface homem-máquina sem contato que se beneficie desta capacidade de reconhecimento.

1.3 OBJETIVOS ESPECÍFICOS

Para alcançar com êxito o objetivo geral deste projeto, foram definidos alguns objetivos secundários:

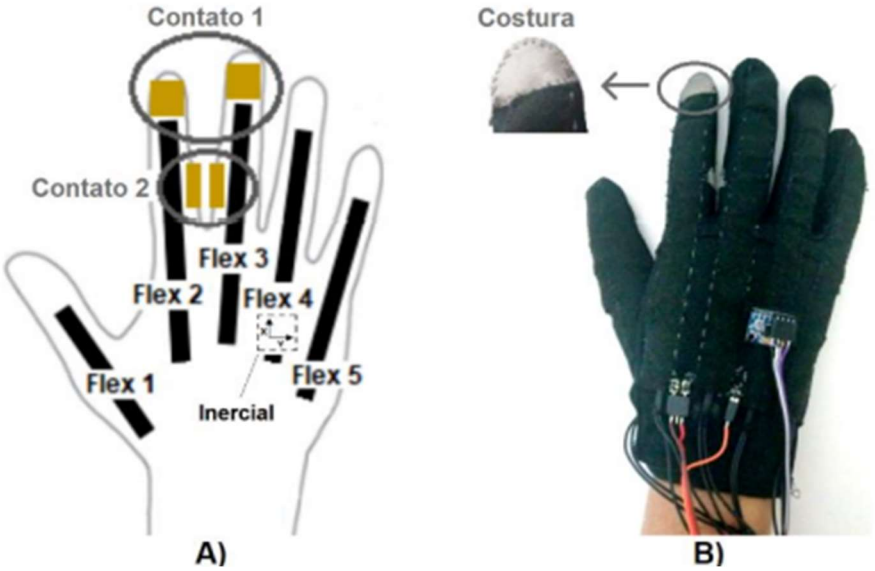
I. Realizar a coleta de bases de dados com os gestos de interesse para o sistema;
II. Analisar quantitativamente as características das bases coletadas;
III. Comparar diferentes técnicas de criar modelos inteligentes capazes de reconhecer os gestos;
IV. Comparar o desempenho dos modelos treinados sobre cada base de dados;
V. Implementar um sistema simples de interface homem-máquina utilizando o modelo inteligente que obteve o melhor desempenho.

2 TRABALHOS RELACIONADOS

A área de reconhecimento de gestos tem sido objeto de diversos estudos recentemente. Há uma expectativa nas aplicações que esta área pode vir a proporcionar e ela tem se beneficiado muito com o desenvolvimento em técnicas de aprendizagem de máquina dos últimos anos. Assim, cada vez mais é possível desenvolver sistemas capaz de distinguir gestos com precisão e confiabilidade.

Ao longo dos anos, diferentes abordagens foram adotadas para resolver problemas que envolvam a classificação de gestos de mãos. Trabalhos como Dias (2020) escolheram desenvolver uma luva com diversos sensores para coletar dados e criar um modelo simples baseado em *MLP* para classificar gestos sobre estes dados. Dias utilizou essa luva para classificar gestos segundo a Língua Brasileira de Sinais. Na Figura 1, é possível visualizar o protótipo construído.

Figura 1: Luva Instrumentada com 5 sensores de flexão, 1 sensor inercial e 2 sensores de contato, em que (a) apresenta o posicionamento dos sensores, (b) apresenta a foto da luva e a imagem ampliada do tecido condutivo costurado sobre a luva



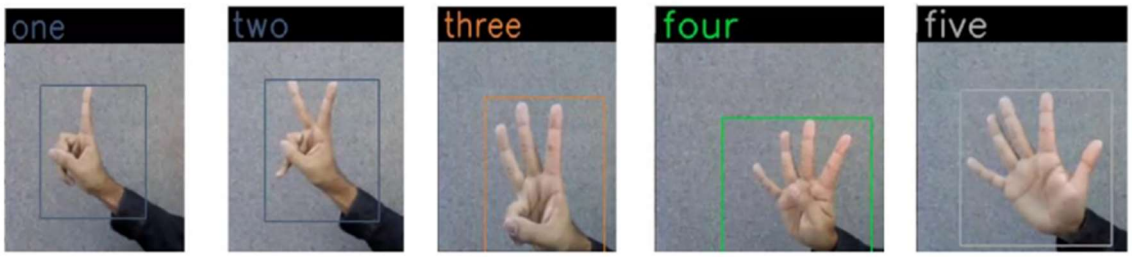
Fonte: DIAS, T. S., 2020.

Embora este tipo de abordagem possa ter um desempenho satisfatório em experimentos e consiga utilizar modelos de classificação muito mais simples, há uma clara desvantagem para fins de aplicação devido à dependência de um equipamento especialmente projetado. Esse tipo de desvantagem pode impossibilitar essa abordagem para diversas áreas de aplicações. Para contornar este tipo de problema, outros trabalhos optam por buscar uma solução utilizando equipamentos que muitas vezes já estão disponíveis, como é o caso de fazer o reconhecimento utilizando imagens capturadas de *webcams* de computador ou outras câmeras digitais.

Em Mujahid *et al* (2021) foi proposto um modelo capaz de analisar imagens e identificar gestos em tempo real. O modelo é baseado em técnicas de aprendizado profundo de redes neurais, seguindo topologias *CNNs*. Na Figura 2, pode-se observar o resultado qualitativo do modelo proposto. Embora essa abordagem seja mais simples e tenha uma complexidade de

implementação menor, pois necessita menos equipamentos e dispositivos, a complexidade na etapa de modelagem e classificação pode aumentar drasticamente.

Figura 2: Resultado do modelo classificador de gestos.



Fonte: Mujahid et al, 2021.

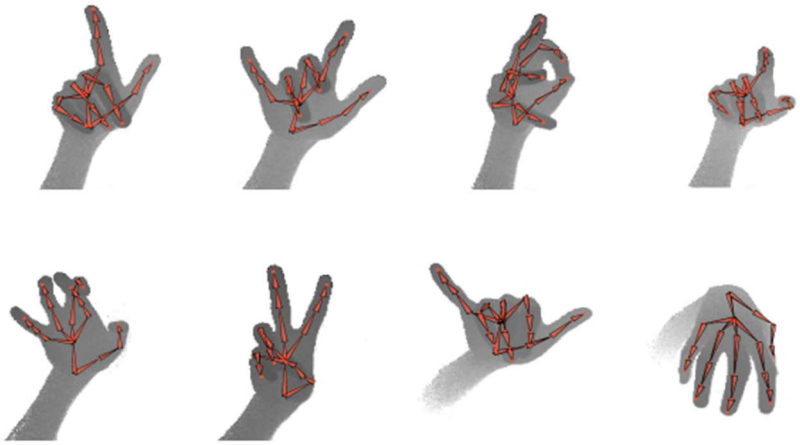
A complexidade desta abordagem decorre da necessidade de construção de um modelo que seja capaz de analisar uma imagem, que é um formato de dados muito denso e com muita informação, e prontamente identificar o gesto. Isso requer uma maior capacidade de abstração de um único modelo, fazendo com que o modelo seja maior, tenha um treinamento mais intenso e, muitas vezes, tenha um tempo de resposta maior. No caso de Mujahid, o treinamento de seu modelo demorou mais de 26 horas, utilizando uma *GPU* de 24 *GB* de memória. O seu *dataset* consistia em 216 imagens e 5 classes diferentes.

Uma maneira de reduzir este problema é restringir a resolução das imagens que o modelo irá usar. O mais comum é realizar a compressão da imagem antes de entregá-la ao modelo, como feito em Chung *et al* (2019) e Bhame *et al* (2014), o que pode causar uma perda de informação nos dados, dificultando o processo de reconhecimento do modelo.

Outra abordagem que vem ganhando relevância consiste em separar o processo de reconhecimento em duas partes: a primeira parte utiliza um modelo que analisa uma imagem e identifica pontos de interesses na mão (chamado de esqueleto) e a segunda parte utiliza apenas os dados destes pontos para classificar os gestos. Dessa forma é possível reduzir consideravelmente a complexidade dos modelos, uma vez que o modelo que analisa a imagem tem uma tarefa mais simples, e o modelo de reconhecimento de gestos pode passar a utilizar uma quantidade muito menor de dados.

Em Sridhar *et al* (2015), foi desenvolvido um modelo para abstrair o esqueleto de uma mão baseando-se em modelos de florestas aleatórias e mistura gaussiana. Desse modo, foi possível evitar a construção de um modelo extenso e complexo, e foi possível atingir uma inferência muito mais rápida, conseguindo operar em vídeos de até 50 *fps*. Na Figura 3 é possível visualizar o resultado obtido pelo modelo.

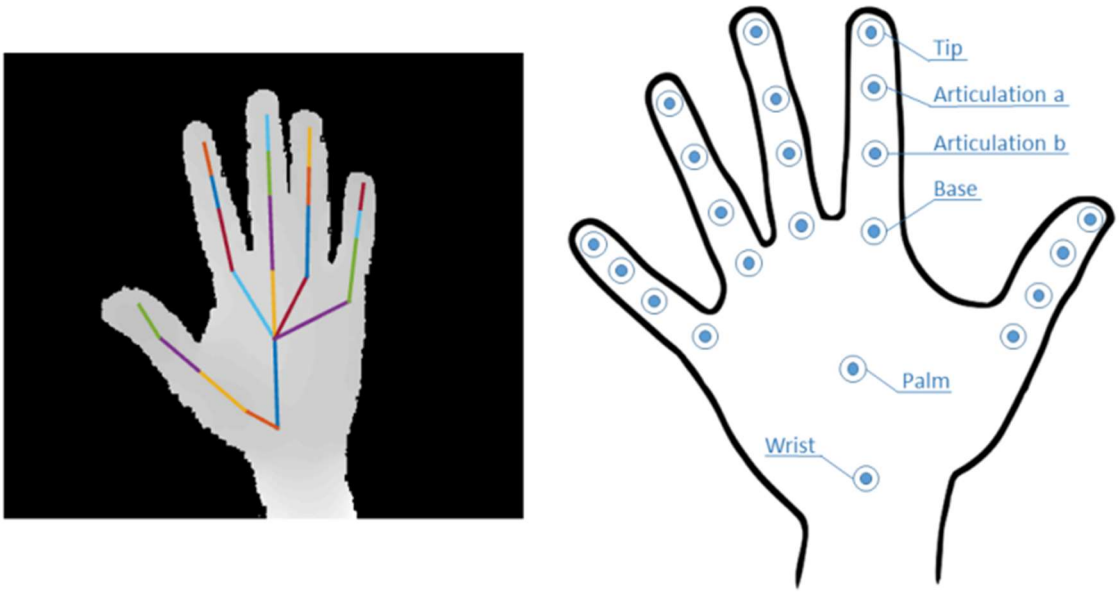
Figura 3: Resultados qualitativos da modelo.



Fonte: Sridhar et al, 2015. Adaptado.

Em trabalhos cujo foco seja conseguir reconhecer gestos com esse tipo de dado processado, é muito comum utilizar conjuntos de dados previamente coletados, como o caso do *dataset* SHREC'17. Se tornou padrão identificar 22 pontos para constituir o esqueleto da mão, podendo ser observados na Figura 4.

Figura 4: Exemplo qualitativo dos dados do dataset SHREC'17.

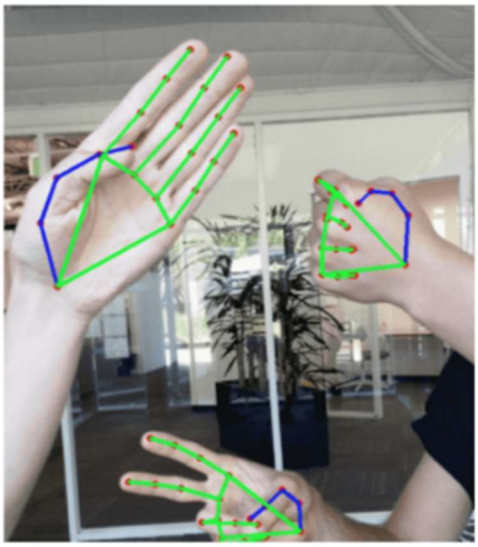


Fonte: Smedt et al, 2017.

Por outro lado, em vários trabalhos e aplicações, ao invés de utilizar um *dataset* já coletado, tem se tornado comum utilizar modelos previamente treinados, para capturar o esqueleto da mão numa imagem. Em particular, um modelo que tem se tornado popular é um modelo fornecido pelo *Google Research*, através de uma *API* chamada *MediaPipe.* Essa *API* consiste em diversos modelos, não apenas para mãos, mas também para rostos e outros objetos. No modelo para mãos, ela identifica os mesmos 21 pontos (não identifica o ponto da

palma da mão como no caso do *dataset* SHREC'17), retornando suas localizações nos eixos *x*, *y* e *z*. Na Figura 5 é possível observar o resultado qualitativo deste modelo. É interessante apontar que esse modelo é capaz de capturar várias mãos numa mesma imagem.

Figura 5: Resultado do modelo de mãos para identificar o esqueleto.



Fonte: Zhang *et al,* 2020.

Esse tipo de ferramenta é útil para trabalhos que desejam fazer a sua própria aquisição de dados para treinar seus modelos. Em Sung *et al* (2021), esse modelo de mãos do *MediaPipe* foi utilizado para criar um classificador de gestos em tempo real. O time de pesquisadores utilizou o modelo para coletar 7307 imagens de 6 classes diferentes. Com essas imagens, foi proposto um modelo baseado em redes neurais com múltiplas camadas que obteve uma taxa de acerto de 87,9%.

Ademais, usando o *MediaPipe*, é possível criar sistemas de controle criativos, que não necessariamente envolvam o uso de aprendizagem de máquina para classificar gestos. Em Islam *et al* (2022) foi desenvolvido um jogo, onde o usuário controla a direção de um carro em movimento utilizando gestos da mão para a esquerda e direita. Eles classificaram estes gestos simplesmente relacionando a matriz da posição dos 21 pontos capturados pelo modelo.

Em Devineau *et al* (2018), foi desenvolvido um modelo baseado em redes neurais convolucionais (*CNNs*) para reconhecer gestos sobre o *dataset* SHREC'17 que obteve uma taxa de acerto de até 91,28% e um tempo de inferência na ordem de 10^{-5} segundos, sem o uso de uma *GPU*. Já em Liu et al (2015), é proposto um modelo baseado em redes de hierárquicas de auto atenção com uma acurácia, sobre o mesmo *dataset*, de 95%.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 INTERFACE HOMEM-MÁQUINA

Interface (ou Interação) Homem-Máquina (*IHM*) é a maneira com que o usuário interage com um sistema computacional. Através dela, o usuário tem a capacidade de controlar que tipo de ações um sistema deve executar (Ramaswamy *et al*, 2014). É importante que uma *IHM* tenha funcionalidade e que seja de fácil uso por parte do usuário. Quando uma interface é bem projetada, ela é compreensível, agradável, controlável e previsível.

Os objetivos de uma *IHM* são os de produzir sistemas usáveis, seguros e funcionais. Esses objetivos podem ser resumidos como desenvolver ou melhorar a segurança, utilidade, efetividade e usabilidade de sistemas que incluem computadores. Nesse contexto, o termo sistemas se refere não somente ao *hardware* e *software*, mas a todo o ambiente que usa ou é afetado pelo uso de tecnologia computacional.

3.2 APRENDIZADO DE MÁQUINA

O aprendizado de máquina, do inglês *Machine Learning* (*ML*), ramo da inteligência artificial, procura resolver problemas do mundo real através de algoritmos programáticos com a capacidade de aprender a partir de dados, conseguindo alcançar uma solução de forma automática sem ser explicitamente instruídos. O aprendizado de máquina tem como objetivo identificar padrões e tomar decisões de maneira automática. Para isso, esse processo de aprendizado deve ser capaz de criar, a partir da experiência passada (conjunto de treino), uma função matemática capaz de relacionar as características de um problema com sua solução.

Trata-se de uma ferramenta poderosa que possui diversos algoritmos que podem apresentar diferentes desempenhos de acordo com cada caso. No contexto de aprendizado de máquina, cada experiência é um exemplo, ou amostra, sobre o problema em questão. Um conjunto de exemplos ou amostras é chamado de base de dados.

Segundo Russell e Norvig (2009), os algoritmos de aprendizado de máquina podem ser classificados em três tipos:

1. **Supervisionado:** quando os dados são rotulados, ou seja, o algoritmo tem conhecimento da saída correta de cada entrada;

2. **Não Supervisionado:** quando os dados de análise não são rotulados, logo, o algoritmo não conhece a saída correta das entradas;

3. **Aprendizado por Reforço:** quando a aprendizagem é feita por sistemas de recompensas, onde o algoritmo é capaz de verificar o impacto de suas decisões e minimizar suas perdas.

Esse projeto de diplomação visa explorar técnicas de aprendizado de máquina supervisionado. Assim, os outros dois tipos de aprendizado de máquina mencionados anteriormente não serão abordados.

3.3 APRENDIZADO SUPERVISIONADO

As tarefas de aprendizado supervisionado de máquina costumam se dividir em duas categorias: tarefas de regressão e tarefas de classificação. As tarefas de regressão têm como objetivo estimar um resultado dentro de uma faixa de possíveis valores. Por sua vez, as tarefas de classificação focam em analisar os dados fornecidos e estimar uma classificação para a situação observada.

Em Russell e Norvig (2009) o aprendizado supervisionado é definido da seguinte forma: dado um conjunto $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \subseteq X \times Y$ de treinamento com n pares de exemplos de entrada e saída, onde cada y_i foi gerado por uma função desconhecida $y_i = f(x_i)$. O objetivo é encontrar uma função h que se aproxime da função verdadeira f, através de um processo iterativo que consiga ajustar os coeficientes (parâmetros) da função h de modo a prover uma menor diferença entre as saídas desejadas.

Além da definição iterativa de parâmetros, modelos de aprendizagem de máquina também contam com a definição de hiper parâmetros. Esses hiper parâmetros são definidos previamente e são mantidos estáticos durante o processo de aprendizagem. Normalmente esses hiper parâmetros são responsáveis por definir como é feito a buscar por parâmetros ideias para a situação, e a definição dos hiper parâmetros adequados pode ser uma tarefa empírica não trivial e extensiva.

3.3.1 Regressão Logística

A regressão logística é um tipo de modelagem estatística capaz de determinar a probabilidade de algum determinado evento ocorrer. Em tarefas de classificação, ela também é capaz de determinar a probabilidade de uma amostrar ser de alguma determinar classe. Essa técnica é comumente utilizada para situação que apresentem duas classes, porém também pode ser utilizada em tarefas que possuam três classes ou mais. El Sanharai e Naudet (2013) concluem que a regressão logística é uma técnica de análise potente e que permite obter uma quantificação da relação entre as variáveis de entrada e o resultado esperado de maneira consistente.

A resposta do modelo de regressão logística pode ser definida pela Equação 1.

$$f(x) = \frac{e^{g(x)}}{1 + e^{g(x)}} \tag{1}$$

Onde a função $g(x)$ representa o peso β_n das variáveis de entrada x_n da tarefa de classificação, segundo a Equação 2.

$$g(x) = \sum_{n=1}^{l} \beta_n x_n \tag{2}$$

Existem diversos métodos matemática de calcular os pesos β_n para cada variável x_n. Além disso, é comum utilizar um fator de regularização para ajudar a evitar *overfitting* durante o treinamento do modelo. Geralmente são utilizadas técnicas de regularização *L1* ou *L2*.

3.3.2 Classificador Bayes Ingênuo

O classificador Bayes ingênuo é um modelo probabilístico baseado no teorema de Bayes. De acordo com Brito (2016), esse tipo de classificador é muito utilizado em aplicações de aprendizado de máquina devido a sua simples compreensão e implementação. O classificador tem como suposição que as variáveis de entrada do modelo são independentes. Essa suposição dá origem ao nome “ingênuo” do classificador.

O processo de treinamento desse classificador gera um peso para cada variável de entrada para cada uma das classes presentes na tarefa. Durante o processo de inferência, por sua vez, o modelo soma os pesos de cada variável de entrada para cada classe e a classe que resultar na maior probabilidade é definida como a resposta do classificador.

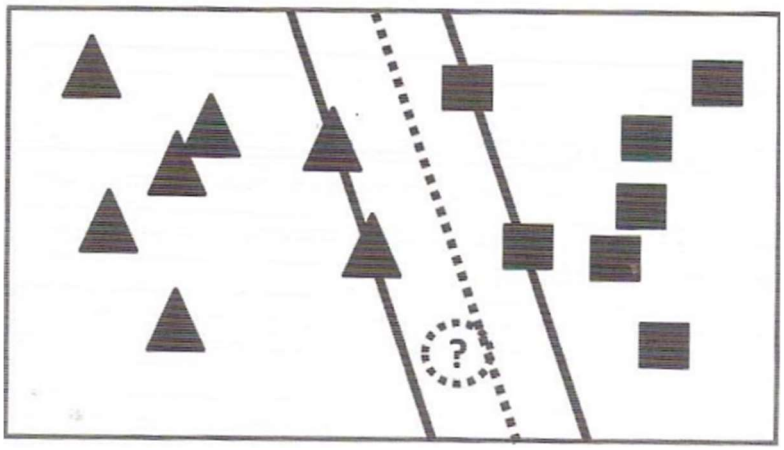
Segundo Alteryx Inc (2018), esse tipo de classificador funciona muito bem, mesmo quando há poucos dados no conjunto de treinamento devido ao fato do classificador ser parametrizado pela média e variância de cada variável de maneira independente.

3.3.3 Máquina de Suporte de Vetor

A técnica de máquina de suporte de vetor, também conhecida como *Support Vector Machine (SVM)*, é uma técnica de classificação que consistem em encontrar a separação das classes de interesse num hiperplano de variáveis de entrada. Segundo Carvalho e Lorena (2003), esse tipo de algoritmo consegue gerar modelos com uma grande capacidade de generalização e robustez.

Na Figura 6 é possível observar um exemplo que representa como esse classificador distingue duas classes de interesse, representados por triângulos e quadrados. Nesse exemplo é possível observar que o classificador define duas linhas divisórias no plano de variáveis de entrada que delimitam a resposta do modelo. Além disso, podemos perceber uma amostra entre as duas linhas divisórias. A classificação desta amostra é definida pela menor distância até cada uma das divisões. No caso, esta amostra é classificada como um triângulo.

Figura 6: Representação gráfica do classificador.



Fonte: Amaral, 2016.

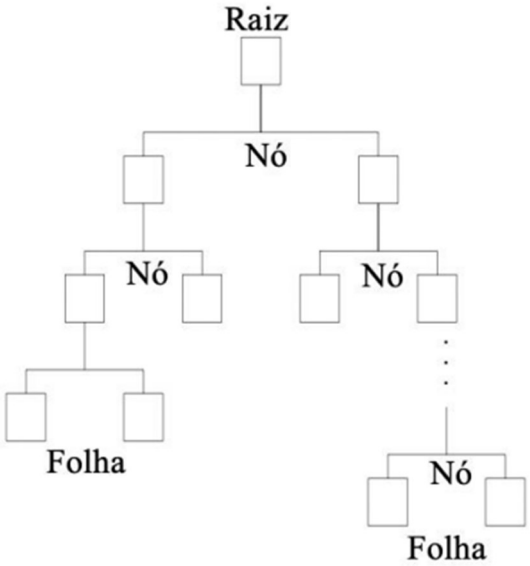
Existem diversos métodos matemáticos de estabelecer essas linhas divisórias para cada classe. Cada método implementa o uso de uma função, neste caso também chamada de *kernel*, de natureza diferente, como uma função linear ou uma função polinomial, por exemplo.

3.3.4 Árvores de Decisão

O modelo de Árvore de decisão é um modelo estatístico, amplamente usado em problemas de aprendizado de máquina, tanto em tarefas supervisionadas como em não supervisionadas. Segundo Russel e Norvig (2009), esse modelo tem como objetivo representar uma função que pode ter um vetor com várias entradas, em um único sinal de saída. Os valores de saídas da função são representados pelas folhas das árvores, que vão sendo criadas conforme ocorre o processo de treinamento.

O processo de treinamento consiste na constante subdivisão de uma base de treinamento em classes menores, chamadas *nós*. Cada nó representa um teste sobre o valor percorrendo o processo lógico da árvore, sendo direcionado por sucessivos testes até cair num resultado final, ou *folha*. A quantidade de nós de uma árvore, assim como a quantidade de folhas depende tanto da complexidade da base de treinamento e do problema específico quanto da topologia do modelo. Na Figura 7 podemos observar a estrutura de uma árvore de decisão genérica.

Figura 7: Estrutura de uma árvore de decisão genérica.



Fonte: O autor, 2022.

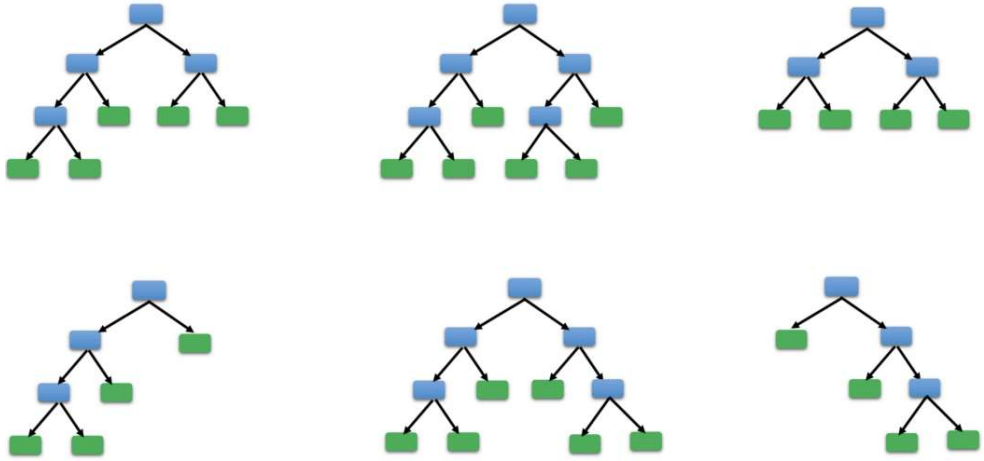
Esse tipo de modelo conta com diversos hiper parâmetros que definem como a árvore será estruturada. Entre eles, a estratégia usada para realizar a divisão dos nós e o critério de avaliação sobre a qualidade dessas divisões são hiper parâmetros comuns de serem explorados.

3.3.5 Floresta aleatória

Um modelo baseado em floresta aleatória expande a ideia das árvores de decisões e faz com que várias árvores trabalhem cooperativamente ou concorrentemente. Esse tipo de modelo foi desenvolvido em 2001, segundo Breiman (2009). Uma floresta aleatória consiste em várias árvores de decisões, onde, durante o processo de treinamento, o conjunto de treino é repartido e utilizado para treinar diversas árvores de decisões com hiper parâmetros aleatórios.

Para realizar uma predição, a amostra é propagada para todas as árvores e é realizado uma votação. O resultado passa a ser a classe mais votada. A Figura 8 ilustra a estrutura de uma floresta aleatória.

Figura 8: Estrutura de uma floresta aleatória genérica.



Fonte: O autor, 2022.

3.3.6 *XGBoost*

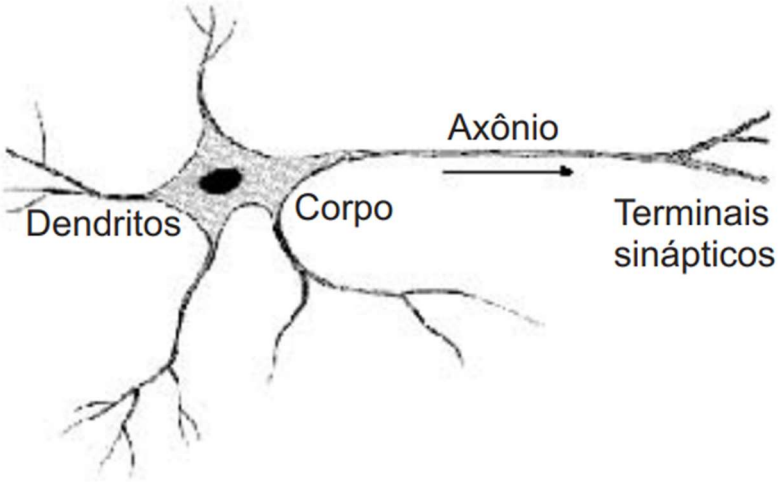
O algoritmo *XGBoost*, ou *Extreme Gradient Boosting*, desenvolvido em 2016, por Chen e Guestrin, é uma das implementações mais eficientes de florestas aleatórias. Esse modelo impulsiona as árvores de decisões, utilizando a técnica *Gradient Boosting,* fazendo com que a cada iteração realizada pelo algoritmo o modelo tente corrigir o erro cometido na iteração anterior.

Dessa forma, é possível atingir um desempenho excepcionalmente bom fazendo com que as árvores fracas, com pouca capacidade preditiva tenham um impacto menor no resultado final. A sua operação consiste em ponderar as observações colocando-se um maior peso nas instâncias difíceis de classificar.

3.3.7 Redes Neurais Artificiais

De acordo com Haykin (1994), as redes neurais artificiais (*RNAs*) são sistemas adaptativos que possuem um mecanismo de processamento de informação semelhante aos de neurônios biológicos. Desta forma, *RNAs* buscam simular formas de aprendizagem presentes em estruturas biológicas, tais como os neurônios de um cérebro humano, representado na Figura 9.

Figura 9: Representação simplificada de um neurônio.



Fonte: Ferneda, 2006.

Segundo Braga *et al* (2000) a estrutura de um neurônio pode ser subdividida entre três seções principais: o copo celular, os dendritos e o axônio. A função dos dendritos é captar informações provenientes do meio externo ou de outros neurônios e conduzi-las ao corpo celular. O corpo celular, por sua vez, reúne as diferentes informações fornecidas pelos dendritos, e após processá-las gera um sinal constante que é conduzido por toda extensão do axônio. O ponto de contato entre a terminação axônica de um neurônio e o dendrito de outro é chamada de sinapse. Assim, os dendritos que estiverem em conexão sináptica com axônio de outros neurônios perceberão o sinal emitido.

Com isso, de acordo com Haykin (2001), uma *RNA* se assemelha com dois aspectos básicos do funcionamento de um neurônio humano, sendo eles:

- O conhecimento é adquirido pela rede a partir de seu ambiente, por intermédio do processo de aprendizagem;
- Forças de conexão entre neurônios (pesos sinápticos) são utilizadas para armazenar o conhecimento adquirido.

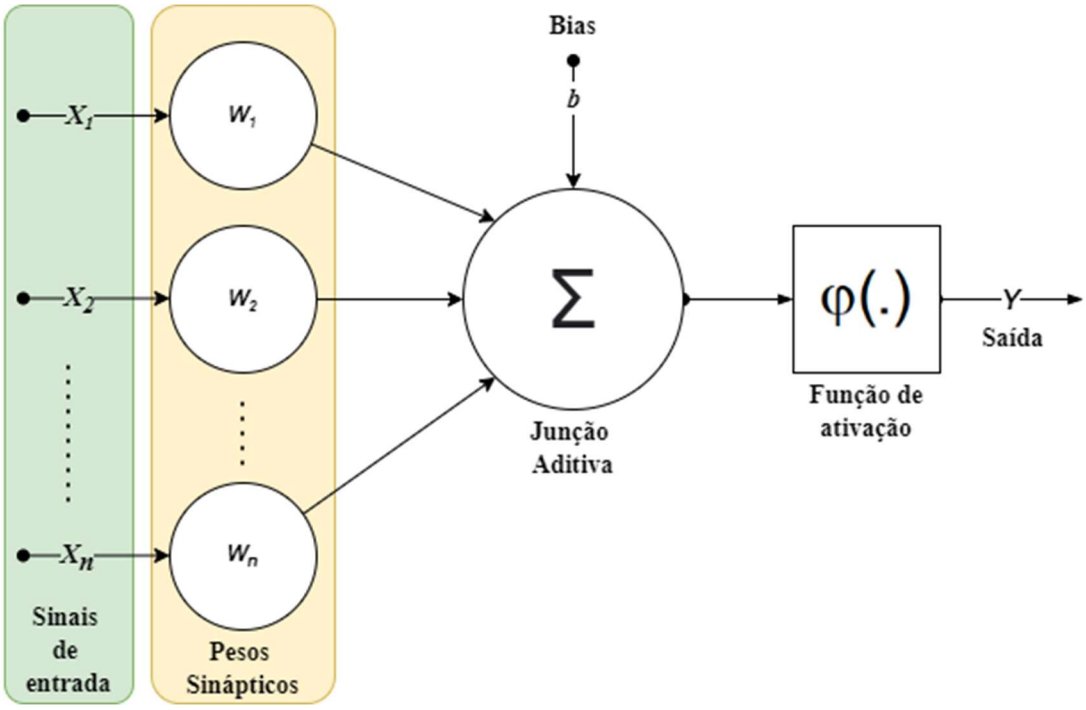
Além disso, a estrutura de uma rede neural é dividida em duas partes: os parâmetros e os hiper parâmetros. Os parâmetros são apenas os pesos sinápticos de cada neurônio, que são ajustados em cada iteração de durante o treinamento da rede. Os hiper parâmetros, por sua vez, não estão sujeitos a modificações durante o treinamento.

A configuração de todos os hiper parâmetros de uma rede neural é chamada de *topologia*, ou *arquitetura*. No caso de redes neurais, alguns hiper parâmetros importantes são: a quantidade de camadas ocultas, a quantidade de neurônios em cada camada, a função de ativação, o algoritmo de otimização e a taxa de aprendizagem do algoritmo de otimização, por exemplo.

3.3.7.1 Estrutura de um Neurônio Artificial

As redes neurais são compostas por diversos neurônios artificiais, que são a unidade básica de processamento dessa estrutura. Cada neurônio responde a uma quantidade de sinais de entrada, x_n, que são ponderados por pesos w_n. Cada neurônio possui uma função de ativação que determina o comportamento do seu sinal de saída. A Figura 10 representa um modelo de neurônio em que o funcionamento se dá pela entrada *x*, junto com um viés (*bias*), *b* é ponderado pelos pesos ω e depois somados.

Figura 10: Modelo artificial de um neurônio.



Fonte: O autor, 2022.

O viés *b* favorece ou limita a possibilidade de ativação do neurônio, enquanto as entradas x_n e os pesos w_n são valores vetoriais, em que os pesos representam o processo sináptico que amplifica cada um dos sinais de entrada. A soma destes termos é obtida através da Equação 3.

$$Z = \omega_n{}^T x_n + b \tag{3}$$

Essa soma então é condicionada por uma função de ativação ϕ. Essa função de ativação é responsável por definir o comportamento da reação do neurônio aos sinais de entrada. Esse comportamento pode ter diversas características, podendo ser linear ou não linear (SHARMA, V; RAI, S; DEV, A. 2012). É comum que as funções de ativação restrinjam a saída para faixas como [0, 1] ou [-1, 1]. O resultado desse condicionado é o sinal de resposta do neurônio artificial a um conjunto de variáveis de entrada e pode ser demonstrado pela Equação 4.

$$y = \phi_{(z)} = \phi_{(\omega_n{}^T x_n + b)} \quad (4)$$

3.3.7.2 Aprendizado de redes neurais

O processo de treinamento, ou de aprendizagem, das redes neurais se dá em duas etapas: a etapa de propagação direta do sinal de entrada até gerar uma predição, ou *Feedforward*, e a etapa de propagação retrógrada do erro da predição, ou *Backpropagation*.

3.3.7.2.1 Propagação Direta ou Feedforward

Durante o processo de propagação direta, ou *Feedforward*, os sinais de entrada x_n da rede são propagados através da camada de entrada, passando pelas camadas escondidas, até gerar um sinal de saída y_n. Esse sinal de saída pode ser expresso pela Equação 5.

$$y_n = \phi^{(2)} \cdot (\omega^{(2)} \phi^{(1)} \cdot \left(\omega^{(1)} x_n + b^{(1)}\right) + b^{(2)}) \quad (5)$$

Onde $\phi^{(i)}$ é a função de ativação da camada *i*.

3.3.7.2.2 Propagação retrógrada ou Backpropagation

Após cada iteração de propagação direta dos sinais de entrada, é possível realizar a etapa de propagação retrógrada dos erros de predições. Esse processo tem como objetivo fazer um ajuste dos pesos de cada parâmetro treinável da rede neural, de modo a garantir o menor erro possível, de acordo com a métrica de erro utilizada como métrica de aprendizagem (RUMERLHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. 1986). Essa operação é representada na Equação 6.

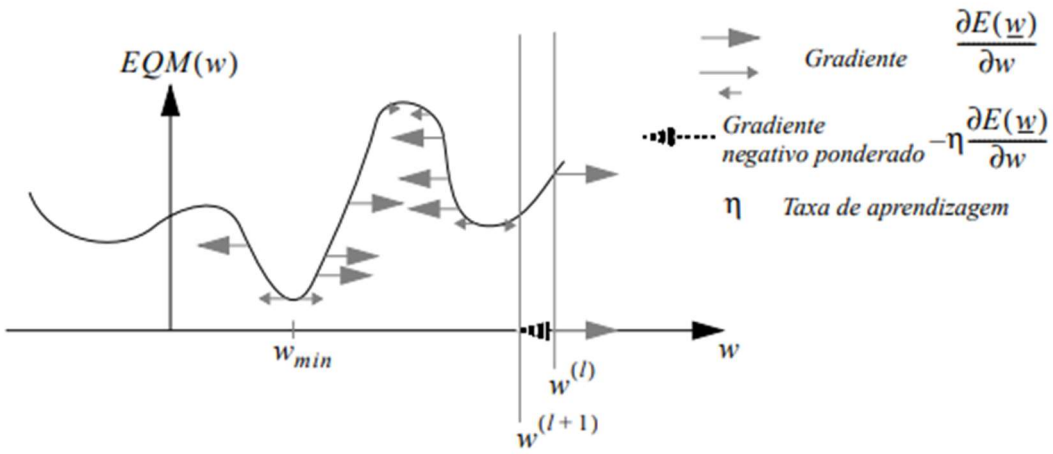
$$\omega_{i+1} = \omega_i + n e_i x_i \quad (6)$$

Onde ω_{i+1} é o peso do neurônio atualizado, ω_i é o peso anterior, n é a taxa de aprendizado do algoritmo de otimização, e_i é o erro e x_i é o sinal de entrada (VEDALDI, A.; LENC, K. 2015). A atualização do peso do neurônio é dada por um algoritmo otimizador que visa, de maneira iterativa, minimizar o erro nas predições da rede. Existem vários algoritmos diferentes, no entanto, todos se baseiam na técnica de descida de gradiente.

3.3.7.2.2.1 Descida de Gradiente

A técnica de descida do gradiente é uma ferramenta padrão para otimizar funções complexas interativamente dentro de um programa de computador. Seu objetivo é: dada alguma função arbitrária, encontrar um mínimo. Para alguns pequenos subconjuntos de funções – aquelas que são convexas – há apenas um único mínimo que também acontece de ser global. Para as funções mais complexas, pode haver muitos mínimos, então a maioria dos mínimos são locais. A Figura 11 mostra a ideia principal da técnica da descida de gradiente.

Figura 11: Gráfico da descida de gradiente.



Fonte: Rauber, 2016.

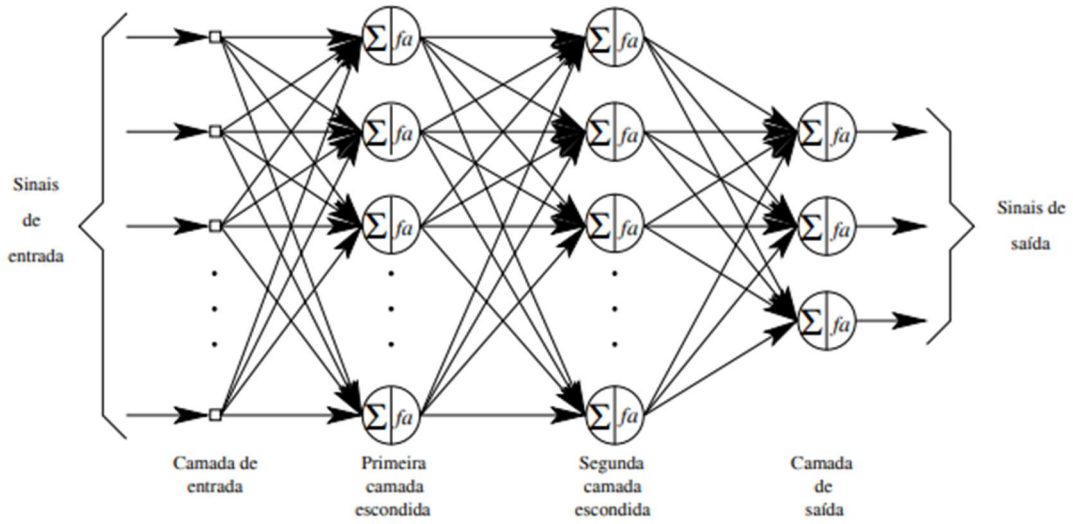
Tenta-se chegar iterativamente ao mínimo global w_{min}. A única informação que é conhecida na primeira iteração é o valor do erro EQM(w^1) = E(w^1) para o peso w_1 atual. Supõe-se que a função do erro seja derivável em todo o domínio. Isso significa que o gradiente da função de erro $\nabla E = \frac{dE(w)}{dw}$ existe.

O gradiente é um vetor. Ele aponta na direção do crescimento da função *E*. Consequentemente, o gradiente negativo aponta na direção de decrescimento da função *E*. A tentativa para chegar no mínimo da função então é a modificação do peso na iteração *1* para a iteração 2 na direção do gradiente negativo -∇E (da descida do gradiente), e assim consecutivamente. Para controlar a velocidade da modificação do peso de (w^1) para (w^2) usa-se um fator de escala, também chamada de taxa de aprendizagem

3.3.7.3 Redes Neurais de múltiplas camadas

Esse tipo de arquitetura se distingue pela presença de duas ou mais camadas ocultas (ou intermediárias), cujos nós computacionais são chamados de neurônios ocultos ou unidades ocultas. A função dos neurônios ocultos é intervir entre a entrada externa e a saída da rede de uma maneira útil. Adicionando-se uma ou mais camadas ocultas, tornamos a rede capaz de extrair estatísticas de ordem elevada. Uma representação deste tipo de rede neural pode ser observada na Figura 12.

Figura 12: Grafo arquitetural de uma rede MLP com duas camadas escondidas.



Fonte: Nied, 2007.

Assim, nesta topologia, recebe-se entradas de *n* neurônios, o neurônio *k* calcula a sua saída através da Equação 7.

$$y_k = \phi \sum_{n=1}^{i} (y_n w_{kn}) + b_k) \tag{7}$$

Em que y_k é a saída calculada pelo neurônio *i*, w_{kn} representa o peso sináptico entre o neurônio *i* e o neurônio *k* e b_k é o peso entre um valor constante e diferente de zero ao neurônio *k*, conhecido como viés ou bias. Se o neurônio estiver ligado às entradas, o termo y_n é substituído pela entrada correspondente.

Uma rede neural de múltiplas possui três tipos de camadas em sua estrutura. Elas sendo:

- **Camada de entrada:** A camada de entrada (*Input layer*) é responsável pelo recebimento de informação externa ao modelo da rede neural. Pode possuir tantos nós de entrada quantos forem necessários;

- **Camada oculta ou escondida:** Camada(s) Oculta(s) (*Hidden layers*) são responsáveis pela computação e propagação de valores de entrada em valores de saída da camada de entrada até a camada de saída. Como elemento de processamento em suas interligações, possuem pesos associados a cada conexão inter-nodal, equivalente às sinapses em um neurônio biológico, bem como um *bias* intrínseco. Além disso, possuem uma função de ativação que é responsável por filtrar a informação que será passada ao próximo neurônio artificial, tornando-o ativo ou não;

- **Camada de saída:** A camada de saída (*output layer*) é responsável por fornecer as variáveis de saída para o modelo. Dependendo do tipo do problema para o qual a rede neural for treinada, pode-se assumir valores discretos ou contínuos.

3.4 OTIMIZAÇÃO DE HIPER PARÂMETROS

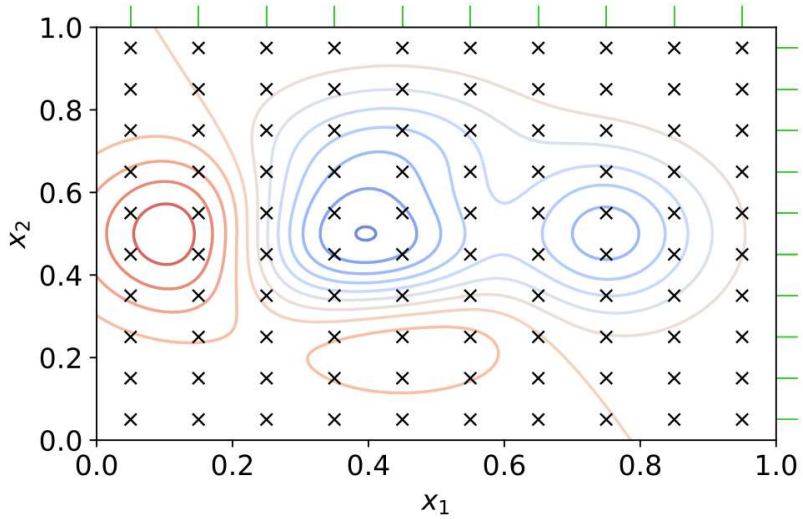
Os algoritmos de aprendizagem de máquina, independente da técnica utilizada, possuem diversos hiper parâmetros que ditam o comportamento do modelo e podem ser ajustados, num processo experimental e iterativo, para obter um desempenho melhor para uma determinada aplicação. O conjunto dos hiper parâmetros de um algoritmo é chamado de *topologia*. No contexto de aprendizagem de máquina, a busca pela melhor combinação de hiper parâmetros ou a melhor *topologia* é um processo conhecido como otimização de hiper parâmetros (PROBST, P., BISCHL. B. 2018).

A busca pelo conjunto de hiper parâmetros que têm o melhor desempenho pode ser feita de duas formas: uma forma manual ou uma forma automática. É comum que a forma manual seja apenas uma primeira abordagem experimental, para testar se a técnica escolhida, seja por redes neurais ou árvores de decisão, consegue reconhecer minimamente algum padrão dos dados. Essas situações dependem da experiência e intuição do usuário que está modelando o sistema. No entanto, quando o objetivo é encontrar a melhor combinação possível é necessário utilizar técnicas automatizadas de pesquisa e otimização, pela enorme quantidade de possibilidades que pode facilmente ultrapassar a casa da dezena de milhares.

No contexto de aprendizado supervisionado é comum ser adotada a otimização de hiper parâmetros usando a técnica *Grid Search*. Essa técnica se baseia na procura e teste exaustivos de todas as combinações possíveis de hiper parâmetros dentro do espaço definido de pesquisa. Ela é uma técnica muito demandante da capacidade computacional. Neste método, é criada uma matriz com possíveis valores de hiper parâmetros.

Cada iteração define uma combinação de hiper parâmetros e essa combinação é ajustada ao modelo. Depois disso, o desempenho do modelo é medido e a combinação que resulta no melhor desempenho do modelo é escolhida. Na Figura 13 podemos observar, para apenas dois hiper parâmetros, como funciona essa técnica.

Figura 13: Resultado da otimização Grid Search para dois hiper parâmetros, onde as linhas representam um valor de erro, sendo linhas vermelhas com um erro maior e linhas azuis com um erro menor.



Fonte: Wikipedia contributors, 2022.

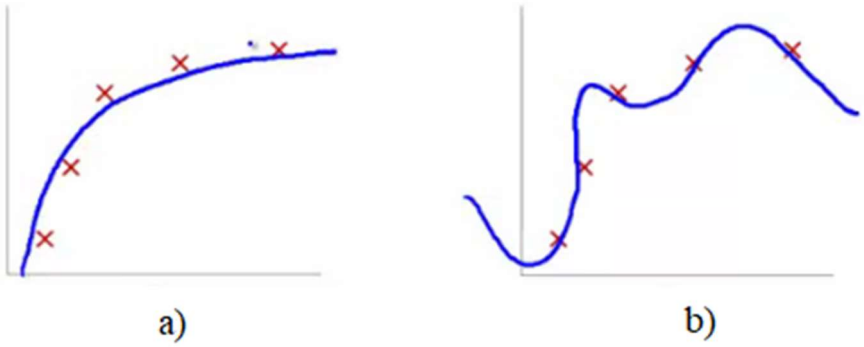
Na Figura 13, é possível observar que o espaço de possibilidades para os dois hiper parâmetros x_1 e x_2 é definido de forma uniforme. As linhas na figura representam a superfície que foi gerada, onde linhas mais vermelhas possuem um erro maior, e linhas azuis representam um erro menor.

3.5 DIVISÃO DA BASE DE DADOS

Para resolver problemas usando técnicas de aprendizado de máquina, é necessário possuir uma base de dados contendo exemplos suficientes para fazer com que o modelo identifique os padrões adequadamente. O quão vasto ou diversa essa base de dados tem que ser depende da situação e do propósito do modelo, mas é rotineiro utilizar uma base de dados com no mínimo milhares de exemplos.

Além da base de dados precisar tem um tamanho adequado, é importante que ela seja utilizada da forma correta, impedindo que o modelo acabe criando um viés ou um *overfitting* sobre os dados. O *overfitting*, ou superajuste, ocorre quando o modelo se ajusta excessivamente bem aos dados que foi utilizado em seu treinamento, no entanto, quando o modelo é aplicado em novos dados ele tem um desempenho drasticamente pior. Nesse caso, podemos considerar que o modelo “decorou” as respostas dos dados usados para treino, e não aprendeu realmente sobre seus padrões e características. Isso pode ser ilustrado na Figura 14.

Figura 14: a) Aprendizado sem *overfitting*. b) aprendizado com *overfitting*.



Fonte: O autor, 2022.

Para evitar *overfitting* ou outros vieses, é necessário que haja uma divisão da base de dados em pelo menos dois conjuntos: um conjunto de treinamento e outro conjunto de testes. Ademais, é comum que haja um terceiro conjunto de validação. A divisão nestes três conjuntos deve ser feita de modo que o conjunto de treinamento seja significativamente maior do que os demais, para garantir que o modelo tenha exemplos o suficiente para aprender.

3.5.1 Conjunto de dados de treinamento

O conjunto de dados de treinamento consiste dos dados sobre os quais o modelo será efetivamente treinado. É sobre este conjunto que o modelo deve aprender a reconhecer padrões e características do problema em questão. Podemos usar uma analogia onde o modelo é um aluno cursando uma determinada disciplina. Este conjunto de dados são os conteúdos e exercícios didáticos que se espera que o aluno estude e resolva por conta própria para fixar o conhecimento da disciplina.

3.5.2 Conjunto de dados de validação

Por sua vez, o conjunto de validação é utilizado para ser uma segunda técnica para impedir o surgimento de vieses por parte do modelo. Esse conjunto é normalmente utilizado também durante o treinamento, porém para fins de avaliação apenas. Em cada iteração, o modelo deve aprender com o conjunto de treinamento e depois ser avaliado sobre o conjunto de validação. O seu desempenho sobre o conjunto de validação serve para realizar o ajuste de seus parâmetros para a próxima iteração.

Seguindo a analogia da seção anterior, o conjunto de validação seriam trabalhos avaliativos ao longo da disciplina para verificar o conhecimento do aluno (modelo) e este poder direcionar melhor o seu estudo.

Embora o seu propósito seja simples, existem algumas maneiras diferentes de usar este conjunto de dados. A maneira mais simples é separar o conjunto previamente ao treinamento, e fornecer este conjunto estático ao processo de aprendizagem do modelo.

3.5.3 Conjunto de dados de teste

Por fim, o último conjunto de dados a ser utilizado pelo modelo é o conjunto de teste. É sobre este conjunto que é avaliado o desempenho final do modelo. Estes dados são totalmente

novos para o modelo, representado o uso do mesmo em situações práticas. Voltando a analogia do aluno cursando uma disciplina, este conjunto de dados representa a prova final da disciplina, que determina o quão bem o aluno (modelo) aprendeu o conteúdo.

3.5.4 Métricas de desempenho

Para mensurar o desempenho do modelo é necessário adotar métricas capazes de determinar o quão distante os resultados obtidos estão dos resultados esperado. Para tarefas de classificação é comum serem utilizadas as métricas de taxa de acerto, *F1-score* e haver uma análise de matriz de confusão resultante.

3.5.4.1 Matriz de confusão

A matriz de confusão é uma maneira de organizar de forma tabular os resultados obtidos e visualizar o desempenho de um modelo. A Tabela 1 apresenta uma matriz de confusão para uma tarefa de classificação binária, onde há duas classes diferentes.

Tabela 1: Matriz de confusão de exemplo.

Resultados obtidos/resultados esperados	*Classe 1*	*Classe 2*
Classe 1	10	5
Classe 2	1	25

Fonte: O autor, 2022.

Nesta tabela, as linhas representam os resultados esperados, ou seja, a classificação verdadeira dos resultados. Já as colunas representam as classificações que o modelo estimou. Idealmente, todos os valores fora da diagonal principal seriam zero de modo que o modelo não produza nenhum erro. Na prática, no entanto, isso geralmente não acontece.

É possível recriar essa tabela de maneira mais abstrata e genérica, evidenciando o tipo dos resultados. Numa matriz de confusão desta natureza podemos ter resultados *Verdadeiros Positivos*, *Verdadeiros Negativos*, *Falsos Positivos* e *Falsos Negativos*. A Tabela 2 demonstra a relação dos resultados. A Tabela 2 apresenta uma matriz de confusão genérica para situações que apresentem mais de duas classes.

Tabela 2: Matriz de confusão demonstrando a relação de positivos e negativos.

Resultados obtidos/resultados esperados	*Classe 1*	*Classe 2*	*Classe 3*
Classe 1	*VP*	*FP*	*FP*
Classe 2	*FN*	*VP*	*FP*
Classe 3	*FN*	*FN*	*VP*

Fonte: O autor, 2022.

3.5.4.2 Taxa de acerto

A taxa de acerto de um modelo representa o número de vezes que o modelo classificou corretamente uma amostra em relação ao número total de classificações realizados. Ela é calculada segundo a Equação 8.

$$Taxa\ de\ acerto = \frac{\sum VP + \sum VN}{\sum VP + \sum VN \ \ \sum FP \ \ \sum FN} \tag{8}$$

3.5.4.3 F1-Score

A métrica *F1-score* representa uma média harmônica entre outras duas diferentes métricas que podem ser abstraídas da matriz de confusão. Ela é calculada segunda a Equação 9.

$$F1score = \frac{2\rho\tau}{\rho + \tau} \tag{9}$$

Onde ρ representa a precisão e τ representa a cobertura do modelo. Elas são calculas, respectivamente, pelas Equações 10 e 11.

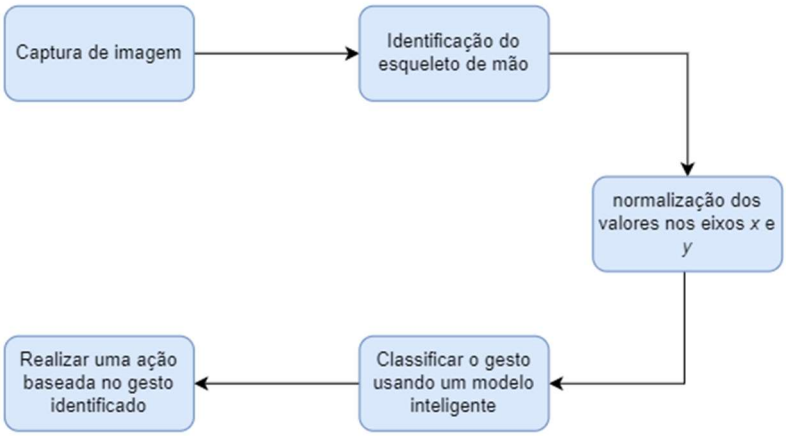
$$\rho = \frac{\sum VP}{\sum VP + \sum FP} \tag{10}$$

$$\tau = \frac{\sum VP}{\sum VP + \sum FN} \tag{11}$$

4 METODOLOGIA

Como exposto no capítulo de Introdução, o objetivo final deste trabalho é desenvolver um sistema para que um usuário possa realizar determinadas ações em um sistema operacional utilizando uma *webcam* e gestos de suas mãos. No entanto, o foco deste trabalho está em avaliar diferentes técnicas de modelagem estatística, comparando o desempenho de diferentes modelos e topologias, encontrando um modelo otimizado para a tarefa. Assim, apenas algumas ações básicas são implementadas para demonstrar o potencial deste tipo de aplicação. Na Figura 15 é apresentado um fluxograma sobre o funcionamento deste sistema.

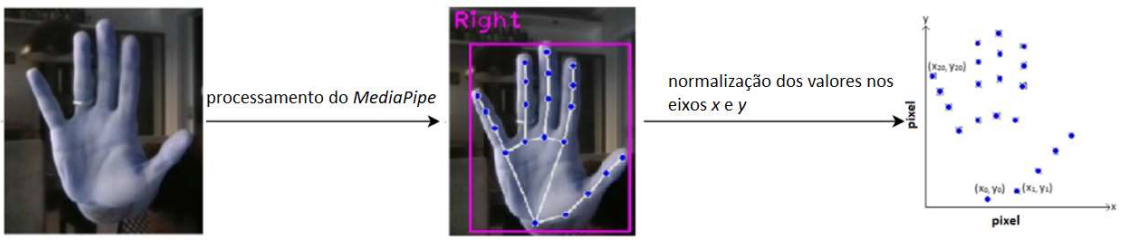
Figura 15: Fluxograma representando o funcionamento do sistema.



Fonte: Autoria própria, 2022.

Nesse fluxograma é possível observar que o funcionamento do sistema começa capturando imagens em tempo real de uma câmera digital. No caso deste trabalho, será utilizada uma webcam de notebook para fazer essa captura. Cada imagem capturada é processada utilizando a *API* chamada *MediaPipe*. Essa *API* é capaz de identificar se uma mão humana está presente na imagem e inferir a localização em pixel, nos eixos x, y e z, de 21 pontos que compõem o esqueleto dessa mão. Ela também é capaz de identificar se é a mão esquerda ou direita do sujeito. Os dados de posição de cada ponto nos eixos x e y passam a ser caracterizados como uma "amostra". Na Figura 16 é possível exemplificar esse processo de identificação.

Figura 16: Exemplo do processamento do *MediaPipe* até resultar em uma amostra.

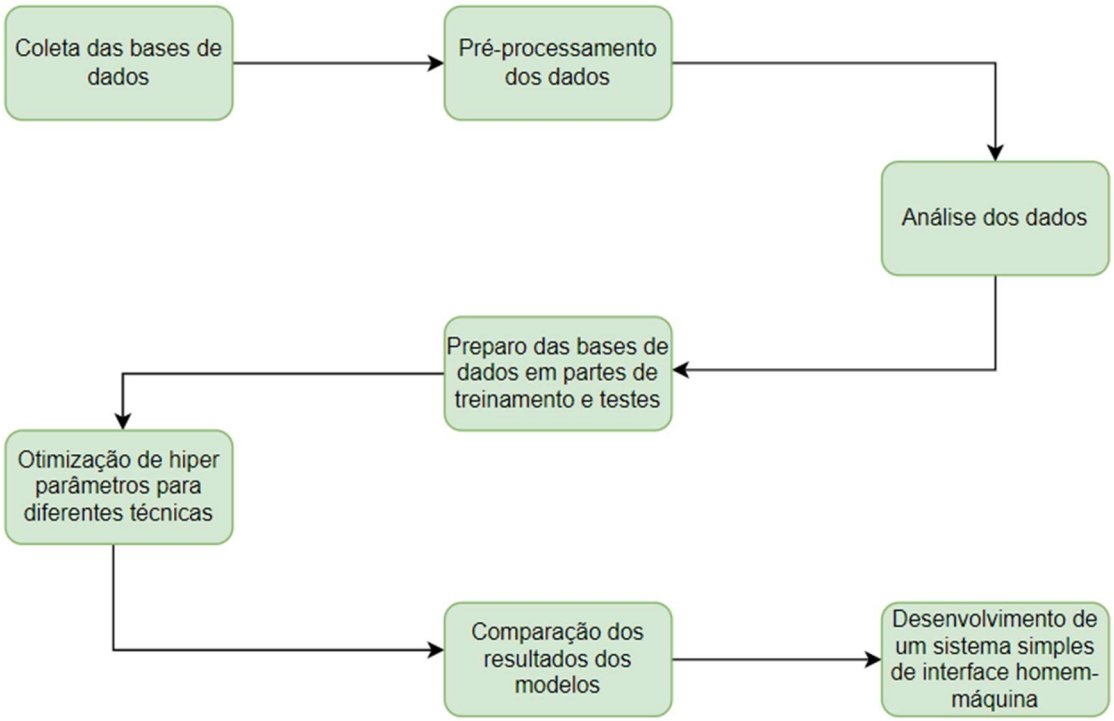


Fonte: O Autor, 2022.

Uma vez que os dados são abstraídos, é possível fornecer esses dados para um modelo classificador capaz de identificar qual gesto foi realizado. Após essa classificação, uma ação é realizada de acordo com o gesto que foi executado.

Esse sistema fica continuamente captando imagens de vídeo e realizando esse processo de funcionamento. A metodologia utilizada para desenvolver esse sistema será abordado mais detalhadamente nas seções a seguir. A Figura 17 mostra um diagrama com os principais passos tomados no desenvolvimento deste projeto de diplomação.

Figura 17. Diagrama do desenvolvimento deste projeto de diplomação.



Fonte. O autor, 2022.

4.1 EQUIPAMENTOS UTILIZADOS

4.1.1 *Hardware*

Para o desenvolvimento deste trabalho, foram utilizados os seguintes equipamentos:

- Notebook Dell Latitude 5420: Ele possui as seguintes características:
  - Processador Intel i7-1185G7;
  - Webcam capaz de capturar imagens com resolução 720p (1280 pixels x 720 pixels) a uma taxa de 30 *fps* (*frames* por segundo);
- Fita métrica retrátil de até 5 metro. Essa fita tem uma resolução de 1 mm. Ela possui uma incerteza de ± 0.5 mm e uma incerteza padrão de ± 0.287 mm.
- Lâmpada *LED* de 4,9 Watts de cor branca 6.500K em um abajur articulado;
- Suporte de *notebook*.

4.1.2 *Software*

Foi utilizado a linguagem de programação *Python* versão 3.8.8 durante todo o projeto. Para a parte de aquisição de dados foram utilizadas as bibliotecas O*penCV* para utilizar a *webcam* de maneira programática, *MediaPipe* para utilizar o modelo que abstrai o esqueleto da mão, *numpy* para realizar operações sobre os dados capturados e *pickle* para salvar os dados capturados. Para a etapa de análise e pré-processamento de dados foram utilizadas as bibliotecas *numpy, pickle, pandas* para trabalhar com os dados de maneira tabular e *plotly* para criar visualizações dos dados.

Para a etapa de criação de modelos inteligentes foram utilizadas as bibliotecas *sklearn*, *tensorflow* e *xgboost*. Para o funcionamento do sistema foram utilizadas as bibliotecas *OpenCV, MediaPipe, numpy* e *PyAutoGUI* para criar comandos para interagir com o sistema operacional do computador.

A Tabela 3 apresenta as versões das ferramentas utilizadas.

Tabela 3: Versão das ferramentas utilizadas.

Ferramenta	*Versão*
OpenCV	4.5.5
MediaPipe	0.8.9
Numpy	1.21.5
Pickle	4.0
Pandas	1.3.5
Plotly	5.5.0
Sklearn	1.0.2
Tensorflow	2.7.0
Xgboost	1.6.1
PyAutoGUI	0.9.53

Fonte: O autor, 2022.

4.2 AQUISIÇÃO DE DADOS

Para desenvolver um modelo de aprendizagem de máquina que seja capaz de ter um bom desempenho é necessário que haja uma grande quantidade de dados e, além disso, que estes dados sejam diversos o bastante para evitar que seja criado um viés por parte deste modelo. Neste trabalho, os gestos de interesse foram definidos com base na facilidade de execução. Como não é de conhecimento do autor a existência de base de dados com estes mesmos gestos, foi necessário realizar as coletas experimentais destes dados.

Nesta seção será explorado, primeiro, quais são os gestos de interesse para este projeto. Em seguida, são abordados os projetos de experimentos das duas coletas que foram realizadas. Cada coleta gerou uma base de dados diferente. Ambas as bases de dados estão disponíveis no repositório deste projeto no site GitHub, dentro da aba “data”.

4.2.1 Gestos de Interesse

Para escolher os gestos que compõem as bases de dados, foram considerados gestos que são naturais e de fácil execução. Para o propósito deste trabalho, foram escolhidos 10 gestos, onde 6 são gestos de interesse que serão utilizados como comandos pelo sistema proposto, e 4 são gestos de não interesse, que são semelhantes a gestos de interesses, porém espelhados. Na Figura 18 são demonstrados esses gestos.

Figura 18: Gestos de interesse do sistema e suas numerações.

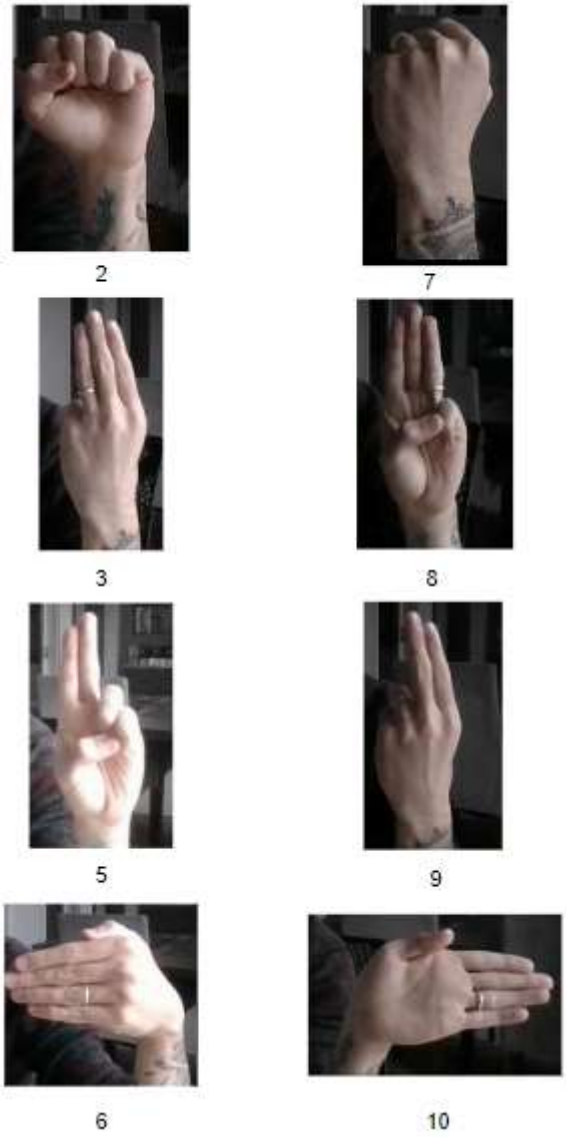
Fonte: O autor, 2022.

Uma preocupação em aplicações de reconhecimento de gestos de mãos é saber lidar com gestos que não são de interesse, ou então, gestos que representam movimentos aleatórios. Muitas vezes, no entanto, essa questão é simplesmente desconsiderada, ou, pelo menos, não é abordado em artigos científicos, como em Liu *et al*, Nguyen *et al*, Qi *et al* e Mujahid *et al*. Em Sung *et al* (2021), por exemplo, foram definidos 6 gestos distintos de interesse que o seu modelo

deveria reconhecer, e outros 15 gestos foram definidos como de não interesses e classificados como uma única classe. Benitez-Garcia *et al* (2021) utilizou uma abordagem parecida para tratar de gestos sem interesse.

Neste projeto, foi determinado que seria mais adequado definir diferentes classes de gestos sem interesse. Essas classes são variações dos gestos de interesse. Essa abordagem foi adotada para reduzir a complexidade destes casos sem interesse, uma vez que, caso fosse utilizado apenas uma classe, ele precisaria ter mais dados que as demais classes de interesse, para que ela consiga generalizar todas as diversas variações de não interesse. Isso iria causar um desequilibro na base de dados, uma vez que as outras classes não teriam tantos dados, o que pode causar um viés na modelagem estatística. A Figura 19 apresenta as classes de interesse e suas variações que não são de interesse.

Figura 19: Gestos de interesse, ao lado esquerdo, e suas variações que não são de interesse, ao lado direito, e suas numerações.

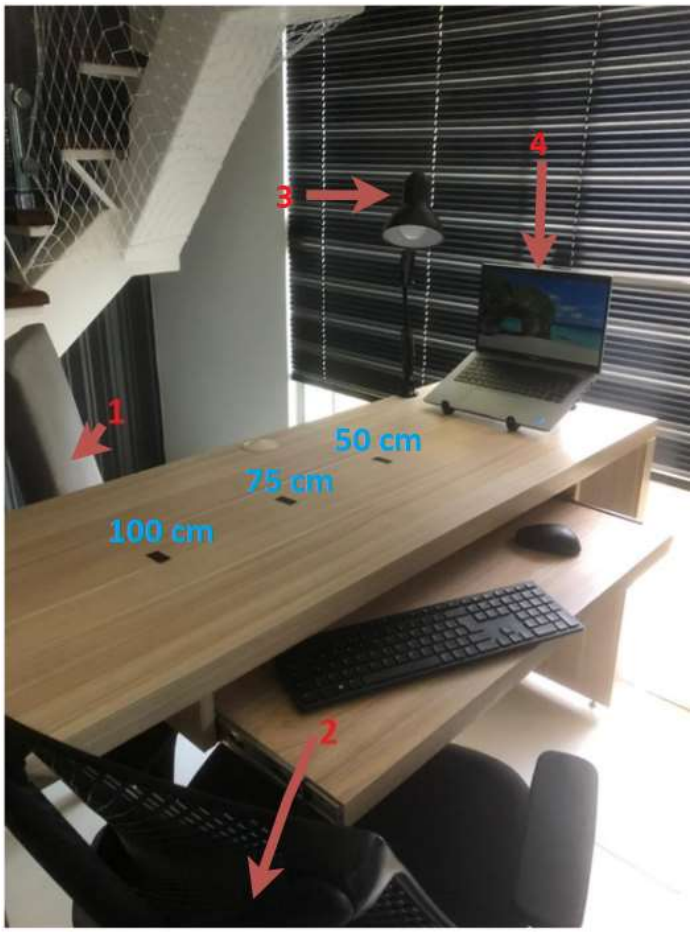


Fonte: O autor, 2022.

4.2.2 Projetos de Experimentos

Neste trabalho são coletadas duas bases de dados diferentes, cada uma com um procedimento de coleta distinto. No entanto, ambas foram coletadas no mesmo ambiente de experimentos. Este ambiente pode ser observado na Figura 20.

Figura 20: Ambiente de experimentos. 1 – Cadeira do voluntário, 2 – Cadeira do condutor do experimento, 3 – Iluminação controlada, 4 – Notebook utilizado para capturada de imagens.



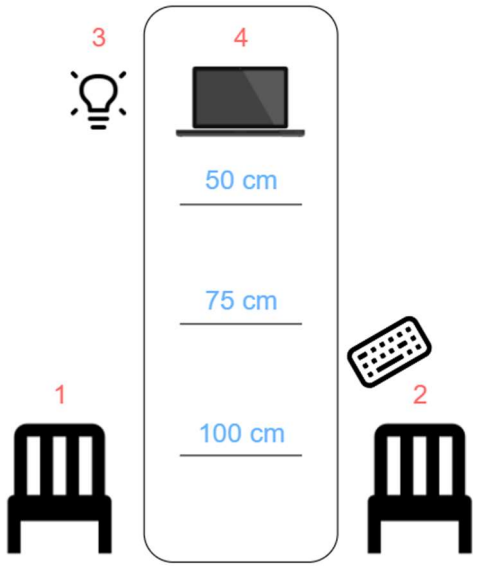
Fonte: O autor, 2022.

Este ambiente foi preparado no apartamento do autor. O sujeito de teste, que produz os gestos, fica sentado na cadeira com a numeração 1. Enquanto isso, o condutor dos experimentos fica posicionado na cadeira com a numeração 2, onde tem o controle do início de cada rodada. O ambiente conta com uma iluminação natural, no entanto, com o intuito de tentar reduzir a variabilidade do fator de iluminação, foi introduzido a presença de uma lâmpada *LED* de 4,9 Watts de cor branca 6.500K. Além disso, todos os testes foram realizados com as cortinas venezianas fechadas, por volta das 17h da tarde, entre os dias 16 e 27 de julho, para a coleta da base de dados aleatorizada, e no dia 13 de agosto para a base de dados não aleatorizada.

Além disso, no corpo da mesa há três marcações com uma fita preta, representado três diferentes distâncias da webcam: 50,00 cm, 75,00 cm e 100,00 cm. A distância foi medida horizontalmente em relação a *webcam* do *notebook* utilizando a fita métrica citada na seção anterior. Podemos estimar que haja uma incerteza seja metade da resolução da fita métrica (1 mm) o que corresponde a uma incerteza-padrão de ± 0.287 mm nas medições de distância,

assumindo conversão de distribuição uniforme para normal. Apesar de haver outras fontes de incerteza, como a incerteza no posicionamento das fitas adesivas, apenas a incerteza na medida das distâncias foi considerada na construção do ambiente de experimentos. Essa incerteza, no entanto, foi desconsiderada no restante deste trabalho. A Figura 21 demonstra a visão superior do ambiente de teste de maneira ilustrativa, seguindo a mesma numeração e equipamentos da Figura 20.

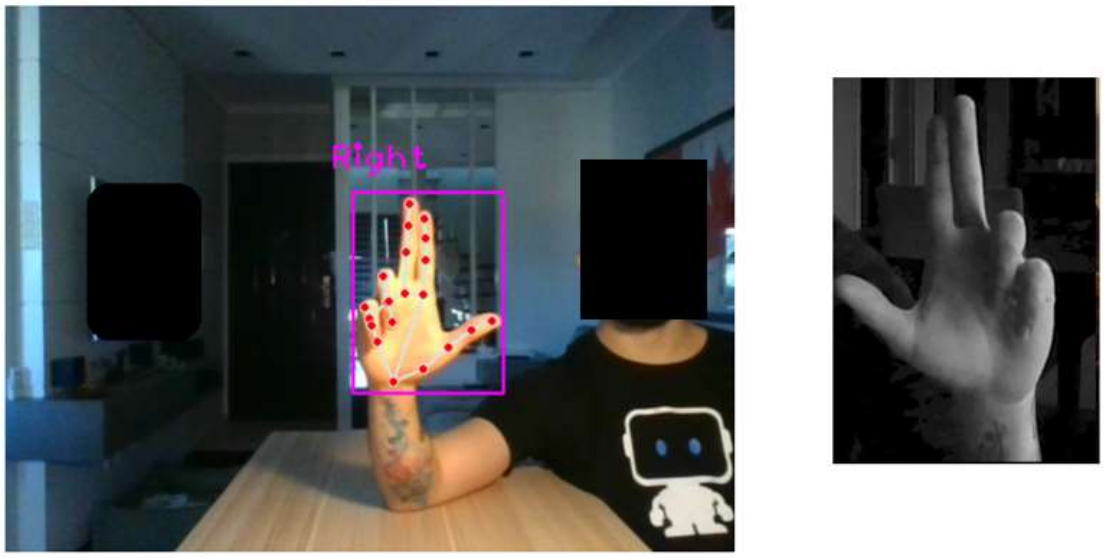
Figura 21. Ilustração da visão superior do ambiente de experimentos. 1 – Cadeira do voluntário, 2 – Cadeira do condutor do experimento, 3 – Iluminação controlada, 4 – Notebook utilizado para capturada de imagens.



Fonte. O autor, 2022.

Nesse ambiente, o voluntário que executa os gestos pode visualizar na tela do *notebook* tanto uma janela que mostra o vídeo da *webcam* sendo capturado em tempo real quanto uma segunda janela que serve para exigir qual gesto ele deve reproduzir. O gesto é exibido de maneira espelhada pois experimentalmente foi notado que é mais confortável para os voluntários. Na Figura 22 é possível observar como foi é esse ponto de vista durante a coleta de dados, reproduzido, neste caso, pelo autor.

Figura 22: Reprodução do ponto de vista dos voluntários pelo autor. a) Tela de captura de dados, b) Tela que seleciona o gesto a ser executado.



a b

Fonte: O autor, 2022.

Para cada *frame* capturado pela *webcam* passa por um processamento por parte do modelo de identificação de esqueleto de mão da *API MediaPipe*, que fornece a localização, nos eixos *x, y* e *z,* da estrutura do esqueleto da mão, com 21 pontos, assim como a informação se a mão observada é a mão direita ou esquerda. Quando esse *frame* é salvo como uma amostra, o eixo *z* é ignorado resultando em uma matriz de formato 21x2, onde cada linha representa um ponto do esqueleto, a primeira coluna representa a posição no eixo *x* e a segunda no eixo *y*.

O procedimento que salva um *frame* como amostra é distinto para cada coleta de base de dados e será abordado novamente nas seções posteriores. Na Tabela 4 é apresentado essa matriz de posições para os pontos que são apresentados no exemplo da Figura 22.

Tabela 4: Matriz de posição dos pontos do gesto exemplo da Figura 20.

Pontos \ Eixos	*X [pixel]*	*Y [pixel]*
Ponto 1	275	385
Ponto 2	295	383
Ponto 3	311	383
...	...	...
Ponto 21	262	405

Fonte: O autor, 2022.

As posições nos eixos *x* e *y* são dados em *pixels*, onde os valores do eixo *x* são valores inteiros entre 0 e 1280, e os valores de *y* são valores inteiros entre 0 e 720. Esses intervalos de valores dependem da resolução de tela do dispositivo que captura as imagens, que neste caso foi abordado na seção anterior explorando equipamentos de *hardware* utilizados.

O sistema que este trabalho desenvolve propõe que seja possível identificar gestos ao longo de uma faixa de operação de 50 cm até 100 cm de distância *webcam*. Para demonstrar essa continuidade na capacidade do sistema, três distâncias são definidas para que sejam feitas baterias de coletas de dados.

Os experimentos a serem realizados para construir as bases de dados estão divididos em dois planejamentos experimentais distintos, cada um com seu objetivo.

4.2.2.1 Projeto de experimentos - base de dados aleatorizada

A base de dados aleatorizada tem como objetivo fornecer dados com uma maior variação entre as amostras e garantir que estas estejam independentes entre si. Para isso, foi definido um procedimento de coleta de dados com o auxílio de 5 voluntários para executar os gestos.

Cada voluntário participa de três baterias de coletas, onde, em cada bateria, ele deve realizar 30 repetições de cada um dos dez gestos de maneira aleatória. Para isso, ele conta com o auxílio de um programa desenvolvido pelo autor que seleciona, de maneira aleatória, um dos gestos.

Vale notar que esse programa permite que o mesmo gesto seja selecionado em sequência, o que acontece puramente pelo acaso, e que, quando são realizadas todas as 30 repetições de um gesto, este não pode mais ser selecionado, o que altera a probabilidade dos gestos restantes.

Quando um gesto é selecionado, o condutor do experimento espera que o voluntário o reproduza e então salva manualmente os dados relativos à repetição utilizando um atalho programado no teclado, que no caso foi utilizado a letra “x”. Uma vez que os dados do gesto são salvos, outro gesto é selecionado, e o processo é repetido até que todos os dados dos gestos sejam coletados.

Durante as execuções dos gestos por parte dos voluntários, o condutor do experimento inspeciona e aponta quando a distância da mão do voluntário se desalinha da marca de distância da bateira. Essa inspeção é feita de maneira visual. Além disso, não há nenhuma restrição quanto a posição na tela que o voluntário deve executar os gestos, contanto que a gesto seja capturado por completo.

A primeira bateria de coleta ocorre na marca de 50 cm de distância da *webcam*, seguido da coleta a 75 cm e, por fim, a 100 cm de distância. Cada bateria tem em média 12 minutos de duração e há um intervalo de aproximadamente 5 minutos entre cada bateria. Na Tabela 5 é possível observar um exemplo fictício sobre a realização das coletas com um voluntário, onde os números representam a ordem de reprodução dos gestos, sendo “1” o primeiro gesto e “300” o último gesto. A ordem de execução real dos gestos não foi registrada.

Tabela 5: Organização de um exemplo de coleta de dados para um único voluntário.

Distância\Gesto	*Gesto 1*	*Gesto2*	*Gesto 3*	*...*	*Gesto 10*
50 cm	3, 5, ...	2, 14, ...	1, 10, ...	...	12, 24, ...
75 cm	7, 25, ...	6, 13, ...	8, 10, ...	...	2, 5, ...
100 cm	10, 17, ...	4, 15, ...	2, 5, ...	...	8, 21, ...

Fonte: O autor, 2022.

Juntamente com uma etapa de introdução e instrução sobre os experimentos e um momento de, aproximadamente, 5 minutos para o voluntário fazer testes e se familiarizar com o processo, o tempo total de coleta para cada voluntário foi de, em média, 1 hora e 15 minutos.

Durante a coleta desta base de dados o autor participou apenas conduzindo os experimentos. Todos os candidatos são destros, foram instruídos sobre o funcionamento dos experimentos e tiveram aproximadamente 5 minutos fazendo testes, cujos dados não foram salvos em nenhum momento. Além disso, devido a duração dos experimentos, cada voluntário participou dos experimentos em dias diferentes.

Assim, essa base de dados consiste em 450 amostras para cada gesto, sendo 30 amostras para cada voluntário. Dessa forma, resultando em um total em 4.500 amostras. É importante notar que esta base de dados opta por ter uma maior qualidade em seus dados, do ponto de vista de generalização, em sacrifício de uma maior quantidade, pois aumentar a quantidade de dados, seguindo os mesmos procedimentos de coleta, implicaria em aumentar a duração dos experimentos com cada voluntário ou em aumentar a quantidade de voluntários. Em ambas as opções, haveria um aumento significativo no tempo empregado e na complexidade de organização para obter essa base de dados.

4.2.2.2 Experimentos da base de dados não aleatorizada

Em contrapartida ao extenso e demorado procedimento de coleta da base de dados aleatorizada, a base de dados não aleatorizada tem como objetivo fornecer um procedimento que seja mais simples e rápido de ser executado, sacrificando a qualidade de seus dados, em questão de variabilidade e generalização, e focando em obter uma quantidade de amostras mais elevada.

Esse procedimento conta com a coleta de dados utilizando apenas um voluntário e deixa de lado o fator aleatório na execução dos gestos. Neste procedimento são realizadas, novamente, três baterias de coletas, uma para cada marcação de distância. Em cada bateria o voluntário executa, em sequência, os 10 gestos definidos. Para cada uma dessas execuções, o voluntário reproduz o gesto e acena estar pronto para o início da coleta. Nessa coleta, cada *frame* capturado pela webcam em tempo real, é salvo como uma amostra, até serem capturados 1.500 amostrar para cada gesto, em uma bateria. Durante a captura dos *frames*, o voluntário realiza leves movimentos para adicionar variações sem desfazer o gesto que está sendo reproduzido.

Dessa forma, é possível capturar amostras muito mais rapidamente do que o procedimento da base de dados aleatorizada pois a captura é feita em tempo real, sem precisar ser feita manualmente pelo condutor. Quando todas as 1.500 amostras de um gesto são

capturadas, um próximo gesto é selecionado, o voluntário se prepara e aponta estar pronto para a coleta.

A sequência de execução dos gestos é definida antes do começo das amostras. Além disso, para a coleta dessa base de dados, o autor foi o voluntário que executava os gestos, assim, não foi preciso realizar uma etapa de introdução dos experimentos. Novamente, a primeira bateria ocorre a uma distância de 50 cm da *webcam*, seguido de uma distância de 75 cm e, então, de 100 cm. Na Tabela 6 é possível observar como foi feita a real ordem de execução dos gestos. Lembrando que a numeração dos gestos é definida na seção 4.2.1.

Tabela 6: Organização da real execução dos gestos para a coleta de dados.

Distância\Gesto	*Gesto 1*	*Gesto2*	*Gesto 3*	*...*	*Gesto 10*
50 cm	1, 2, ...	1501, 1502, ...	3001, 3002, ...	...	13.501, 13.502, ...
75 cm	1, 2, ...	1501, 1502, ...	3001, 3002, ...	...	13.501, 13.502, ...
100 cm	1, 2, ...	1501, 1502, ...	3001, 3002, ...	...	13.501, 13.502, ...

Fonte: O autor, 2022.

Cada bateria teve uma duração média de 12 minutos, levando em consideração que para cada gesto houve um tempo de posicionamento para as amostras começarem a serem capturadas, para realizar a captura de 15.000 amostras utilizando a webcam do notebook que consegue capturar até 30 *frames* por segundo (30 *fps*). Assim, a base de dados completa possui 45.000 amostras e foi construída em, em torno de, 40 minutos.

4.3 Análise, preparo e pré-processamento das bases de dados coletadas

Uma vez que as bases de dados tenham sido coletadas e disponibilizadas para uso, neste projeto, será realizada uma etapa de pré-processamento seguida de uma análise onde as características das amostras são exploradas. Em seguida é realizado uma etapa de preparo, onde as bases de dados são dividias em partes, onde essas partes são destinadas ao treinamento dos modelos, validação e teste. Por fim, os resultados dessas etapas são apresentados no capítulo posterior sobre análise dos resultados do trabalho.

4.3.1 Pré-processamento de dados

Cada amostra disponível, tanto para a base aleatorizada quanto para a base não aleatorizada, é uma matriz de formato 21x2, onde cada linha é um dos 21 pontos do esqueleto da mão, sendo a primeira coluna a posição no eixo *x* e a segunda coluna a posição no eixo *y*. Os valores no eixo *x* são número inteiros entre 0 e 1200 e no eixo *y* são valores inteiros entre 0 e 720.

No entanto, como esses dados serão usados para treinar modelos que utilizam técnicas de aprendizado de máquina, é vantajoso que esses intervalos sejam normalizados, como demonstra Singh e Singh (2020). Além desse tipo de processamento poder afetar positivamente

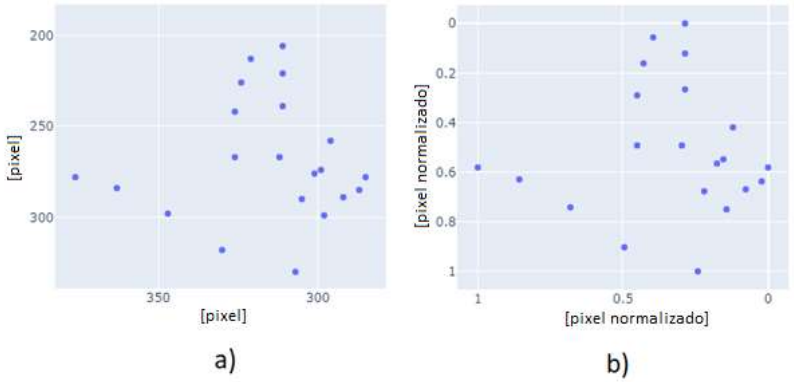
o desempenho dos modelos, ele também é muito importante para o caso deste trabalho pois assim desvincula os dados da amostra com o local na tela que o gesto foi feito.

É comum que os intervalos sejam normalizados para um intervalo de –1 a +1 ou 0 a +1. Nesse caso optou-se por normalizar os dois intervalos entre 0 e +1. Para isso foi utilizada a Equação 12.

$$x' = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{12}$$

Onde x é a série de valores originais que está sendo normalizada, $\max(x)$ é o valor máximo da série original, $\min(x)$ é o valor mínimo da série original, x_i é o valor da série original a ser normalizado e x' é o valor normalizado. Assim, a série normalizada resultante consiste em valores de ponto flutuante. Na Figura 23 é possível observar uma comparação entre uma amostra com seus valores originais e seus valores normalizados.

Figura 23: a) Amostra com seus valores originais. b) Amostra com seus valores normalizados



Fonte: O autor, 2022.

4.3.2 Análise dos dados

O objetivo dessa análise é avaliar a consistência e relação dos dados dos gestos para as três diferentes distâncias. Dessa forma, é possível avaliar se a premissa do trabalho de que é possível identificar gestos em todo o intervalo de distância de 50 cm até 100 cm pode ser mantida, ou se é necessário implementar uma etapa de compensação de distância.

Para isso, cada gesto é analisado de maneira independente. Para cada gesto, são construídas três séries contendo todas as amostras para cada distância de coleta. Com os dados das amostras de cada série, é possível calcular as posições médias para cada um dos 21 pontos que compõem o esqueleto da mão. Dessa forma, é possível abstrair uma matriz de posição, de tamanho 21x2, que representa o “gesto médio” para cada distância e comparar graficamente os resultados. Com as Equações 13 e 14 é possível calcular essa matriz de posição.

$$P_{ix} = \frac{1}{l}\sum_{n=1}^{l} S_{nix} \tag{13}$$

$$P_{iy} = \frac{1}{l}\sum_{n=1}^{l} S_{niy} \quad (14)$$

Onde P_{ix} é o valor da posição do ponto *i* no eixo x, P_{iy} é o valor da posição do ponto *i* no eixo y, *I* é o tamanho da série de amostras *S*, S_{nix} é o valor da posição do ponto *i* do elemento *n* da série de amostras *S* no eixo x e S_{niy} é o valor da posição do ponto *i* do elemento *n* da série de amostras *S* no eixo y. Resultando na matriz de posição representada pela Tabela 7.

Tabela 7: Matriz de posição representando o gesto médio de uma série de amostras.

Pontos \ Eixos	*X*	*Y*
Ponto 1	P_{1x}	P_{1y}
Ponto 2	P_{2x}	P_{2y}
Ponto 3	P_{3x}	P_{3y}
...	...	...
Ponto 21	P_{21x}	P_{21y}

Fonte: O autor, 2022.

Além disso, é possível quantificar a diferença geométrica entre dois gestos médios da mesma classe de gesto utilizando a Equação 15.

$$Dif_{G1,G2} = \frac{1}{l}\sum_{n=1}^{l}\sqrt{(G1_{nx} - G2_{nx})^2 - \left(G1_{ny} - G2_{ny}\right)^2} \quad (15)$$

Onde n é a identificação do ponto que compõe o gesto, $G1_{nx}$ é o valor no eixo *x* do ponto n do gesto $G1$, $G1_{ny}$ é o valor no eixo *y* do ponto n do gesto $G1$, $G2_{nx}$ é o valor no eixo *x* do ponto n do gesto $G2$ e $G2_{ny}$ é o valor no eixo *y* do ponto n do gesto $G2$.

Utilizando essa métrica, primeiro é avaliada a diferença média entre todos os pares de gestos médios da base de dados, bem como a diferença máxima e a diferença mínima. Então, são analisadas isoladamente as diferenças obtidas entre os gestos médios de distância 50 cm e 75 cm, seguido dos gestos médios de 50 cm e 100 cm, e, por fim, os gestos médios de 75 cm e 100 cm. Para cada par, é possível obter a diferença média, e os seus valores máximos e mínimos.

Em seguida, é possível realizar uma análise sobre cada um dos vinte e um pontos que compõem o esqueleto da mão, desconsiderando o fator distância. Nessa análise, são calculados os valores de média e desvio padrão nos eixos x e y, separadamente, de cada ponto. Isso é feito para cada gesto. Dessa forma, é possível entender melhor as regiões em que um ponto específico do esqueleto se distribuí ao longo da base de dados. A média e o desvio padrão são calculados segundo as Equações 16 e 78, respectivamente.

$$M_{ki} = \frac{1}{l}\sum_{n=1}^{l} P_{kni} \quad (16)$$

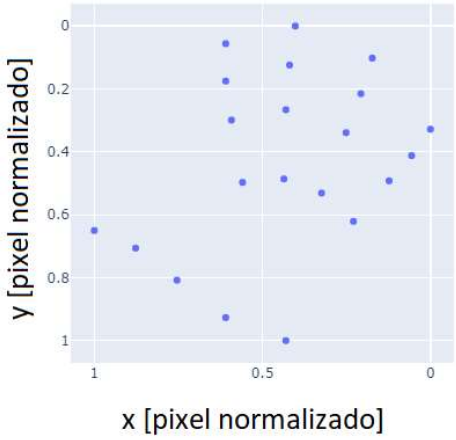
$$DP_{ki} = \sqrt{\frac{1}{l}\sum_{n=1}^{l}(P_{kni} - M_{ki})^2} \quad (17)$$

Onde, P_{kni} é o valor no eixo *i* do ponto k da amostra de número n, l é o número total de amostras de uma classe de gestos da base de dados, M_{ki} é o valor médio no eixo *i* do ponto *k* e DP_{ki} é o desvio padrão no eixo *i* do ponto *k*.

Essa análise é feita individualmente para cada gesto. Dessa forma, é possível compreender melhor a variabilidade obtida na execução de cada gesto.

Por fim, é realizada uma última análise, ainda sobre cada gesto, tendo como referência um gesto de referência. Esse gesto referência é demonstrado na Figura 24.

Figura 24: Gesto de referência.



Fonte: O autor, 2022.

Cada amostra, de cada gesto, é comparada com esse gesto de referência e é calculado a média das distâncias dos pontos da amostra para os pontos do gesto. Todos os dez gestos são comparados sobre o mesmo gesto de referência definido acima. Essa média é calculada pela Equação 18.

$$m_k = \frac{1}{21}\sum_{n=1}^{21}\sqrt{(Ref_{nx} - P_{knx})^2 + (Ref_{ny} - P_{kny})^2} \quad (18)$$

Onde m_k é a média das distâncias entre os pontos da amostra k, Ref_{nx} é a posição no eixo *x* do ponto *n* do gesto de referência, P_{knx} é a posição no eixo *x* do ponto *n* da amostra *k*, Ref_{ny} é a posição no eixo *y* do ponto *n* do gesto de referência e P_{kny} é a posição no eixo *y* do ponto *n* da amostra *k*.

4.3.3 Preparo das bases de dados

Uma vez que os dados das duas bases tenham sido normalizados, então, é necessário que seja feito um preparo no sentido de dividir os dados de cada base em partes que terão diferentes finalidades na etapa de treinamento de modelos inteligentes. Dessa forma, a base de

dados aleatorizada é dividida em três partes, uma parte de treinamento que contêm 70% dos dados, uma de validação contendo 15% e uma parte de teste que tem 15% dos dados da base. A base de dados não aleatorizada, por sua vez, é divido apenas em parte de treinamento, contendo 80% dos dados, e em parte de validação com 20% dos dados. O motivo de dividir a base de dados não aleatorizada apenas em partes de treinamento e validação é porque será utilizado a parte de teste da base de dados aleatorizada.

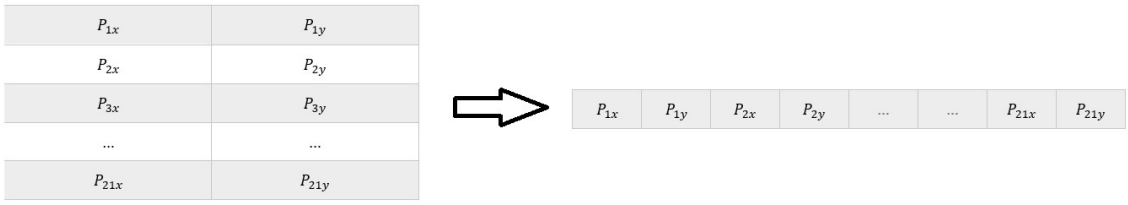
Antes que seja feito essa separação, os dados de cada base de dados são embaralhados, para que a distribuição dos gestos em cada parte de treinamento, validação e teste seja aproximadamente igual. Após que as partes sejam preparadas, é importante um último passo que visa equilibrar os dados de treinamento das duas bases, ou seja, fazendo com que os dados de treinamento tenham a mesma quantidade de amostras para cada gesto. Isso evita que haja o desenvolvimento de um viés de escolha nas etapas de treinamento de modelos inteligente. Para manter o equilíbrio dos dados, todas as classes passam a ter a mesma quantidade de amostras que a classe com a menor quantidade. O excesso de amostras é totalmente descartado. O descarte destes dados em excesso não acarreta um problema pois apenas é descartado uma quantidade pequena em relação ao restante dos dados, o que será demonstrado no capítulo de Resultados.

4.4 TREINAMENTO DE UM MODELO INTELIGENTE PARA CLASSIFICAÇÃO DE GESTOS

Com as bases de dados já constituídas, pré-processadas e preparadas, a próxima parte deste trabalho é realizar uma modelagem estatística para definir um modelo que consiga obter o melhor desempenho na classificação dos gestos. Para isso, é primeiramente realizada uma bateria de otimização de hiper parâmetros para cada técnica proposta. Nessa etapa, cada combinação de hiper parâmetro, ou seja, topologia, é treinada sobre os dados de treinamento e avaliado sobre os dados de validação, ambos da base de dados aleatorizada.

Para todas as técnicas, exceto para a baseada em rede neural e *MLP*, é necessário que os dados das amostras tenham seu formato alterado. O formato matricial 21x2 original das amostras devem ser transformados em uma matriz de formato 1x42, ou seja, virando um vetor. A Figura 25 demonstra uma amostra sendo transformada.

Figura 25: Transformação do formato de uma amostra.



Fonte: O autor, 2022.

Para mensurar o desempenho, serão adotadas duas métricas que são abordadas no capítulo de fundamentação teórica:

- **Taxa de acerto de classificação:** representando o desempenho geral do modelo, sendo definido pela razão da quantidade de predições corretas pela quantidade total de predições;
- ***F1-score*:** representa a média harmônica entre a taxa de acerto e precisão das inferências, definida pela equação 2 apresentada na fundamentação teórica;

Podemos apresentar as técnicas propostas, bem como os hiper parâmetros que serão explorados abaixo:

1. **Regressão logística**: Para esta técnica serão explorados os hiper parâmetros a respeito do algoritmo de otimização utilizado, podendo ser “sag”, “saga”, “lbfgs” e “newton-cg”, e se haverá uso da técnica de regularização conhecido como “L2”, podendo ser “sim” ou “não”;

2. **Classificador Bayes ingênuo**: Esta técnica não proporciona nenhuma mudança em seus hiper parâmetro ou algoritmos internos;

3. **Máquina de suporte de vetor**: Para essa técnica foi apenas possível definir quatro diferentes topologias, mudando apenas a função *kernel* utilizada. Foram exploradas as funções “linear”, “poly”, “rbf” e “sigmoid”;

4. **Árvore de decisão**: Nesse caso são explorados dois diferentes hiper parâmetros, o critério para medir a qualidade da repartição de um nó e a estratégia de repartição. O critério pode ser “gini” ou “entropy”. A estratégia pode ser “best” ou “random”;

5. **Floresta aleatória**: Nesta técnica foi possível explorar três hiper parâmetros: o número de árvores de decisão que compõem a floresta aleatória, o critério para medir a qualidade da repartição de um nó e se todos os dados são utilizados em todas para treinar cada árvore. Foram testados valores de 1, 2, 4, 8, 16, 32, 64 e 128 para a quantidade de árvores. Os critérios podem ser “gini” ou “entropy”;

6. **XGBoost**: Neste caso, três hiper parâmetros foram explorados. A quantidade de árvores de decisão que compõem o modelo, podendo ser um valor de 32, 64, 128, 256, 512 ou 1024. A taxa de aprendizagem, podendo ser um valor de 0.2, 0.3, 0.4, 0.5, 0.6, 0.7. A estratégia de crescimento das árvores, podendo ser “depthwise” ou “lossguide”;

7. **Rede neural simples**: Para o caso da rede neural simples foram abordos três hiper parâmetros. O algoritmo otimizador, que pode ser “adam” ou “sgd”. A quantidade de neurônios na sua camada oculta, podendo ser um valor de 32, 64, 128, 512, 1024, 2048 ou 4096. A função de ativação, que pode ser “relu”, “sigmoid” ou “tanh”;

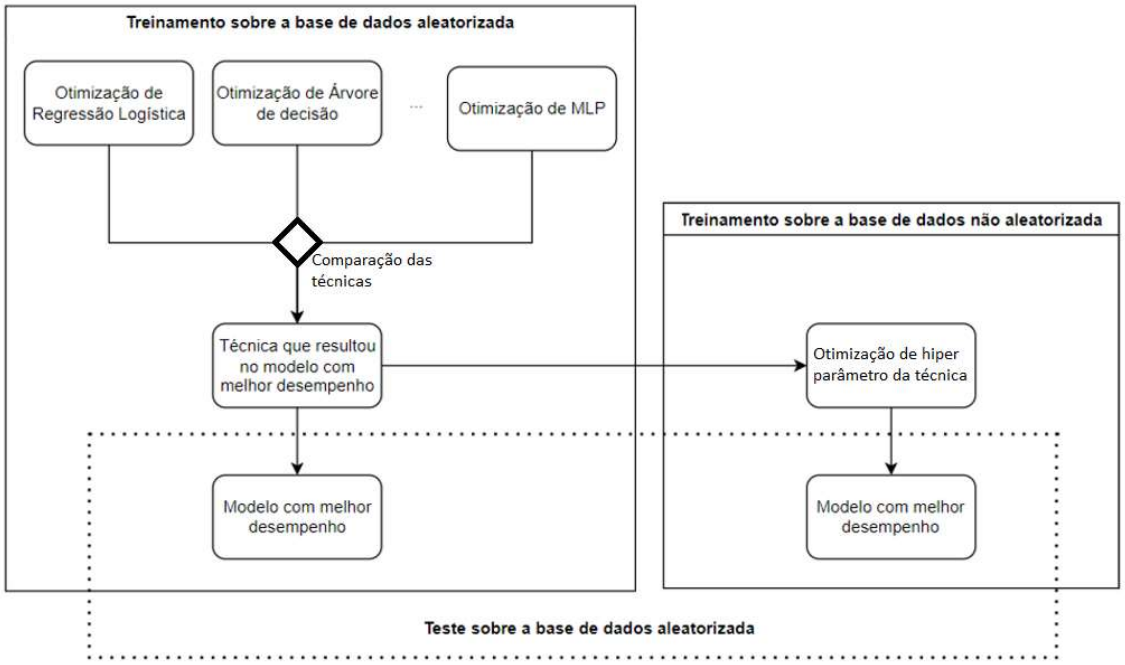
8. **Rede neural de múltiplas camadas**: Para esta técnica houve uma maior quantidade de possibilidades. O primeiro hiper parâmetro abordado é a quantidade de camadas ocultas, podendo ser de 2 camadas até 5. O segundo hiper parâmetro é a quantidade

de neurônios na primeira camada oculta, que pode ser de 32, 64 ou 128. A quantidade na segunda e terceira camada podem ser de 32, 64, 128, 256, 512, 1024, 2048 e 4096. A quarta e quinta camada podem ser de 32, 64 e 128. Além disso, foi utilizado o algoritmo otimizador “adam” e a função de ativação “sigmoid” em todas as camadas ocultas.

Depois dessa etapa de otimização, a topologia que obteve o melhor resultado para cada uma das técnicas é, então, avaliada sobre os dados de teste da base. Após isso, a técnica que resultou no melhor modelo é utilizada para criar um segundo modelo treinando sobre a base de dados não aleatorizada.

Nessa etapa novamente é realizado um processo de otimização de hiper parâmetros com o mesmo universo definido para a etapa anterior. Neste processo, as topologias são treinadas sobre os dados de treinamento e avaliadas sobre os dados de validação da base de dados não aleatorizada. Por fim, as duas topologias com melhor desempenho são comparadas sobre os dados de teste da base aleatorizada. A Figura 26 apresenta um diagrama desse processo.

Figura 26: Diagrama dos processos de otimização e ranqueado do modelo de melhor desempenho.



Fonte: O autor, 2022.

Para chegar num resultado mais consistente, cada topologia é treinada e testada trinta vezes, mudando apenas o estado aleatório dos algoritmos de aprendizagem. Cada topologia é treinada sobre os dados de treinamento de suas bases de dados originarias. Após as trinta rodadas, os seus desempenhos são comparados graficamente e quantitativamente. O intuito dessa segunda comparação é a verificar se a base de dados não aleatorizada consegue gerar um modelo com uma capacidade de classificação semelhante à da base de dados aleatorizada, uma

vez que o procedimento de coleta da base não aleatorizada é muito mais simples e rápido que a da aleatorizada.

4.5 USO DO SISTEMA

Finalmente, com um modelo otimizado capaz de identificar os gestos especificados de maneira consistente, é proposto um sistema que consiga utilizar estes gestos para realizar algumas tarefas em um computador pessoal. No caso deste trabalho, as ações que são implementadas são simples e de pouco complexidade, pois o intuito desta terceira parte é servir como uma prova de conceito sobre a capacidade e flexibilidade deste tipo de aplicação. As ações que são possíveis de serem implementados são apenas limitadas pela criatividade e pela lógica de controle a ser desenvolvida.

Assim, podemos definir as seguintes ações:

1. ***Play/Pause* no *player* do sistema:** essa ação é acionada quando o usuário fecha sua mão;
2. **Avançar no navegador:** o acionamento desta ação é feito quando o usuário realiza o gesto que aponta sua mão para o lado, com a palma da mão virada contra a câmera, com a mão direita;
3. **Voltar no navegador:** esta ação é similar a avançar no navegador, no entanto, ela é acionada quando o usuário utiliza sua mãe esquerda;
4. ***Scroll down* no navegador:** essa ação é acionada quando o Usuário aponta três dedos para cima, com a palma da mão virada contra a câmera;
5. ***Scroll up* no navegador:** quando o Usuário aponta três dedos para baixo, com a palma da mão virada contra a câmera, essa ação é acionada;
6. **Avançar slide:** essa ação utiliza o mesmo gesto que a ação avançar no navegador, porém, ela é acionada quando o sistema identifica que a tela principal está em modo de apresentação;
7. **Voltar slide:** similar a ação anterior, essa ação uso o mesmo gesto que voltar no navegador, porém é realizada se a tela principal estiver em modo apresentação;
8. **Movimentar o *mouse*:** essa ação é feita quando o usuário aponta dois dedos para cima, com a palma da mão virada para a câmera;
9. **Apertar botão esquerdo do mouse:** quando o usuário aponta o seu polegar e dois dedos para cima, com a palma da mão virada para a câmera, essa ação é feita. O sistema mantém essa ação até o momento que o usuário muda de gesto, assim é possível tanto clicar em itens quanto arrastá-los.

5 RESULTADOS

Neste capítulo serão explorados os resultados obtidos das análises propostas no capítulo de Metodologia. Primeiro será abordado a análise proposta sobre as bases de dados. Em seguida, são comparados o desempenho dos modelos sobre a base aleatorizada. Depois disso, a técnica que resultou no modelo de melhor desempenho é utilizada para encontrar a topologia mais adequada, treinada sobre a base de dados não aleatorizada. Em seguida, é possível comparar os dois modelos e obtidos. Por fim, é feita uma demonstração sobre um sistema simples que utiliza a interface proposta.

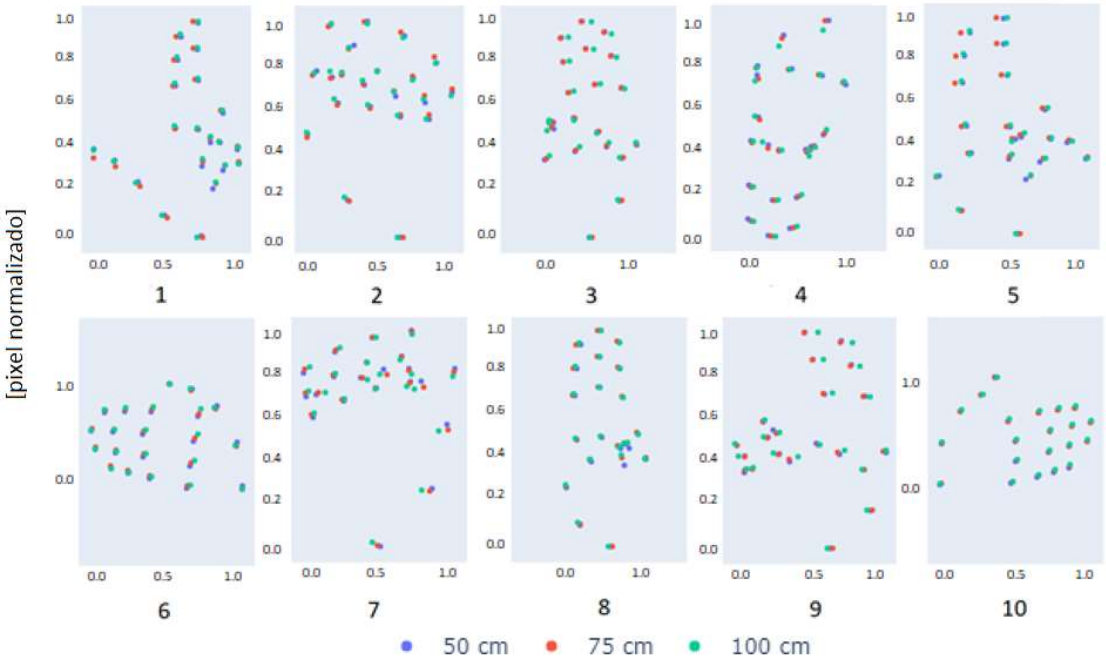
5.1 ANÁLISE DAS BASES DE DADOS

Nesta seção, serão analisadas as duas bases de dados que foram coletadas. Primeiramente será feita uma análise dos gestos médios de cada distância, para cada gesto. Em seguida é analisada a distribuição dos gestos que compõem cada base de dados.

5.1.1 Análise da base de dados aleatorizada

A base de dados aleatorizada reúne 450 amostras para cada gesto, sendo 150 amostras para cada uma das três distâncias definidas. Com isso, ela possui um total de 4500 amostras. Cada uma dessas amostras já foi previamente normalizada, conforme descrito na metodologia. Como primeira análise, podemos avaliar a diferença entre os gestos médios nas três distâncias, 50cm, 75 cm e 100 cm, para cada gesto. A maneira que esse gesto médio é construído é discutido no capítulo da Metodologia. A Figura 27 demonstra uma figura comparativa destes gestos médios, onde cada cor representa uma distância diferente.

Figura 27: Sobreposição dos gestos médios de cada distância da base de dados aleatorizada.

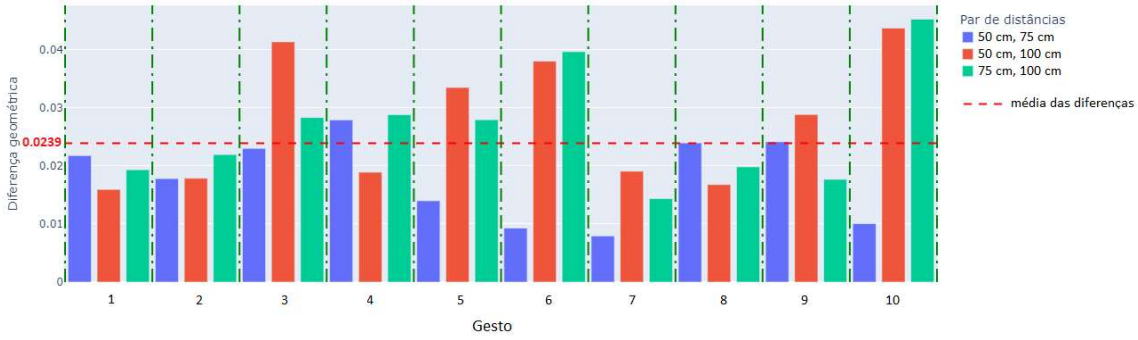


Fonte: O autor, 2022.

Usando a Equação 15 descrita na Metodologia podemos quantificar a diferença entre dois gestos médios executados a distâncias. Dessa forma, a maior diferença observada, de 0,04524 [pixel normalizado], foi entre a execução com 75 cm e 100 cm de distância para o gesto de identificação 3. Por sua vez, a menor diferença observada foi de 0,00794 [pixel normalizado], entre a execução com 50 cm e 75 cm do gesto de identificação 10.

Além disso, na Figura 28 é possível visualizar mais detalhadamente como são as diferenças geométricas entre cada gesto. Na figura, podemos analisar gesto a gesto qual foi a diferença medida, ainda utilizando a Equação 15. Mantendo o padrão de cores da Figura 27, a cor azul representa as diferenças entre os gestos médios feitas a uma distância de 50 cm e 75 cm. A cor vermelha remete a diferença entre os gestos médio de 50 cm e 75 cm. E a cor verde entre os gestos médios de 75 cm e 100 cm.

Figura 28: Diferenças geométricas entre cada gesto médio.

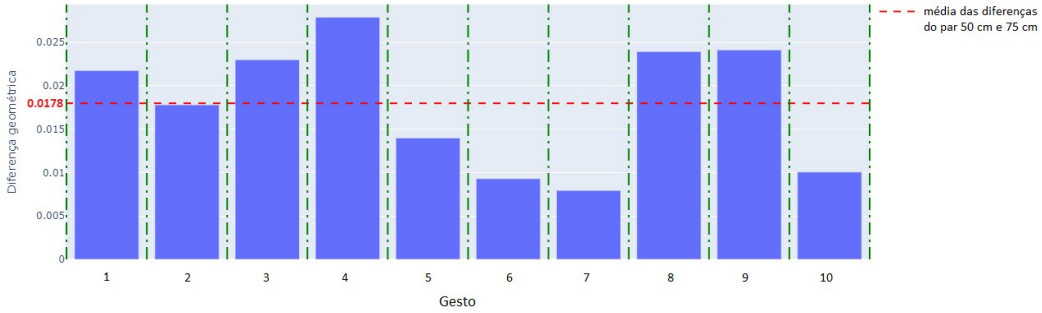


Fonte: O autor, 2022.

Analisando, podemos perceber que a diferença média entre todos os valores foi de 0,0239 [pixel normalizado]. Para a diferença dos gestos médios para o par de distâncias entre 50 cm e 75 cm (cor azul), apenas as diferenças para dois gestos foram maiores que está média. O par de distâncias entre 50 cm e 100 cm (cor vermelha) teve cinco valores de diferenças de gestos maiores que a média. Por sua vez, o par de distâncias entre 75 cm e 100 cm (cor verde) também teve cinco valores maiores que a média. Esses resultados indicam que há uma maior similaridade, de modo geral, entre os gestos médios do par de distâncias entre 50 cm e 75 cm.

Na Figura 29 podemos observar isoladamente as diferenças medidas para o par de distâncias entre 50 cm e 75 cm (cor azul). Podemos observar uma média de 0,0178 [pixel normalizado]. A maior diferença observada neste caso foi de 0,0279 [pixel normalizado] para o gesto número 4 e a menor diferença foi de 0,0079 [pixel normalizado] para o gesto 7.

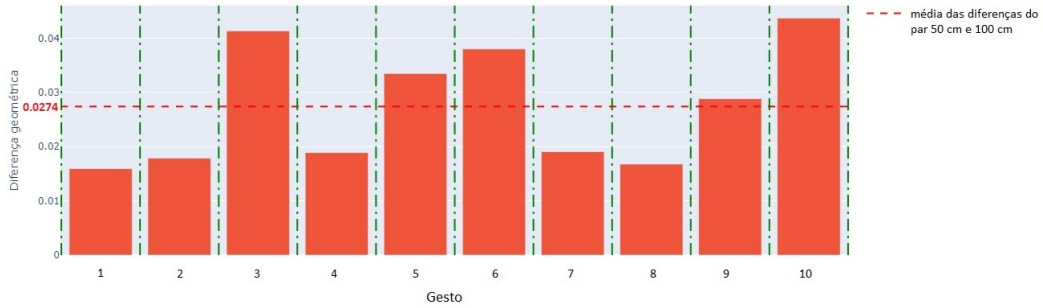
Figura 29: Diferenças geométricas dos gestos realizados a 50 cm e 75 cm.



Fonte: O autor, 2022.

Na Figura 30 são apresentadas as diferenças para o par de distâncias entre 50 cm e 100 cm (cor vermelha), onde foi observado uma diferença média de 0,0274 [pixel normalizado]. A maior diferença foi observada para o gesto número 10, no valor de 0,0437 [pixel normalizado]. Já a menor diferença geométrica foi para o gesto número 1, sendo de 0,0159 [pixel normalizado].

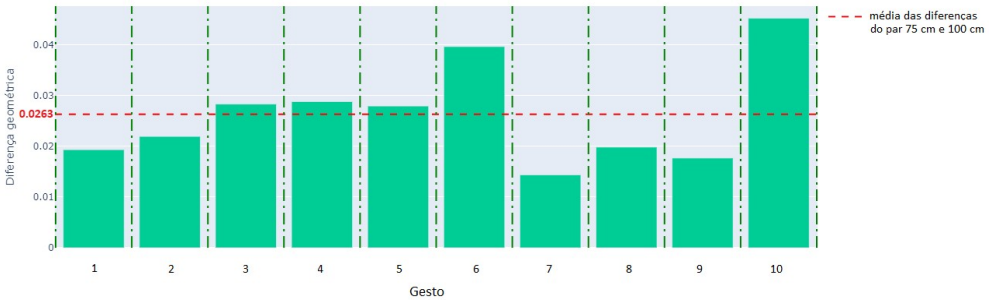
Figura 30: Diferenças geométricas dos gestos realizados a 50 cm e 100 cm.



Fonte: O autor, 2022.

Na Figura 31 podemos observar as diferenças geométricas para o par de distâncias entre 75 cm e 100 cm, onde foi medido uma diferença média de 0,0263 [pixel normalizado], a menor entre todas as cores. Para este caso, as maiores e menores diferenças foram de 0,0452 [pixel normalizado] e 0,0144 [pixel normalizado], respectivamente. A maior diferença foi observada para o gesto número 10 e a menor para o gesto número 7.

Figura 31: Diferenças geométricas dos gestos realizados a 75 cm e 100 cm.

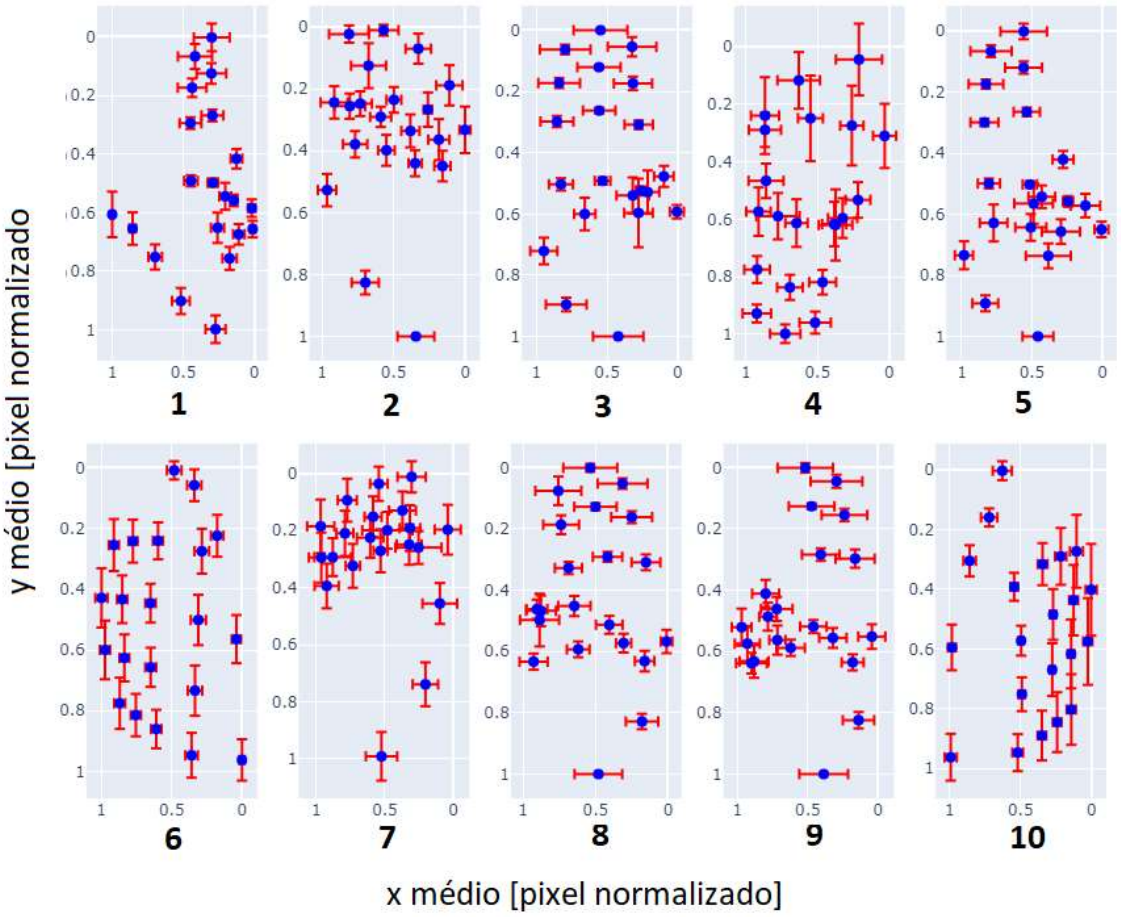


Fonte: O autor, 2022.

Com esses resultados, é possível concluir que o par de distância entre 50 cm e 100 cm teve uma maior diferença entre os seus gestos médios. Esse resultado é esperado devido as coletas

Em seguida, utilizando as equações 16 e 17 e seguindo a análise proposta na metodologia, é possível avaliar a distribuições dos pontos de cada gesto de maneira individual. Na Figura 32 é apresentado os resultados obtidos dessa análise.

Figura 32: Resultado da análise dos pontos de cada gesto, onde os pontos azuis são os valores médios e as faixas vermelhas têm um tamanho de um desvio padrão em cada sentido.



Fonte: O autor, 2022.

Com essa análise é possível observar o comportamento de cada ponto para os gestos. Como as faixas vermelhas têm um tamanho de um desvio padrão, em ambos os sentidos, podemos avaliar a variabilidade dos pontos. Os pontos que possuem uma faixa vermelha menor representam pontos no esqueleto da mão que se mantinham, de maneira mais consistente, em uma mesma posição na execução do gesto.

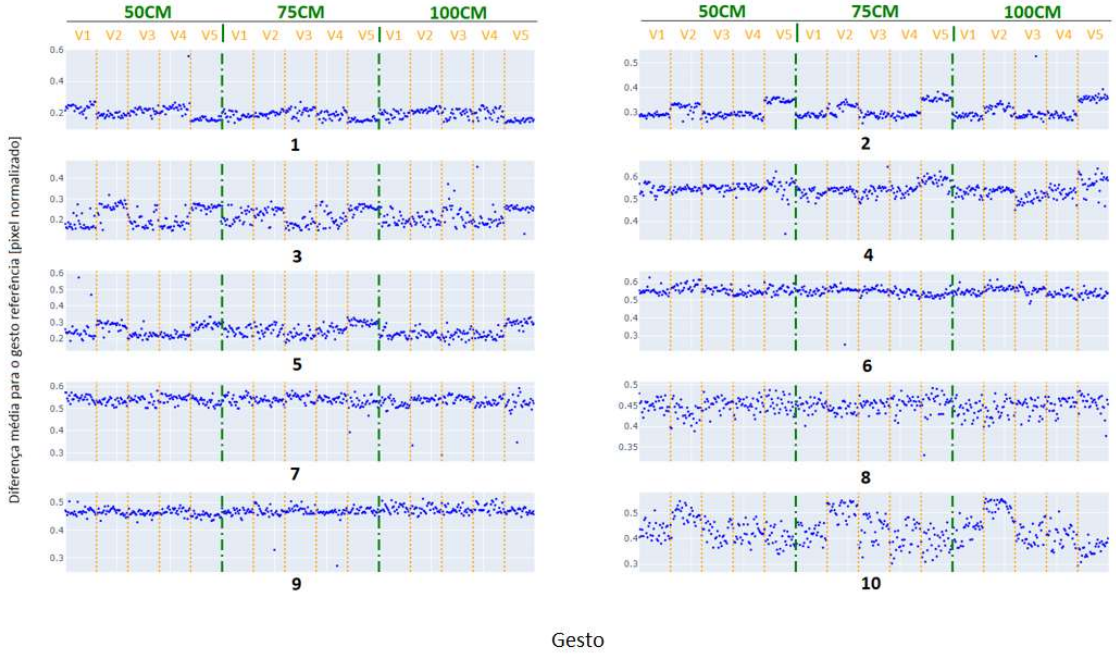
O gesto que apresentou o maior desvio padrão médio no eixo *x* é o gesto número 9 com um valor de 0,116 [pixel normalizado], e o que apresentou o menor foi o gesto número 10 com um valor de 0,035 [pixel normalizado]. Para o eixo *y*, por sua vez, o gesto que apresentou o maior desvio padrão médio foi o gesto 10, com um valor de 0,085 [pixel normalizado] e o que apresentou o menor foi o gesto número 3, com apenas 0,028 [pixel normalizado]. Além disso, observando os resultados, podemos perceber os gestos que foram mais fáceis ou difíceis de serem executados. Para o gesto 1, por exemplo, podemos notar que de modo geral que a dispersão dos seus 21 pontos é baixa, o que pode indicar que a execução do gesto é feita de modo mais confortável para o usuário do que gestos como de número 4, que, avaliando diretamente no gráfico, claramente podemos notar uma dispersão maior na posição dos seus pontos.

Além disso, em alguns gestos podemos notar que apenas alguns pontos possuem uma dispersão mais elevada, como o caso dos gestos 6 e 10, onde os pontos mais à esquerda e a direita, respectivamente, demonstram uma variação maior no eixo *y*. Isso demonstra que os

voluntários tiveram maior dificuldade nas execuções destes gestos para alinhar com o exemplo sugerido.

Ainda, podemos perceber essas características de dificuldade de execução de cada gesto quando comparamos cada amostra com o gesto de referência definido na metodologia. Na Figura 33 é apresentado a distância média de cada amostra para esse gesto. Nesta figura podemos segmentar os pontos tanto pelo voluntário que executou as amostras quanto pela distância da *webcam* durante a execução. Cada voluntário é identificado pelo símbolo “V1”, “V2”, e assim por diante.

Figura 33: Distância média de cada amostra em relação ao gesto de referência.



Fonte: O autor, 2022.

Analisando essa figura, primeiro é importante levar em consideração que como a métrica representa uma distância geométrica para um gesto de referência, um valor maior não significa uma característica negativa.

Na figura, podemos perceber como um voluntário reproduziu cada gesto em cada uma das distâncias. Em alguns gestos, alguns voluntários apresentaram um comportamento específico na maneira com que executam o gesto. Como é o caso do voluntário 5 durante o gesto número 2: neste caso o voluntário apresentou valores maiores que os demais voluntários, porém com uma dispersão baixa, o que indica que o voluntário talvez tenha realizado o gesto com a mão com alguma inclinação maior que os demais. Além disso, esse comportamento se mantém para as três distâncias.

Esse tipo de situação também acontece para o voluntário 2 durante o gesto 2, para o voluntário 5 durante o gesto 1 e para o voluntário 2 durante o gesto 10, por exemplo. Esses comportamentos característicos demonstram alguma tendência anatômica do voluntário de executar o gesto alguma inclinação ou com algum giro em algum eixo de sua mão.

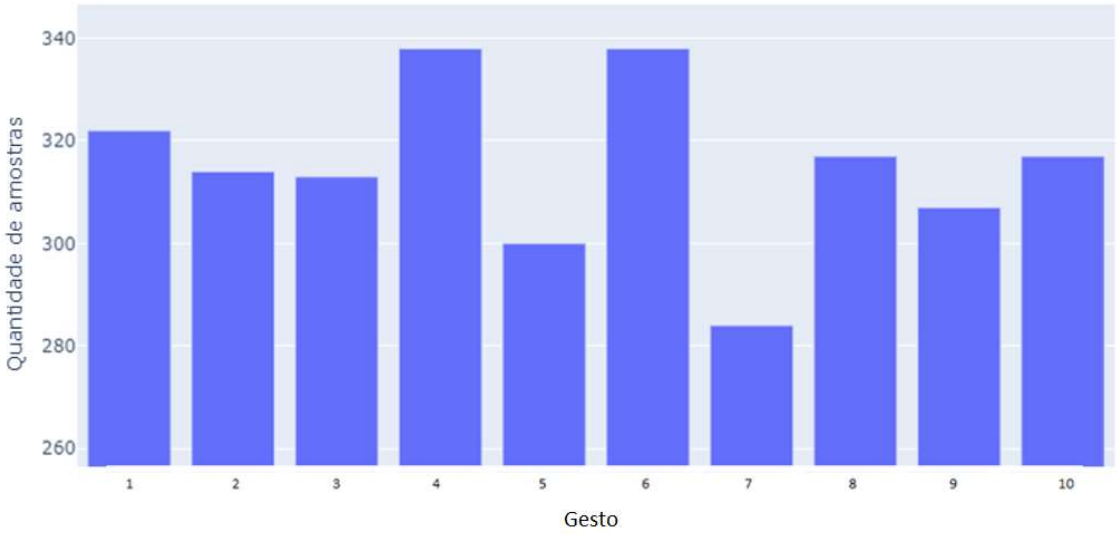
Além desses comportamentos característicos de cada voluntário, podemos, também, perceber que apesar das diferentes distâncias, os valores se mantiveram semelhantes para

todas as distâncias. Isso indica que os voluntários conseguiram realizar de maneira consistente os gestos durante todo o experimento, sem haver algum viés em uma distância específica.

Ainda com base na Figura 33, podemos também notar que para alguns gestos, os pontos se mostraram com uma alta variabilidade para cada voluntário, como é o caso do gesto número 8 e 10, por exemplo. Nestes casos, esta variabilidade pode estar relacionada a dificuldade de execução do gesto, podendo assumir uma posição desconfortável e cansativa para a mão do voluntário. Por outro lado, para alguns gestos, como o gesto 6 e 9, por exemplo, houve uma variabilidade menor e os valores tenderam a ficar perto de uma média, o que pode indicar que realizar este gesto foi mais confortável para o voluntário.

Com essa análise sobre a base de dados concluída, é possível agora prepará-la para ser utilizada para treinar modelos inteligentes. Para isso ela então é dividida em três partes. Conforme exposto na Metodologia, 70% das amostras dessa base de dados são destinadas a treinamento de modelos, 15% destinadas a validação e 15% destinadas a testes. Para concluir esse capítulo, podemos verificar o equilíbrio dos dados que serão destinados a treinamento. Assim, na Figura 34 é apresentado a distribuição dos gestos dos dados de treinamento.

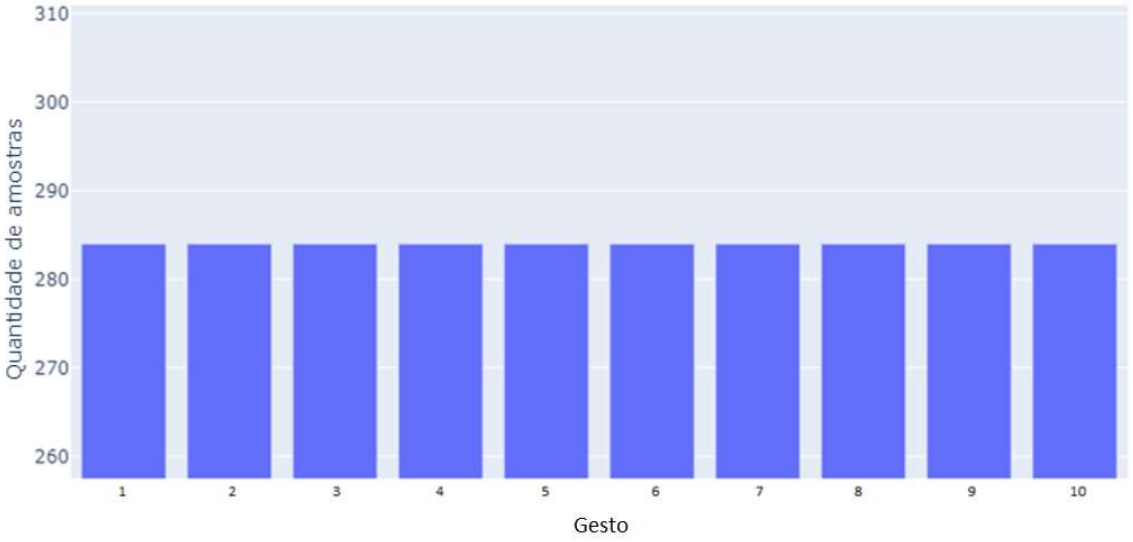
Figura 34: Distribuição dos dados de treinamento antes de serem equilibrados.



Fonte: O autor, 2022.

Podemos ver um claro desequilíbrio nos dados de treinamento. Os gestos 4 e 6 contêm a maior quantidade de amostras, sendo 338 cada. Por outro lado, o gesto com menor quantidade é o gesto 7, com apenas 284. Para equilibrar esta parte de treinamento, é estabelecido uma quantidade máxima de 284 amostras para cada gesto, onde o excedente é descartado. Dessa forma, na Figura 35 é possível observamos os dados de treinamento após ser equilibrada

Figura 35: Distribuição equilibrada dos dados de treinamento.

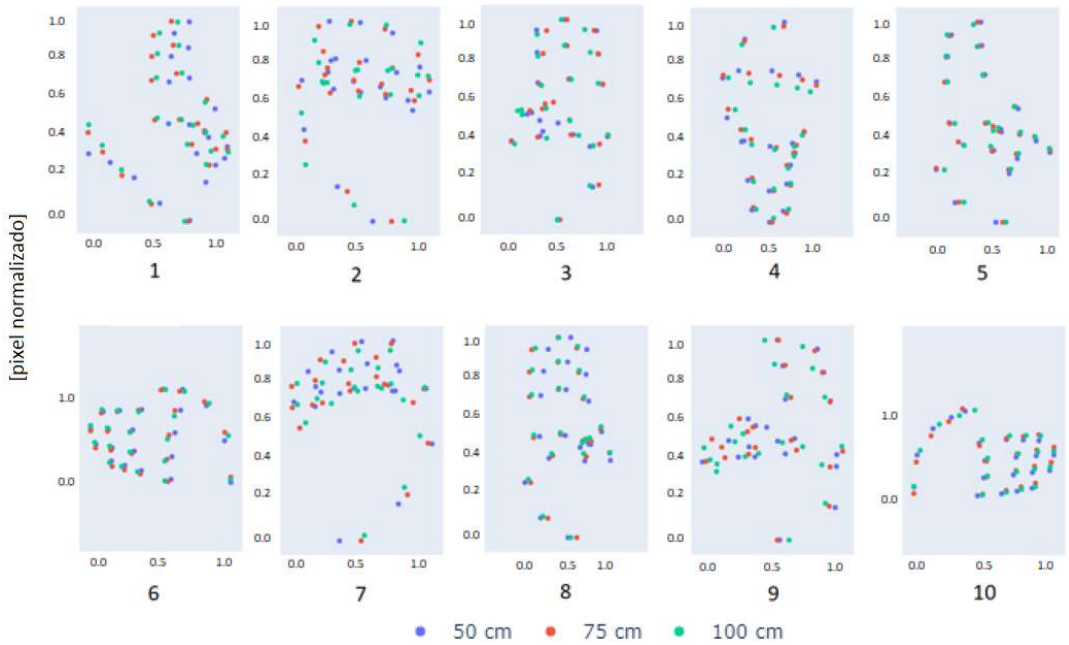


Fonte: O autor, 2022.

5.1.2 Análise da base de dados não aleatorizada

A base de dados não aleatorizada conta com 4500 amostras para cada gesto, sendo 1500 para cada distância. Assim, ela tem em sua totalidade 45000 amostras de gestos. Assim como a base aleatorizada, estas amostras foram normalizadas. De maneira análoga a análise da base de dados aleatorizada, podemos começar verificando os gestos médios para cada distância. A Figura 36 apresenta uma visualização comparativa dos gestos, onde cada cor representa uma distância.

Figura 36: Resultado da análise da base de dados não aleatorizada.

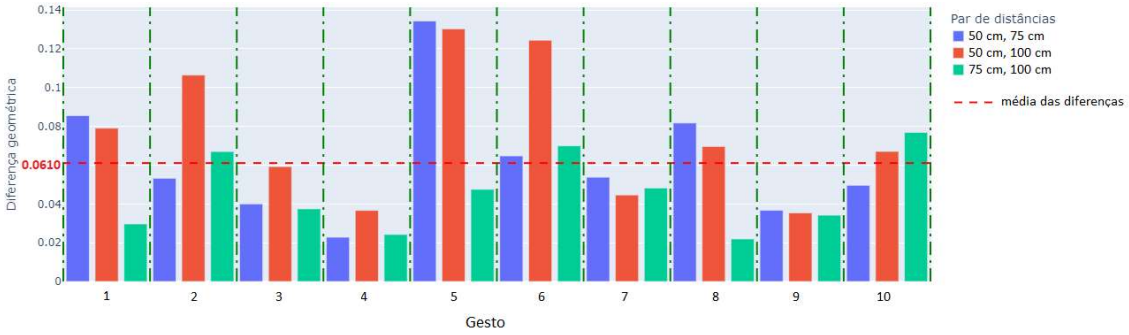


Fonte: O autor, 2022.

Utilizando a Equação 15 podemos quantificar a diferença entre cada gesto médio. A maior diferença obtida foi de 0,13434 [pixel normalizado] entre a execução com 50 cm e 100 cm de distância para o gesto de identificação 7. A menor diferença foi de 0,02191 [pixel normalizado] entre a execução com 50 cm e 75 cm de distância para o gesto de identificação 8.

Além disso, na Figura 37 podemos entender de forma mais detalhada como são as diferenças geométricas entre os gestos. Na figura, podemos analisar gesto a gesto qual a diferença medida, ainda utilizando a Equação 15. Mantendo o padrão de cores da Figura 36, o par de distâncias entre 50 cm e 75 cm (cor azul) representa as diferenças entre os gestos médios feitas a uma distância de 50 cm e 75 cm. O par de distâncias entre 50 cm e 100 cm (cor vermelha) remete a diferença entre os gestos médio de 50 cm e 75 cm. E o par de distâncias entre 75 cm e 100 cm (cor verde) entre os gestos médios de 75 cm e 100 cm.

Figura 37: Diferenças geométricas entre cada gesto médio.

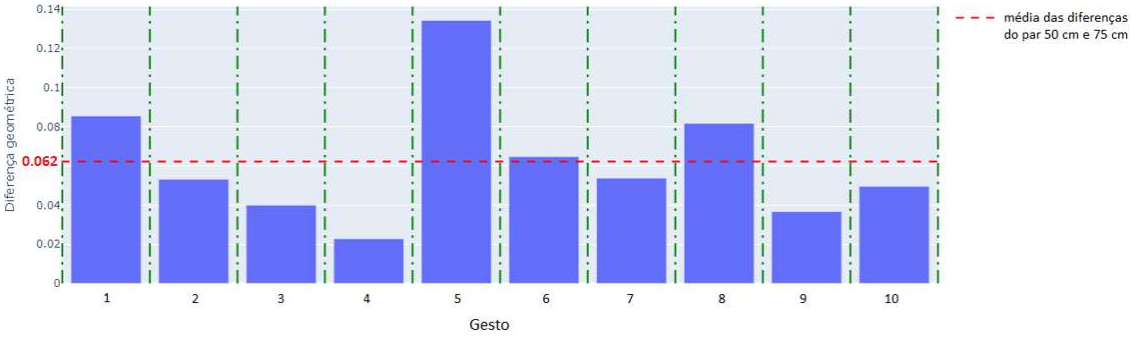


Fonte: O autor, 2022.

Ainda, podemos perceber que a diferença média entre todos os valores foi de 0,061 [pixel normalizado]. Para o par de distâncias entre 50 cm e 75 cm, as diferenças para quatro gestos foram maiores que está média. O par de distâncias entre 50 cm e 100 cm (cor vermelha) teve a maior quantidade de diferenças de gestos maiores que a média, ocorrendo 6 vezes. Por sua vez, o par de distâncias entre 75 cm e 100 cm teve a menor quantidade, ocorrendo apenas três vezes.

Na Figura 38 podemos observar isoladamente as diferenças medidas para o par de distâncias entre 50 cm e 75 cm (cor azul) e podemos observar uma média de 0,0622 [pixel normalizado]. A maior diferença observada neste caso foi de 0,1343 [pixel normalizado] para o gesto número 5 e a menor diferença foi de 0,0228 [pixel normalizado] para o gesto número 4.

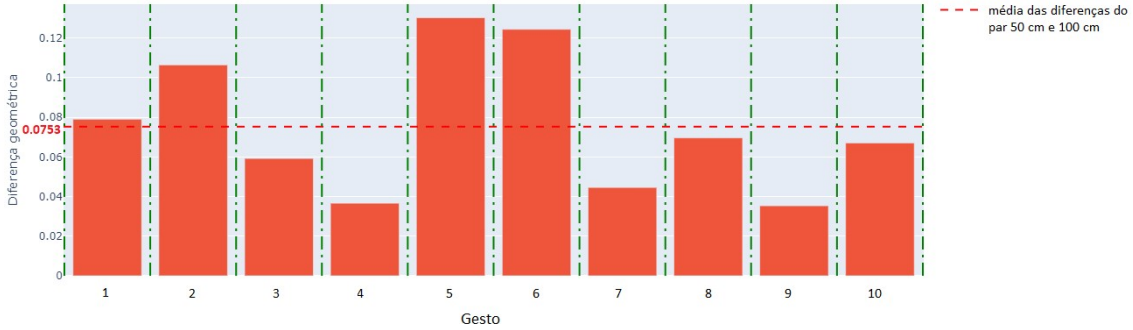
Figura 38: Diferenças geométricas dos gestos realizados a 50 cm e 75 cm.



Fonte: O autor, 2022.

Na Figura 39 é apresentado as diferenças para o par de distâncias entre 50 cm e 100 cm, onde foi observado uma diferença média de 0,0753 [pixel normalizado]. A maior diferença foi observada para o gesto número 5, no valor de 0,1303 [pixel normalizado]. Já a menor diferença geométrica foi para o gesto número 9, sendo de 0,0353 [pixel normalizado].

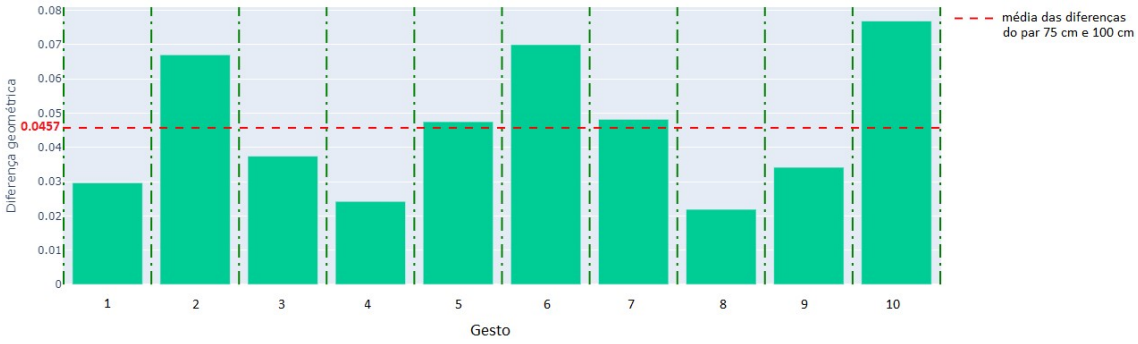
Figura 39: Diferenças geométricas dos gestos realizados a 50 cm e 100 cm.



Fonte: O autor, 2022.

Já na Figura 40 podemos observar as diferenças geométricas para o par de distâncias entre 75 cm e 100 cm (cor verde), onde foi medido uma diferença média de 0,0457 [pixel normalizado], a menor entre todas as cores. Para este caso, as maiores e menores diferenças foram de 0,0768 [pixel normalizado] e 0,0219 [pixel normalizado], respectivamente. A maior diferença foi observada para o gesto número 10 e a menor para o gesto número 8.

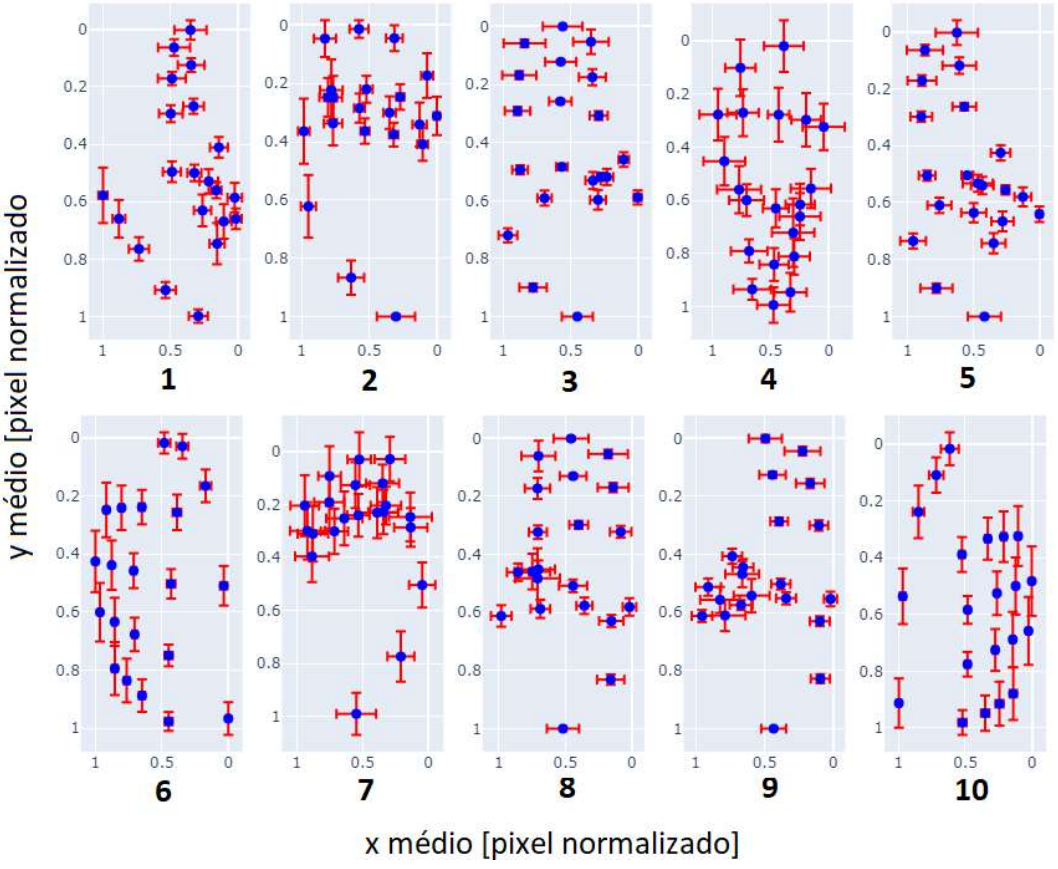
Figura 40: Diferenças geométricas dos gestos realizados a 50 cm e 100 cm.



Fonte: O autor, 2022.

Na sequência, da mesma para a seção anterior, podemos realizar a análise a respeito das distribuições dos pontos de cada gesto de maneira individual. Para isso, são utilizadas as Equações 16 e 17 para calcular a média e o desvio padrão para cada um dos vinte e um pontos do esqueleto. Com essas duas medidas podemos construir a Figura 41, onde os pontos de cor azul são os valores médios de cada ponto, e as faixas vermelhas possuem um tamanho de um desvio padrão, de acordo com o eixo em questão.

Figura 41: Resultado da análise dos pontos de cada gesto, onde os pontos azuis são os valores médios e as faixas vermelhas têm um tamanho de um desvio padrão em cada sentido.



Fonte: O autor, 2022.

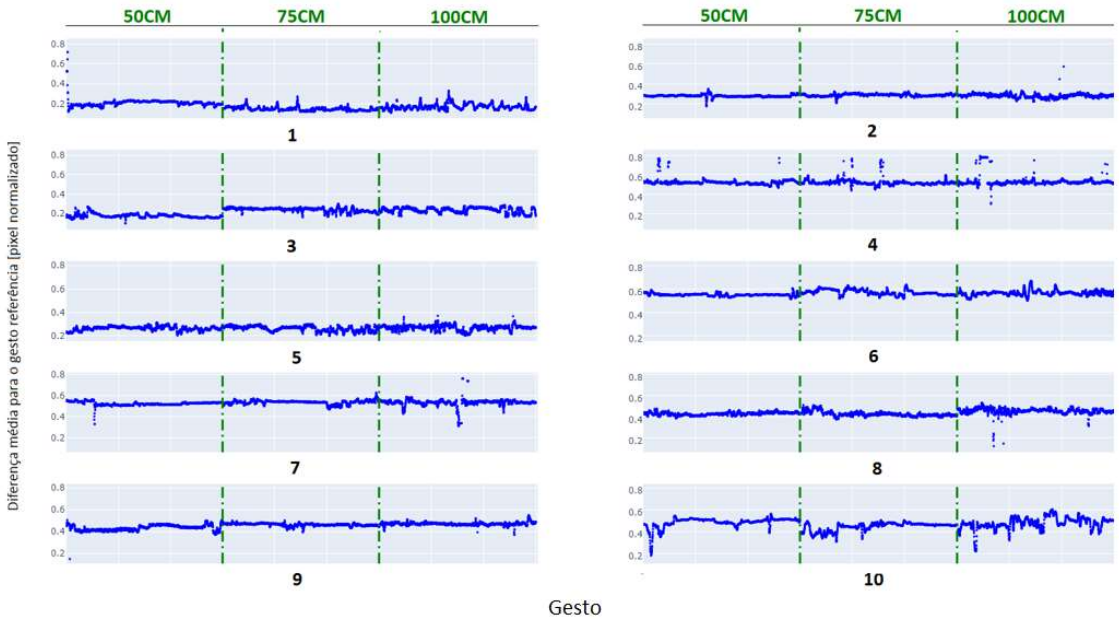
A partir dessa figura, podemos notar como cada ponto se distribuiu no espaço conforme o único voluntário, no caso dessa base de dados, os executava. É possível observar que alguns gestos possuem mais pontos com faixas vermelhas pequenas, como o caso do gesto 1 e 3, assim como outros possuem faixas vermelhas maiores de maneira generalizada, como para o gesto 4 e 7, por exemplo. Ainda, alguns gestos, como os gestos 6 e 10, têm alguns pontos com uma faixa vermelha mais acentuada apenas seguindo o eixo *y*.

O gesto que apresentou o maior desvio padrão médio no eixo *x* é o gesto número 4, com um valor de 0,149 [pixel normalizado], e o que apresentou o menor foi o gesto número 6, com um valor de 0,022 [pixel normalizado]. Para o eixo *y*, por sua vez, o gesto que apresentou o maior desvio padrão médio foi novamente o gesto 4, com um desvio padrão de 0,081 [pixel normalizado] e o que apresentou o menor foi o gesto número 3, com apenas 0,182 [pixel normalizado].

Além disso, é possível notar que os resultados foram semelhantes com os da base aleatorizada, obtidos na seção anterior. Isso demonstra a compatibilidade dos dados de cada base de dados.

Além desses resultados, podemos comparar as amostras obtidas para essa base de dados com o gesto de referência definido na metodologia. Na Figura 42 é possível observar como se deu a diferença de cada amostra com esse gesto. Neste caso, como essa base de dados foi coletada a partir de um único voluntário, a figura foi segmentada apenas segundo as distâncias da *webcam* durante a execução dos gestos.

Figura 42: Distância média de cada amostra em relação ao gesto de referência.



Fonte: O autor, 2022.

Para analisar esses resultados primeiro é importante ressaltar que a métrica que está sendo utilizada representa uma diferença geométrica do gesto para um gesto de referência escolhido arbitrariamente. Dessa forma, um valor elevado não remete uma característica negativa para o gesto em questão. Ainda, no caso desta base de dados, é importante relembrar

que ela foi coletada utilizando uma captura contínua e em tempo real, assim, é esperado que os pontos tenham uma correlação temporal.

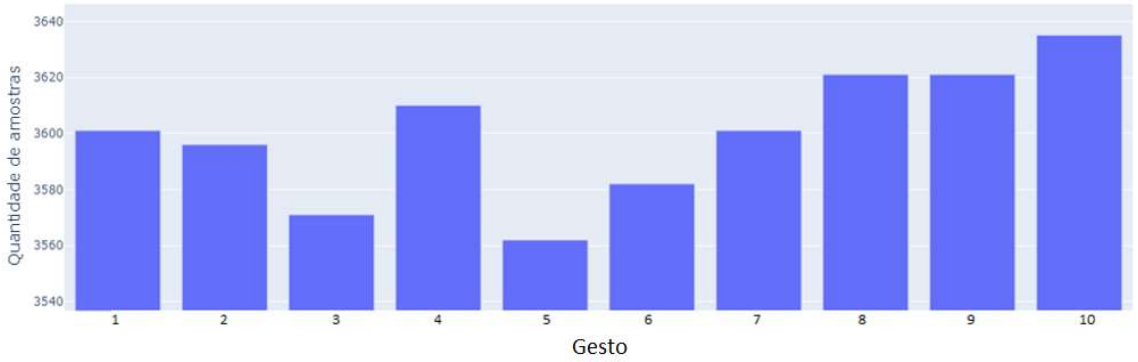
Com isso, podemos perceber que os valores obtidos para os gestos têm, de maneira geral, uma dispersão menor que os resultados obtidos com cinco voluntários. Isso indica que as amostras têm uma variabilidade menor que o caso anterior. Ainda, podemos notar que para alguns gestos, ao longo da coleta houve algumas amostras que demonstram uma diferença ligeiramente maior que as demais, o que pode indicar que o voluntário tenha executado o gesto de maneira diferente ou errônea, como para o caso dos gestos 2, 4 e 6, por exemplo.

Além disso, também é possível perceber que em alguns casos o voluntário executou o gesto com algumas alterações para adicionar variabilidade nos dados, conforme descrito na metodologia. Esse caso pode ser exemplificado pelo gesto 3, 5 e 10 de maneira mais clara. Em seus gráficos, podemos ver que houve situações que os valores da métrica aumentam ou diminuem de maneira mais consistente, sem ser casos pontuais.

Ainda, no caso dessa base de dados, é possível observar para alguns gestos, a troca de distância não afetou consideravelmente a métrica de avaliação das amostras, como é o caso dos gestos 2, 4, 7 e 9. Já para os casos, a troca da distância resultou ou numa mudança do valor médio da métrica, como observado no gesto 3, como uma mudança na dispersão dos valores obtidos, como o caso do gesto 10, por exemplo.

Após que a análise da base de dados como um todo seja concluída, podemos prepará-la para ser utilizado nas próximas etapas de treinamento de modelos. Essa base de dados é dividida em apenas duas partes: 80% das amostras pertencem a parte de treinamento e 20% pertencem a parte de validação. Por última análise podemos verificar o equilíbrio dos dados que serão destinados a treinamento pelos mesmos motivos expressos para seção anterior. Assim, na Figura 43 é apresentado a distribuição dos gestos dos dados de treinamento.

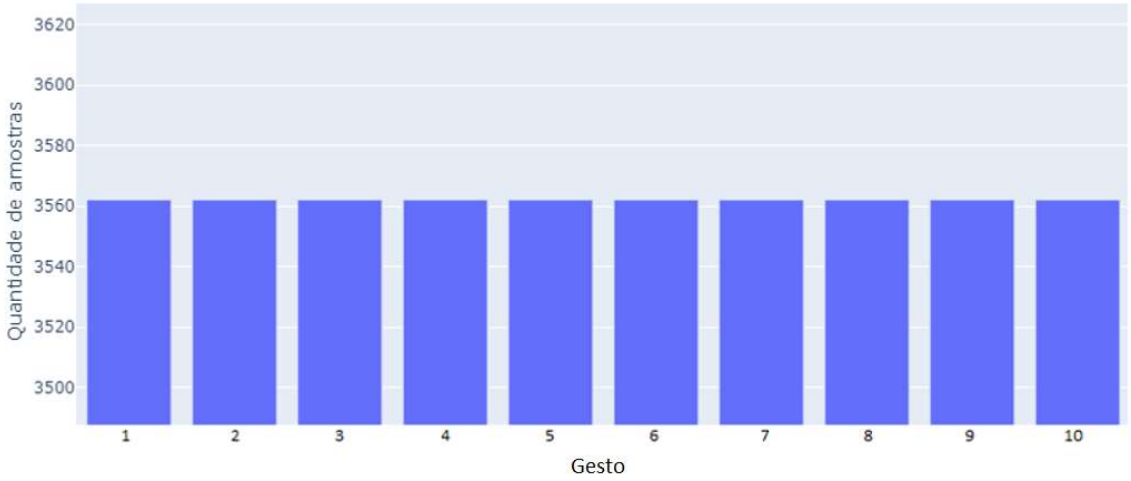
Figura 43: Distribuição dos dados de treinamento antes de serem equilibrados.



Fonte: O autor. 2022.

Claramente há um desequilíbrio nos dados. O gesto de identificação 10 é o gesto com maior quantidade de amostras, tendo 3635 amostras. Por sua vez, o gesto com identificação 5 é o que possui menor amostras, tendo apenas 3562 amostras. Dessa forma, para equilibrar os dados de treinamento, é estabelecido um limite de 3562 amostras para cada gesto e o excedente é descartado. A Figura 44 representa os dados de treinamento já equilibrados.

Figura 44: Distribuição equilibrada dos dados de treinamento.



Fonte: O autor. 2022.

5.2 TREINAMENTO DE MODELOS INTELIGENTES

Uma vez que as bases de dados sejam coletadas e previamente preparadas, podemos partir para a etapa de treinar diversos modelos, baseados em diferentes técnicas, para encontrar o mais adequado para a situação. Para esse fim, primeiramente será utilizado a base de dados aleatorizada. Sobre ela é feito uma etapa de otimização de hiper parâmetros para cada uma das técnicas. Nessa etapa, cada modelo é treinado sobre os dados de treinamento e avaliado sobre os dados de validação e a melhor combinação é então avaliada sobre os dados de teste. Assim, é possível elencar o modelo que teve os melhores resultados sobre os dados de teste.

Uma vez que esse modelo é encontrado, é feita uma nova etapa de otimização de hiper parâmetros para essa mesma técnica, sobre a base de dados não aleatorizada. Nesta etapa, as mesmas combinações de hiper parâmetros da etapa anterior são exploradas. Por fim, o modelo otimizado sobre os dados de treinamento e de validação da base não aleatorizada é testado sobre os dados de teste da base aleatorizada e comparado com o que foi treinado sobre a base aleatorizada.

5.2.1 Otimização de modelos

Neste capítulo são apresentados apenas as combinações de hiper parâmetros (topologias) que resultaram nos melhores desempenhos. Além disso, as possiblidades de combinação são definidas no capítulo de Metodologia.

5.2.1.1 Otimização do modelo baseado em regressão logística

Foram explorados um total de 8 topologias diferentes. A Tabela 8 apresenta as três topologias que obtiveram os melhores resultados. Sendo elas:

- Topologia 1: Usou o algoritmo de otimização “sag” e a técnica L2 de regularização;
- Topologia 2: Usou o algoritmo de otimização “lbfgs” e técnica L2 de regularização;

- Topologia 3: Usou o algoritmo de otimização “sag” e nenhuma técnica de regularização.

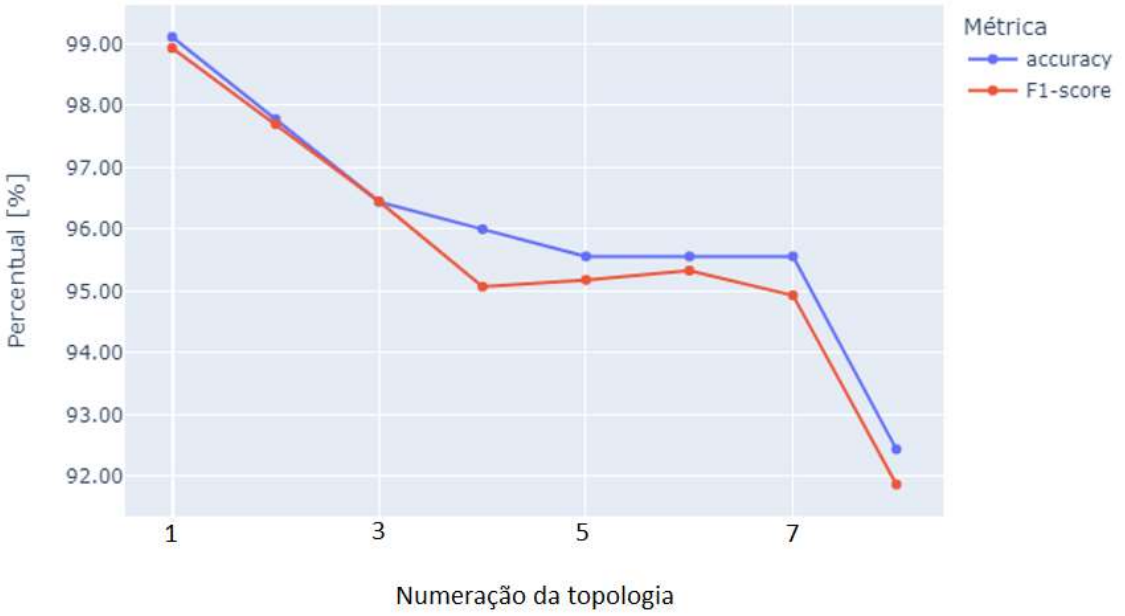
Tabela 8: Resultados das melhores topologias baseado em regressão logística.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	98,67%	98,34%
Topologia 2	97,78%	97,64%
Topologia 3	97,33%	97,35%

Fonte: O autor, 2022.

Desta otimização de hiper parâmetros é possível perceber que a melhor topologia é a que utiliza o algoritmo de otimização “sag” e utiliza técnica de regularização. O uso dessa técnica de regularização se mostrou o diferencial entre a melhor topologia e a terceira melhor, que utiliza o mesmo algoritmo de otimização. Na Figura 45, é possível observar o desempenho de todas as topologias testadas.

Figura 45: Desempenho de todas as topologias baseadas em regressão logística.



Fonte: O autor, 2022.

As duas topologias com os piores desempenho, por sua vez, não utilizaram técnica de regularização. A com pior desempenho utilizou o algoritmo de otimização “sag” e a com o segundo pior desempenho utilizou o algoritmo “netwon-cg”.

5.2.1.2 Otimização do modelo baseado em Classificador Bayes Ingênuo

Devido a simplicidade do algoritmo que fundamenta esse classificador, não houve hiper parâmetros a serem explorados. Assim, foi treinado um único modelo neste caso e a Tabela 9 apresenta seus resultados.

Tabela 9: Resultado do modelo.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia	98,22%	89,50%

Fonte: O autor, 2022.

5.2.1.3 Otimização do modelo baseado em Máquina de Suporte de Vetor

Para essa otimização foram treinadas apenas quatro topologias, sendo elas:

- Topologia 1: usou o kernel rbf;
- Topologia 2: usou o kernel linear;
- Topologia 3: usou o kernel poly;
- Topologia 4: usou o kernel sigmoid.

A Tabela 10 apresenta os resultados obtidos.

Tabela 10: Resultados das topologias exploradas.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	98,22%	98,12%
Topologia 2	98,22%	97,74%
Topologia 3	97,78%	97,38%
Topologia 4	64,00%	54,14%

Fonte: O autor, 2022.

Nesta comparação podemos perceber que o desempenho da topologia 4 foi significativamente pior que o as demais topologias. Estas demais topologias obtiveram resultados parecidos.

5.2.1.4 Otimização do modelo baseado em Árvore de Decisão

Novamente foram exploradas apenas quatro combinações de hiper parâmetros. A Tabela 11 apresenta os resultados de cada topologia. As topologias são definidas como:

- Topologia 1: usou splitter “random”, e critério “gini”;
- Topologia 2: usou splitter “best”, e critério “gini”;
- Topologia 3: usou splitter “best”, e critério “entropy”;

- Topologia 4: usou splitter “random”, e critério “entropy”.

Tabela 11: Resultados das topologias exploradas.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	99,11%	99,09%
Topologia 2	97,78%	97,62%
Topologia 3	97,33%	96,09%
Topologia 4	96,89%	96,90%

Fonte: O autor, 2022.

Analisando os resultados, é possível observar que a topologia 1 teve um desempenho mais notável que as demais topologias. As outras três topologias tiveram resultados mais próximos, diferindo pouco entre si.

5.2.1.5 Otimização do modelo baseado em Floresta Aleatória

Foram exploradas um total de 32 topologias diferentes, onde as três que tiveram um melhor desempenho foram:

- Topologia 1: usou 64 estimadores, usou *bootstrap* e critério “gini”;
- Topologia 2: usou 16 estimadores, usou *bootstrap* e critério “gini”;
- Topologia 3: usou 32 estimadores, usou *bootstrap* e critério “entropy”.

A Tabela 12 apresenta o resultado dessas topologias.

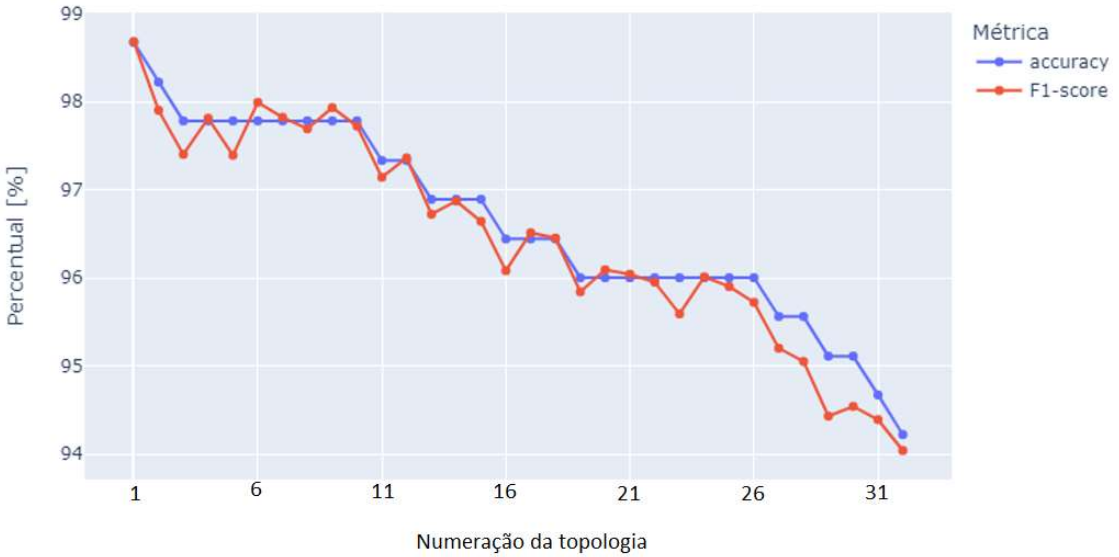
Tabela 12: Resultado das três melhores topologias exploradas.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	98,67%	98,68%
Topologia 2	98,22%	97,90%
Topologia 3	97,78%	97,40%

Fonte: O autor, 2022.

As três melhores topologia utilizaram o hiper parâmetro *bootstrap*, onde a que teve o melhor desempenho conta, ainda com 64 estimadores e utilizou o critério “gini”. Na Figura 46 é possível analisar o desempenho das demais topologias testadas.

Figura 46: Desempenho de todas as topologias baseadas em floresta aleatória.



Fonte: O autor, 2022.

A topologia que obteve o pior desempenho em ambas as métricas utilizou apenas 2 estimadores, não utilizou *bootstrap* e usou o critério "gini". A segundo pior topologia utilizou 4 estimadores e o critério "entropy". Ela também não utilizou o hiper parâmetro *bootstrap*.

5.2.1.6 Otimização do modelo baseado em *XGBoost*

Neste caso foi possível treinar 72 modelos com topologias diferentes. A Tabela 13 demonstra o desempenho das três melhores topologias, que estão apresentadas abaixo:

- Topologia 1: usou 128 estimadores, usou uma taxa de aprendizado de 0,3 e uma política de crescimento "depthwise";
- Topologia 2: usou 128 estimadores, usou uma taxa de aprendizado de 0,2 e uma política de crescimento "depthwise";
- Topologia 3: usou 32 estimadores, usou uma taxa de aprendizado de 0,4 e uma política de crescimento "lossguide".

Tabela 13: Resultados das melhores topologias.

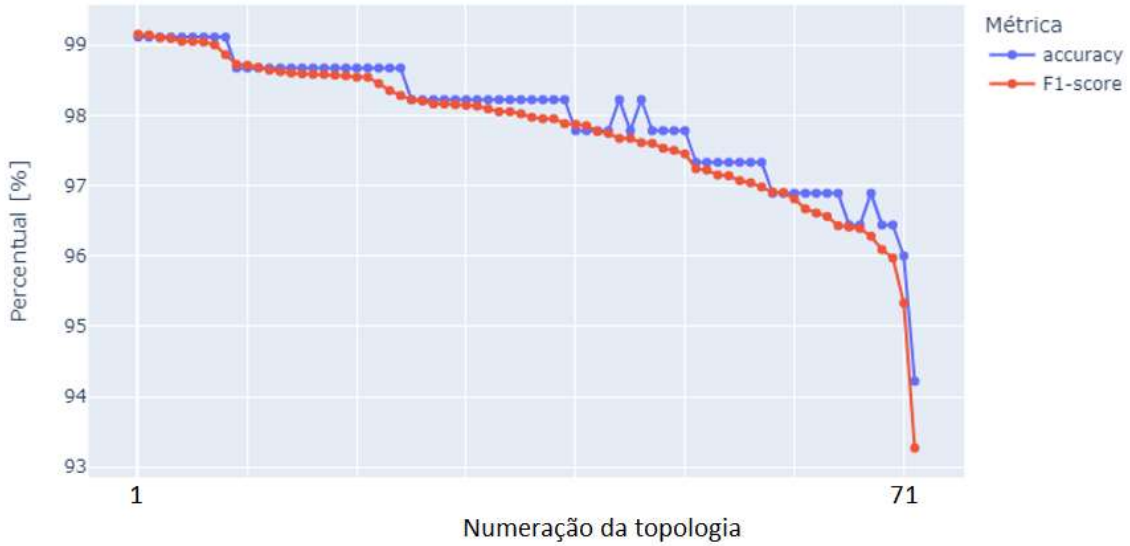
Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	99,11%	99,15%
Topologia 2	99,11%	99,14%
Topologia 3	99,11%	99,05%

Fonte: O autor, 2022.

Com base na tabela de resultados podemos perceber que a única diferença no desempenho dos modelos foram os valores obtidos na métrica *F1-score*. Essa diferença se deve a forma com que essa métrica é calculada, como sendo uma média harmônica entre a precisão

e cobertura, sendo a Equação 9. Além desses resultados, na Figura 47 podemos avaliar o desempenho do restante das topologias analisadas.

Figura 47: Desempenho de todas as topologias baseadas em *XGBoost*.



Fonte: O autor, 2022.

A topologia que obteve os piores resultados usou 32 estimadores, teve uma taxa de aprendizado de 0,4 e uma política de crescimento “depthwise”.

5.2.1.7 Otimização do modelo baseado em rede neural simples

Foram avaliadas 72 diferentes topologias baseadas em uma rede neural simples, onde as três que obtiveram os melhores resultados são:

- Topologia 1: Usou 1024 neurônios em sua única camada oculta, usou o algoritmo otimizador “adam” e usou a função de ativação “sigmoid” em sua camada oculta;
- Topologia 2: Usou 4096 neurônios em sua única camada oculta, usou o algoritmo otimizador “adam” e usou a função de ativação “relu” em sua camada oculta;
- Topologia 3: Usou 4096 neurônios em sua única camada oculta, usou o algoritmo otimizador “adam” e usou a função de ativação “sigmoid” em sua camada oculta.

Na Tabela 14 é possível observar os resultados delas.

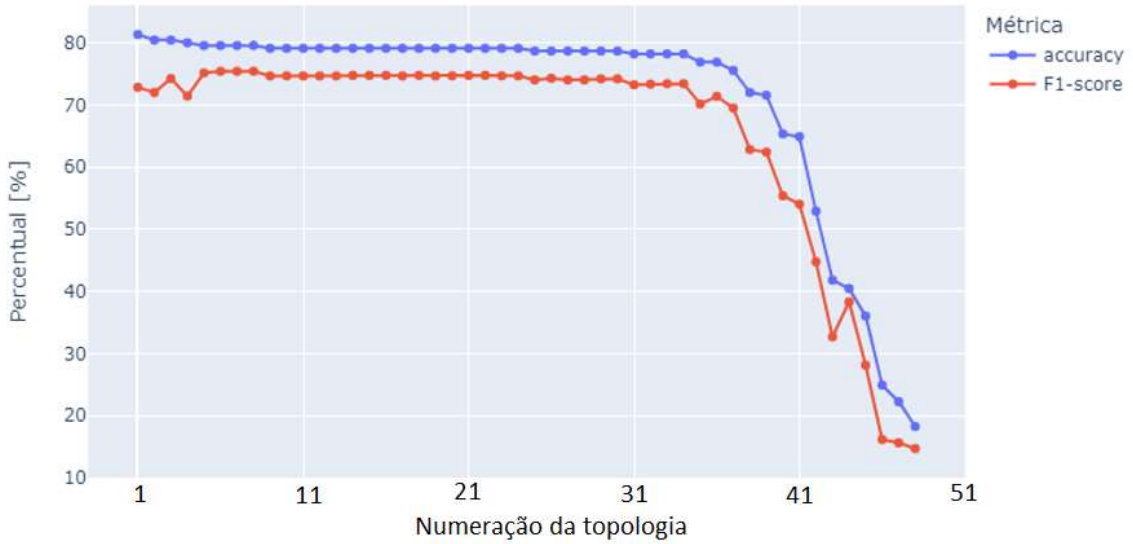
Tabela 14: Resultado das três melhores topologias.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	81.33%	72.83%
Topologia 2	80.44%	74.24%
Topologia 3	80.44%	72.00%

Fonte: O autor, 2022.

A topologia 1 apresentou o melhor resultado em taxa de acerto entre as topologias analisadas. Por sua vez, a topologia 2 apresentou um valor de *F1-Score* mais elevado, porém com uma taxa de acerto menor. Ademais, a Figura 48 demonstra o desempenho de todas as topologias exploradas nesse processo de otimização de hiper parâmetros.

Figura 48: Desempenho de todas as topologias baseadas em uma rede neural simples.



Fonte: O autor, 2022.

A topologia explorada que obteve os piores resultados contava com 2048 neurônios na sua camada oculta, utilizou o algoritmo de otimização “sgd” e a função de ativação “sigmoid”.

5.2.1.8 Otimização do modelo baseado em *MLP*

Para esse caso, foram definidas um total de 6912 possível topologias com as combinações dos hiper parâmetros apresentados no capítulo de Metodologia. No entanto, por questão de tempo e complexidade computacional, apenas 40 dessas topologias foram treinadas. Elas foram escolhidas de maneira aleatória dentro do universo de possibilidades. Elas são:

- Topologia 1: Possuí três camadas ocultas, onde a primeira tem 128 neurônios, a segunda tem 64 e a terceira tem 64;
- Topologia 2: Possuí quatro camadas ocultas, onde a primeira tem 64 neurônios, a segunda tem 1024, a terceira tem 128 e a quarta tem 32;
- Topologia 3: Possuí quatro camadas ocultas, onde a primeira tem 128 neurônios, a segunda tem 128, a terceira tem 512 e a quarta tem 64.

Na Tabela 15 é apresentado os resultados obtidos de cada topologia.

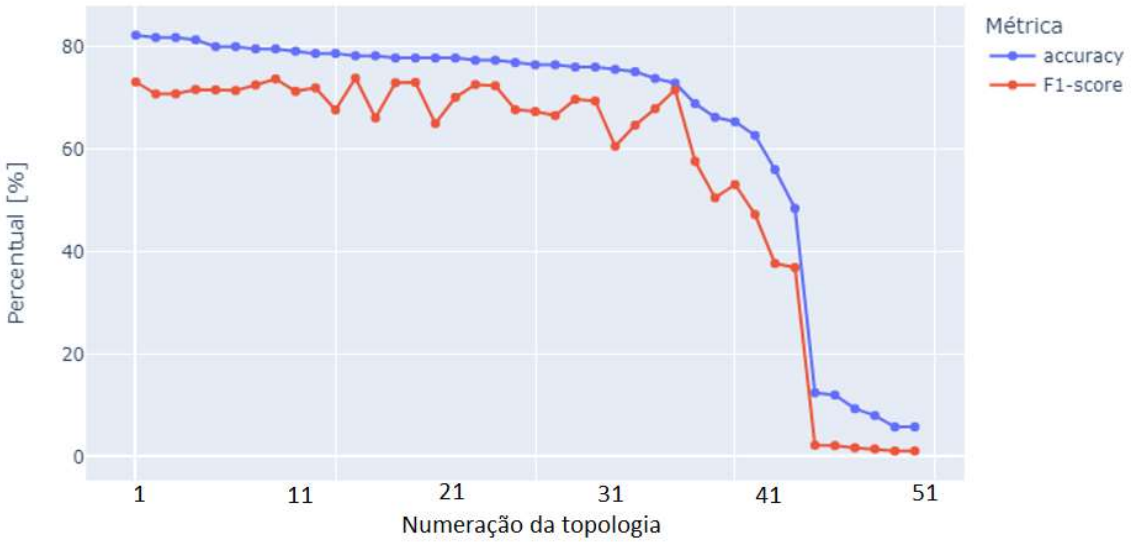
Tabela 15: Resultados das melhores topologias.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	82.22%	73.11%
Topologia 2	81.78%	70.79%
Topologia 3	81.78%	70.79%

Fonte: O autor, 2022.

Entre as topologias analisadas, a topologia 1 apresentou o melhor desempenho em ambas as métricas avaliadas. É interessante notar que a topologia conseguiu obter um resultado melhor, mesmo tendo menos camadas ocultas, em comparação com as topologias 2 e 3. Ainda, na Figura 49, é possível visualizar o desempenho de todas as topologias exploradas.

Figura 49: Desempenho de todas as topologias baseadas em *MLP*.



Fonte: O autor, 2022.

Dentre as topologias explorada, a que produziu os piores resultados utilizou 4 camadas ocultas, onde a primeira possuía 32 neurônios, a segunda 4096, a terceira 512 e a quarta 32.

5.2.2 Comparação dos modelos otimizados sobre os dados de teste

Após a otimização de cada uma das técnicas propostas, podemos avaliar cada uma delas sobre os dados de teste da base de dados aleatorizada o e elencar as melhores técnicas. A Tabela 16 apresenta o resultado da melhor topologia de cada técnica. Nela podemos identificar as três técnicas que obtiveram as melhores métricas como sendo, em primeiro lugar, a técnica baseada na técnica *XGBoost*, seguida da baseada em uma árvore de decisão e, por fim, a baseada em uma floresta aleatória.

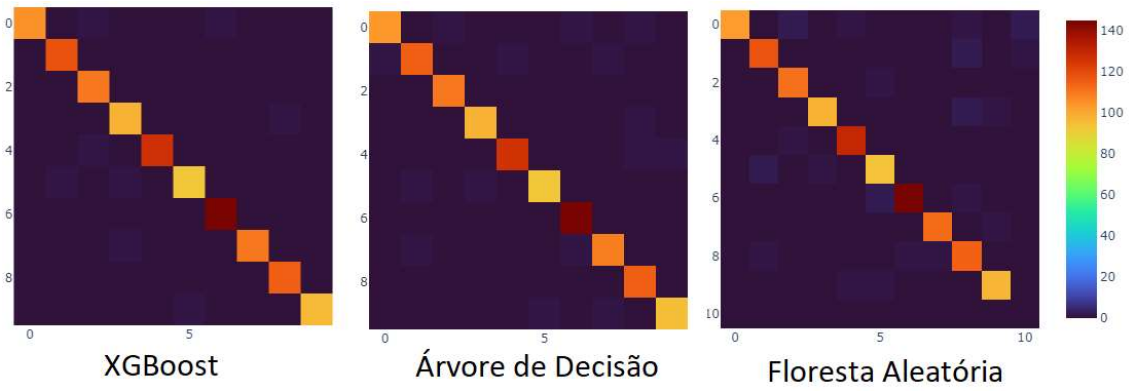
Tabela 16: Resultados das técnicas propostas sobre a base de dados aleatorizada.

Técnica\Métricas	*Taxa de acerto*	*F1-Score*
XGBoost	99,11%	99,15%
Árvore de decisão	99,11%	99,09%
Floresta aleatória	98,67%	98,68%
Regressão Logística	98,67%	98,34%
Máquina de suporte de vetor	98,22%	98,12%
Bayes Ingênuo	97.24%	88.14%
MLP	82.22%	73.11%
Rede neural	81.33%	72.83%

Fonte: O autor, 2022.

Observando a tabela, também, podemos notar que o desempenho dessas três técnicas foi muito semelhante, tanto para a acurácia quanto para métrica *F1-score*. Além disso, podemos entender ainda mais sobre o desempenho das técnicas avaliando as suas matrizes de confusão. A matriz de confusão mostra um mapa de calor representando as classificações que a técnica gerou. O eixo *x* representa as respostas esperadas, ou seja, a real classe dos gestos, e o eixo *y* representa a resposta da técnica. Analisando as matrizes de confusão na Figura 50 podemos perceber que os erros de classificação das técnicas foram, novamente, muito semelhantes.

Figura 50: Matrizes de confusão das três técnicas com melhor desempenho.



Fonte: O autor, 2022.

Dessa forma, a técnica campeão é definida como sendo a baseada na técnica *XGBoost*. Porém, visto que o desempenho delas foi muito semelhante, seria possível seguir com o desenvolvimento do sistema utilizando qualquer uma delas.

5.2.3 Otimização de um modelo baseado em *XGBoost* sobre a base de dados não aleatorizada

Neste processo de otimização foi possível explorar o melhor universo que foi explorado para técnica baseada em *XGBoost* na seção anterior, ou seja, um total de 72 topologias diferentes. Vale lembrar que estes modelos são treinados e avaliadas sobre a base de dados não aleatorizada. As três melhores topologias são:

- Topologia 1: usou 128 estimadores, usou uma taxa de aprendizado de 0,5 e uma política de crescimento “depthwise”;
- Topologia 2: usou 256 estimadores, usou uma taxa de aprendizado de 0,7 e uma política de crescimento “depthwise”;
- Topologia 3: usou 1028 estimadores, usou uma taxa de aprendizado de 0,3 e uma política de crescimento “depthwise”.

Na Tabela 17 é possível observar os resultados obtidas.

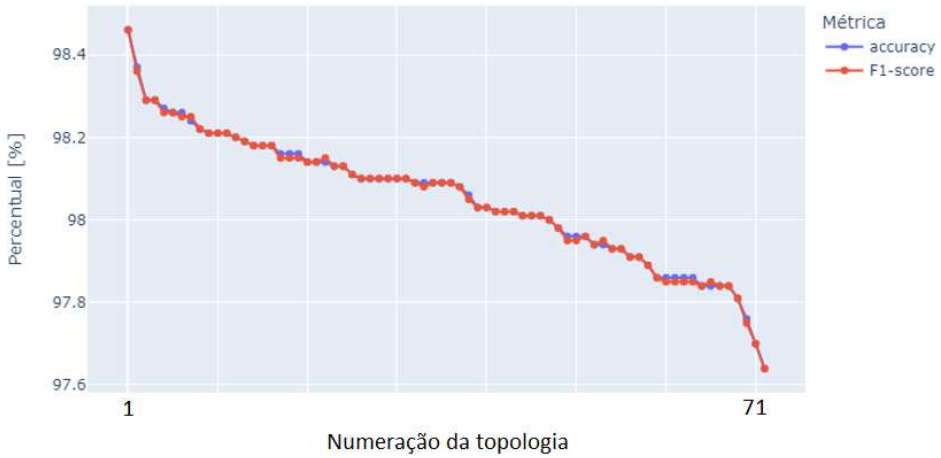
Tabela 17: Resultados das três melhores topologias.

Topologias\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	98,46%	98,46%
Topologia 2	98,37%	98,36%
Topologia 3	98,29%	98,26%

Fonte: O autor, 2022.

Com base na tabela de resultados, podemos perceber que o desempenho entre as três melhores topologias foi muito parecido. Além disso, as métricas de taxa de acerto e *F1-score* foram quase iguais. Assim, a melhor topologia explorada utilizou 128 estimadores, uma política de crescimento “depthwise” e uma taxa de aprendizado de 0,5. Além desses resultados, na Figura 51 podemos avaliar o desempenho do restante das topologias analisadas.

Figura 51: Desempenho de todas as topologias baseadas em XGBoost.



Fonte: O autor, 2022.

Entre todas as topologias exploradas, a que obteve os piores resultados utilizou 512 estimadores, uma taxa de aprendizagem de 0,2 e uma política de crescimento “depthwise”.

5.2.4 Comparação dos modelos treinados sobre a base de dados aleatorizada e não aleatorizada

Por fim, podemos comparar o desempenho das duas topologias baseadas em *XGBoost* sobre os dados de teste da base aleatorizada. Por questão de clareza, nesta seção chamaremos a topologia treinada sobre a base de dados aleatorizada de “topologia 1” e ela é sempre treinada sobre os dados de treinamento da base aleatorizada. A topologia treinada sobre a base de dados não aleatorizada é, então, chamada de “topologia 2” e ela sempre será treinada sobre os dados de treinamento da base de dados não aleatorizada.

Para obter resultados que sejam mais consistentes, cada topologia foi treinada e avaliada trinta vezes de maneira independente, mudando o estado aleatório do algoritmo. As topologias são sempre avaliadas sobre os mesmos dados de teste da base aleatorizada.

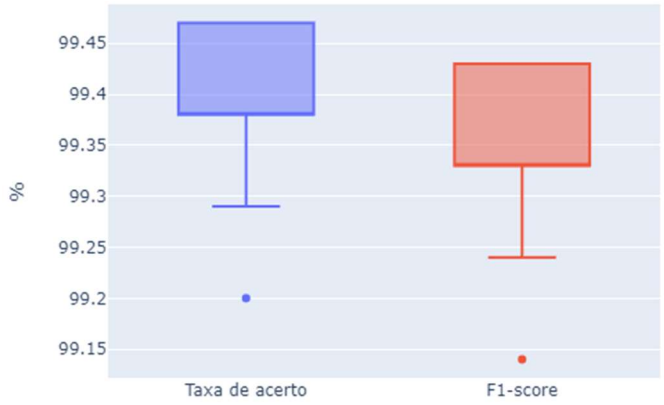
Na Figura 52 podemos observar os gráficos *boxplot* das duas métricas referentes as trinta avaliações da topologia 1 sobre os dados de teste. Para a taxa de acerto, podemos perceber que os valores máximo e mínimo foram de 99,56% e 99,29%, respectivamente. Ademais, o valor médio foi de 99,425% e o valor mediano foi de 99,38%. Já para a métrica *F1-score*, o valor médio foi de 99,375% e o valor mediano foi de 99,42%. Seu valor máximo foi de 99,51% e o mínimo foi de 99,24%.

Figura 52: Gráficos *boxplots* do desempenho referentes as trinta avaliações da topologia 1 sobre os dados de teste.

Fonte: O autor, 2022.

Para a topologia 2, por sua vez, podemos realizar a mesma análise. A Figura 53 apresenta os mesmos gráficos *boxplots* referentes as trinta avaliações da topologia 2 sobre os dados de testes. Para a taxa de acerto podemos observar um valor médio de 99,40% e um valor mediano de 99,38%. Além disso, seus valores máximo e mínimos foram de 99,47% e 99,20%. Já para a métrica *F1-score*, os valores máximos e mínimos foram de 99,43% e 99,14% respectivamente. Seu valor médio foi de 99,355% e seu valor mediano foi de 99,33%.

Figura 53: Gráficos boxplots do desempenho referentes as trinta avaliações da topologia 2 sobre os dados de teste.



Fonte: O autor, 2022.

Finalmente, podemos apresentar os resultados de cada topologia na Tabela 18.

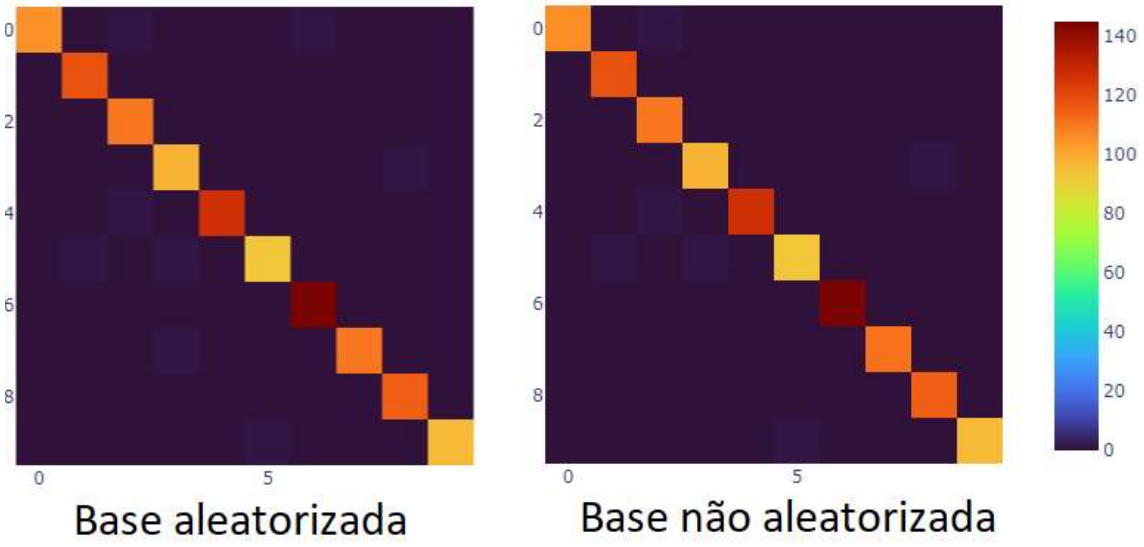
Tabela 18: Resultados das topologias sobre os dados de testes da base aleatorizada.

Topologia\Métricas	*Taxa de acerto*	*F1-Score*
Topologia 1	99,425 ± 0,135%	99,375 ± 0,175%
Topologia 2	99,400 ± 0,069%	99,355 ± 0,075%

Fonte: O autor, 2022.

É possível observar que os resultados obtidos foram muito semelhantes o que indica que a base de dados não aleatorizada é capaz de gerar modelos capazes de generalizar fora de seus dados de treinamento. O que é um ponto muito positivo levando em consideração a facilidade de coleta desta base. Ademais, na Figura 54 podemos comparar as matrizes de confusão dos dois modelos.

Figura 54: Matriz de confusão dos modelos.



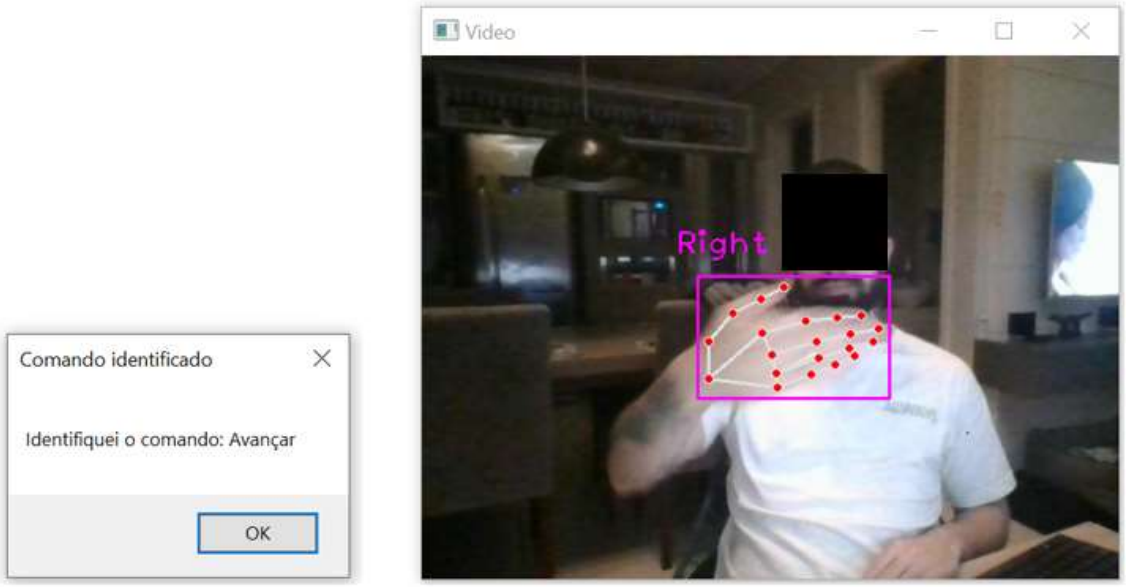
Fonte: O autor, 2022.

Com base nessa matriz de confusão, podemos observar que os dois modelos cometeram, na maior parte, os mesmos erros, confundindo uma classe pela mesma classe.

5.3 DEMONSTRAÇÃO DO SISTEMA COMPLETO

Com o objetivo de demonstrar de maneira simples um sistema que utiliza o reconhecimento de gestos, nesta secção podemos observar figuras que ilustram o uso desse sistema. Na Figura 55 podemos demonstrar o uso de um comando que avançar no navegador ou em uma apresentação de *slides*. Para facilitar a demonstração, simplesmente é apontado na tela qual o comando que foi reconhecido.

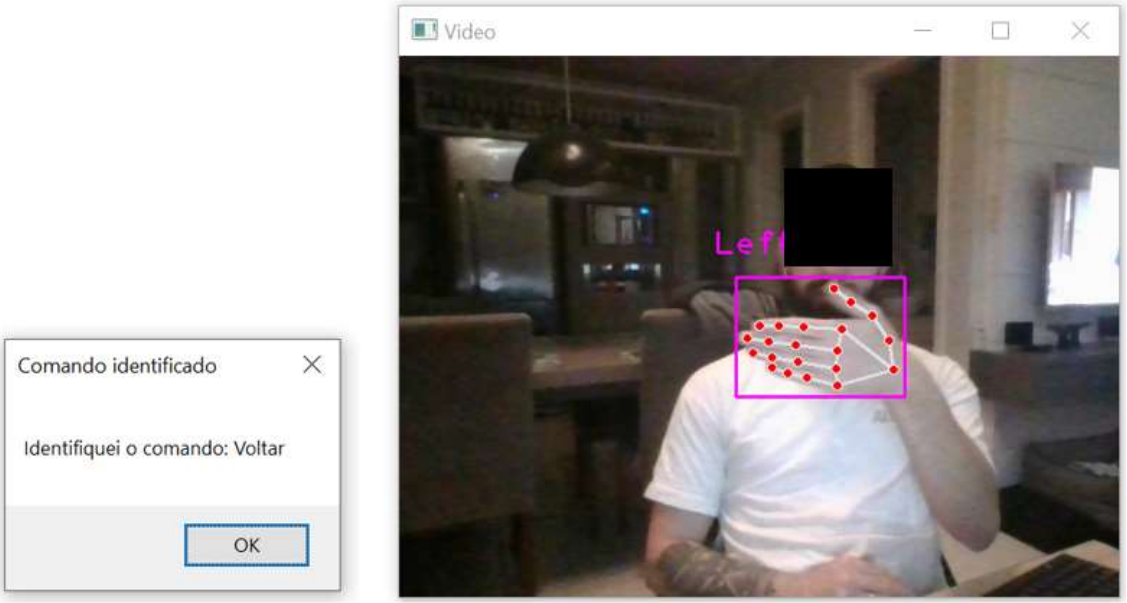
Figura 55: Demonstração do comando "Avançar" do sistema.



Fonte: O autor, 2022.

Na Figura 56 também é possível visualizar um comando, apontando-o na tela. Neste caso, o comando volta para a página anterior no navegador ou para o *slide* anterior na apresentação. Esse comando usa o mesmo gesto que o comando “avançar”, porém ele deve ser executado com a mão esquerda do usuário.

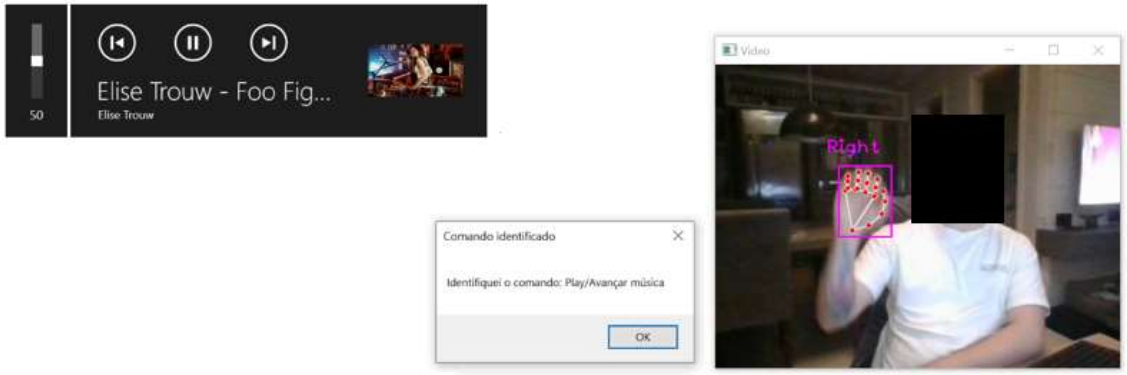
Figura 56: Demonstração do comando "Voltar" do sistema.



Fonte: O autor, 2022.

Por fim, na Figura 57, é possível observar o comando que dá *play* ou *pause* na música que está sendo tocada no computador. A janela escura, no canto superior esquerdo, indicado que o comando foi executado com sucesso.

Figura 57: Demonstração do comando "Play/Pause" do sistema.



Fonte: O autor, 2022.

6 CONCLUSÃO

Com a realização deste trabalho foi possível realizar a captura com sucesso de duas bases de dados com projetos de experimentos distintos. Foi possível, ainda, avaliar quanto a consistência das amostras em suas classes, obtendo uma amostra média para cada distância e comparando-as usando o conceito de diferença geométrica de seus pontos. Assim foi possível observar que a base de dados aleatorizada possuiu a menor diferença média, assim como um menor intervalo de valores. Este resultado se mostra interessante, uma vez que essa base de dados foi composta com o auxílio de 5 voluntários e com um projeto de experimento mais rígido, e a base de dados não aleatorizada, composto de maneira mais simples e apenas com um voluntário, demonstrou ter amostras mais variadas entre si.

Além disso, com os resultados da etapa de treinamento de modelos inteligentes para a classificação de gestos foi possível realizar uma busca extensiva de diferentes técnicas e topologias e identificar que a técnica *XGBoost* resultou no modelo que obteve o melhor desempenho entre todas as outras. No entanto, com a exceção das duas técnicas baseadas em redes neurais, as outras técnicas também tiveram um resultado aceitável na tarefa de classificação, e poderiam ser utilizadas para este fim. Isso demostra que com o auxílio a *API MediaPipe* para extrair o esqueleto da mão numa imagem é possível reduzir bastante a complexidade da tarefa, possibilitando modelos simples a terem um bom desempenho.

Ademais, utilizando a técnica XGBoost para treinar dois modelos, um treinado sobre a base de dados aleatorizada e outro sobre a base de dados não aleatorizada, foi possível obter resultados semelhantes nas duas métricas abordados. Isso implica que ambas as bases de dados são capazes de produzir modelos com uma capacidade classificativa equivalente. Esse resultado demostra que o projeto de experimentos da base de dados não aleatorizada pode ser adotado em trabalhos semelhantes sem comprometer o desempenho dos modelos gerados.

Finalmente, foi possível desenvolver um sistema de interface homem-máquina controlado por gestos com comandos simples para demonstrar a viabilidade desse tipo de aplicação.

6.1 TRABALHOS FUTUROS

Partindo deste trabalho de conclusão de curso, é possível abordar, em trabalhos futuros, alguns aspectos interessantes. Primeiramente, é possível utilizar as bases de dados já constituídas para trabalhos com os mais diversos objetivos. Além disso, é possível levar em consideração os resultados obtidos a partir das duas bases de dados e adotar apenas a metodologia para gerar bases de dados segundo o projeto da base de dados não aleatorizada, uma vez que obteve resultados semelhantes.

Ainda, é possível explorar diferentes técnicas para classificar gestos, como técnicas baseadas em redes neurais convolucionais ou baseadas em grafos. Por fim, é possível desenvolver sistema mais robustos e práticos para os mais diversos fins que utilizem o tipo de interface homem-máquina aqui abordado.

7 REFERÊNCIAS BIBLIOGRÁFICAS

Lopatovska, I., Rink, K., Knight, I., Raines, K., Cosenza, K., Williams, H., Martinez, A. (2019). Talk to me: Exploring user interactions with the Amazon Alexa. Journal of Librarianship and Information Science, 51(4), 984–997. https://doi.org/10.1177/0961000618759414

W. Qi, S. E. Ovur, Z. Li, A. Marzullo and R. Song, "Multi-Sensor Guided Hand Gesture Recognition for a Teleoperated Robot Using a Recurrent Neural Network," in IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 6039-6045, July 2021, doi: 10.1109/LRA.2021.3089999.

N. Zengeler, M. Grimm, C. Borgmann, M. Jansen, S. Eimler and U. Handmann, "An Evaluation of Human Detection Methods on Camera Images in Heavy Industry Environments," 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2019, pp. 205-210, doi: 10.1109/ICIEA.2019.8834060.

Sagayam, K.M., Hemanth, D.J. Hand posture and gesture recognition techniques for virtual reality applications: a survey. Virtual Reality 21, 91–107 (2017). https://doi.org/10.1007/s10055-016-0301-0

DIAS, Thiago Simões. LUVA INSTRUMENTADA PARA RECONHECIMENTO DE PADRÕES DE GESTOS EM LIBRAS. 95 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2020.

Mujahid, A.; Awan, M.J.; Yasin, A.; Mohammed, M.A.; Damaševičius, R.; Maskeliūnas, R.; Abdulkareem, K.H. Real-Time Hand Gesture Recognition Based on Deep Learning YOLOv3 Model. Appl. Sci. 2021, 11, 4164. https://doi.org/10.3390/app11094164

H. -Y. Chung, Y. -L. Chung and W. -F. Tsai, "An Efficient Hand Gesture Recognition System Based on Deep CNN," 2019 IEEE International Conference on Industrial Technology (ICIT), 2019, pp. 853-858, doi: 10.1109/ICIT.2019.8755038.

V. Bhame, R. Sreemathy and H. Dhumal, "Vision based hand gesture recognition using eccentric approach for human computer interaction," 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014, pp. 949-953, doi: 10.1109/ICACCI.2014.6968545.

Sridhar, S., Mueller, F., Oulasvirta, A., Theobatl, C. “Fast and Robust Hand Tracking Using Detection-Guided Optimization”.Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on , vol., no., pp.3213-3221, 7-12 June 2015. https://doi.org/10.48550/arXiv.1602.04124

Quentin de Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Joris Guerry, Bertrand Le Saux, et al.. SHREC’17 Track: 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset. 3DOR - 10th Eurographics Workshop on 3D Object Retrieval, Apr 2017, Lyon, France. pp.1-6, ff10.2312/3dor.20171049ff. ffhal-01563505f

Zhang, Fan & Bazarevsky, Valentin & Vakunov, Andrey & Tkachenka, Andrei & Sung, George & Chang, Chuo-Ling & Grundmann, Matthias. (2020). MediaPipe Hands: On-device Real-time Hand Tracking. https://doi.org/10.48550/arXiv.2006.10214

Sung, George & Sokal, Kanstantsin & Uboweja, Esha & Bazarevsky, Valentin & Baccash, Jonathan & Bazavan, Eduard & Chang, Chuo-Ling & Grundmann, Matthias. (2021). On-device Real-time Hand Gesture Recognition. https://doi.org/10.48550/arXiv.2111.00038

Islam, Rafid & Rahman, Ratun & Ahmed, Akib & Jany, Rafsan. (2022). NFS: A Hand Gesture Recognition Based Game Using MediaPipe and PyGame. 10.48550/arXiv.2204.11119. https://doi.org/10.48550/arXiv.2204.11119

G. Devineau, F. Moutarde, W. Xi and J. Yang, "Deep Learning for Hand Gesture Recognition on Skeletal Data," 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018, pp. 106-113, doi: 10.1109/FG.2018.00025.

Liu, Jianbo, Ying Wang, Shiming Xiang and Chunhong Pan. “HAN: An Efficient Hierarchical Self-Attention Network for Skeleton-Based Gesture Recognition.” ArXiv abs/2106.13391 (2021): n. pag. https://doi.org/10.48550/arXiv.2106.13391

K. Faceli, A. C. Lorena, J. Gama, A. C. P. d. L. Carvalho, et al. Inteligência artificial: Uma abordagem de aprendizado de máquina. 2011

Russell and P. Norvig. Artificial Intelligence: A Modern Approach. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

EL SANHARAWI, M.; NAUDET, F. Understanding logistic regression. Journal français d’ophtalmologie, v. 36, n. 8, p. 710–5, 2013. https://doi.org/10.1016/j.jfo.2013.05.008

BRITO, Edeleon Marcelo Nunes. Mineração de Textos: Detecção automática de sentimentos em comentários nas mídias sociais. Belo Horizonte, 2016 Dissertação (Sistemas de Informação e Gestão do Conhecimento) - UNIVERSIDADE FUMEC, 2016.

ALTERYX INC. A ferramenta Classificador Naive Bayes. Alteryx. 2018.

LORENA, Ana Carolina e CARVALHO, André Carlos Ponce de Leon Ferreira. Introdução às máquinas de vetores suporte (support vector machines). São Carlos: ICMC-USP. Disponível em: https://repositorio.usp.br/directbitstream/a7ed198b-f6a3-4cec-b132-7e113bd51424/BIBLIOTECA_113_RT_192.pdf. Acesso em: 02 out. 2022.

AMARAL, F. Aprenda Mineração de Dados: Teoria e prática. São Paulo: Alta Books, 2016

BREIMAN, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. ArXiv:1603.02754v3, [s. l.], 2016. https://doi.org/10.48550/arXiv.1603.02754.

S. Haykin, Neural Networks: a comprehensive foundation. New York: MacMillan College Publishing Co., 1994.

Ferneda, E. (2006). Redes neurais e sua aplicação em sistemas de recuperação de informação. Ciência Da Informação, 35(1). https://doi.org/10.18225/ci.inf.v35i1.1149

BRAGA, A. de.; LUDERMIR, T. B.; CARVALHO, A. C. P. de L. F. Redes Neurais Artificiais teoria e aplicações, Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora S.A., 2000.

HAYKIN, S. Redes Neurais- Princípios e Práticas. BOOKMAN, São Paulo, 2ª ed. 2001. 900 p

MACHADO, W, C.; FONSECA JÚNIOR, E. S. Redes Neurais Artificiais aplicadas na previsão do VTEC no Brasil. Boletim de Ciências Geodesicas, v.19, n.2, p. 227-246, 2013

SHARMA, V.; RAI, S.; DEV, A. A comprehensive study of artificial neural networks. International Journal of Advanced Research in Computer Science and Software Engineering, v. 2, n. 10, p. 278–283, out. 2012. ISSN 2277 128X.

Andrea Vedaldi and Karel Lenc. 2015. MatConvNet: Convolutional Neural Networks for MATLAB. In Proceedings of the 23rd ACM international conference on Multimedia (MM '15). Association for Computing Machinery, New York, NY, USA, 689–692. https://doi.org/10.1145/2733373.2807412

Rauber, T.W.; Redes Neurais Artificiais; UFES 2016

Nied, A.; TREINAMENTO DE REDES NEURAIS ARTIFICIAIS BASEADO EM SISTEMAS DE ESTRUTURA VARIAVEL COM TAXA DE APRENDIZADO ADAPTATIVA; UFMG, 2007

PROBST, P.; BISCHL, B.; BOULESTEIX, A.-L. Tunability: Importance of hyperparameters of machine learning algorithms. arXiv preprint arXiv:1802.09596, 2018

Nguyen, Xuan & Brun, Luc & Lezoray, Olivier & Bougleux, Sébastien. (2019). A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. https://doi.org/10.48550/arXiv.1904.12970

Benitez-Garcia, Gibran & Prudente-Tixteco, Lidia & Castro-Madrid, Luis & Toscano-Medina, Rocio & Olivares Mercado, Jesus & Sanchez-Perez, Gabriel & García Villalba, Luis. (2021). Improving Real-Time Hand Gesture Recognition with Semantic Segmentation. Sensors. 21. 356. https://doi.org/10.3390/s21020356

Singh, Dalwinder & Singh, Birmohan. (2019). Investigating the impact of data normalization on classification performance. Applied Soft Computing. 105524. 10.1016/j.asoc.2019.105524.

L. Guo, Z. Lu and L. Yao, "Human-Machine Interaction Sensing Technology Based on Hand Gesture Recognition: A Review," in *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 300-309, Aug. 2021, doi: 10.1109/THMS.2021.3086003.

G. De Pasquale, S. -G. Kim and D. De Pasquale, "GoldFinger: Wireless Human–Machine Interface With Dedicated Software and Biomechanical Energy Harvesting System," in *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 1, pp. 565-575, Feb. 2016, doi: 10.1109/TMECH.2015.2431727.

Ramaswamy, A., Monsuez, B. and Tapus, A. (2014). Model driven software development for human-machine interaction systems, Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction, pp. 270–271