# Análise da aplicação de visão computacional e aprendizado de máquina na tradução da Libras para texto

Letícia M. G. de Morais*, Ana Kelly N. Costa*, Welizangela M. Almeida†,
Náthalee C. de A. Lima‡, Rosana C. B. Rego‡
*Computational Intelligence Laboratory, Universidade Federal Rural do Semi-Árido, Pau dos Ferros, Brasil
†Universidade Estadual do Rio Grande do Norte, Mossoró, Brasil
‡Departamento de Engenharias e Tecnologia, Universidade Federal Rural do Semi-Árido, Pau dos Ferros, Brasil
leticia.morais30463@alunos.ufersa.edu.br, ana.costa03379@alunos.ufersa.edu.br, welizangela_almeida@outlook.com
nathalee.almeida@ufersa.edu.br, rosana.rego@ufersa.edu.br

***Resumo*—A Língua Brasileira de Sinais (Libras) é um sistema linguístico completo utilizado predominantemente por pessoas surdas no Brasil. Apesar de sua importância, a comunicação entre surdos e ouvintes ainda enfrenta dificuldades significativas, devido à baixa fluência da população em geral e à limitada disponibilidade de intérpretes. A fim de reduzir essa barreira, este trabalho propõe uma abordagem de visão computacional e aprendizado de máquina para o reconhecimento automático de sinais em Libras. Foi utilizado um conjunto de dados obtido a partir da fusão de três bases distintas, totalizando 44 classes. O *framework MediaPipe* foi utilizado para extrair coordenadas tridimensionais dos corpos e mãos dos sinalizadores e essas informações foram utilizadas para treinar modelos de aprendizado de máquina. Os resultados experimentais demonstraram que o classificador *Random Forest* foi o modelo que obteve a melhor performance com relação ao demais, com acurácia, precisão, *recall* e *F1-score* de 99,56%.**

***Palavras-chave* —Inteligência Artificial, Visão computacional, Aprendizado de máquina, Libras, Tradução.**

## I. Introdução

A Língua Brasileira de Sinais (Libras) é uma língua visual-espacial natural utilizada pela comunidade surda brasileira [1]. Assim como outras línguas de sinais ao redor do mundo, a Libras possui uma estrutura linguística completa, com níveis fonológicos, morfológicos, sintáticos e semânticos próprios [2], [3]. É composta por gestos manuais combinados com expressões faciais e corporais, permitindo a comunicação de ideias, sentimentos e conceitos de forma fluida [4].

A inclusão social de pessoas surdas passa, necessariamente, pela superação das barreiras de comunicação. Entretanto, a falta de fluência em Libras por parte de grande parcela da população e a escassez de intérpretes são entraves para essa inclusão [5]. Nesse contexto, ferramentas digitais como o VLibras [1] e o SpreadTheSign [6] desempenham um papel relevante ao facilitar o acesso ao vocabulário em Libras, promovendo o aprendizado autônomo e o contato com diferentes línguas de sinais, contribuindo para a difusão da cultura surda e o fortalecimento da acessibilidade comunicacional. Além disso, o uso de tecnologias baseadas em Inteligência Artificial (IA) vem se destacando como uma alternativa promissora, ao permitir o desenvolvimento de ferramentas de tradução automática entre a Libras e o português oral/escrito.

Nos últimos anos, avanços na visão computacional e no aprendizado de máquina impulsionaram a criação de sistemas capazes de reconhecer sinais por meio de imagens e vídeos. Redes Neurais Convolucionais (CNNs), Redes Neurais Recorrentes (RNNs) e técnicas de aprendizado supervisionado como *Support Vector Machines* (SVMs) têm sido amplamente utilizadas para essa finalidade [7]–[9]. Ainda assim, muitos desafios persistem, como a escassez de bases de dados públicas em Libras, a variabilidade dos sinais entre usuários e as limitações impostas por fatores ambientais, como iluminação e ângulo de captura [10], [11].

Neste contexto, o presente artigo propõe uma abordagem baseada na extração de coordenadas espaciais 3D dos movimentos das mãos e do corpo, utilizando o *framework MediaPipe* da Google. Essas coordenadas são posteriormente utilizadas como atributos de entrada em um modelo de classificação supervisionado, visando o reconhecimento automático de sinais da Libras. A proposta busca validar a viabilidade de se utilizar dados cinemáticos — em vez de imagens ou vídeos brutos — para treinar algoritmos eficientes, precisos e com baixo custo computacional.

A relevância desta pesquisa está na potencial contribuição para a criação de ferramentas acessíveis que favoreçam a comunicação entre surdos e ouvintes, além de enriquecer o campo de estudo do reconhecimento gestual por IA com uma abordagem de classificação alternativa que faz o uso de coordenadas espaciais para reconhecimento dos sinais. Ademais, este trabalho responde diretamente à lacuna evidenciada por [12] em uma revisão sistemática da literatura, a qual identificou uma carência significativa de aplicações com foco na Libras — visto que a maioria dos estudos analisados prioriza línguas de sinais estrangeiras. A exploração dessa lacuna, portanto, também se configura como uma das principais contribuições do presente estudo.

Como contribuição prática, este artigo propõe um *pipeline* de reconhecimento baseado na extração de coordenadas

tridimensionais por meio do *MediaPipe* — técnica ainda pouco explorada nesse contexto — e valida sua eficácia por meio da fusão de três bases públicas, abrangendo 44 sinais distintos. Também são comparados diversos algoritmos de classificação supervisionada, com destaque para o *Random Forest*, que apresentou os melhores resultados. Por fim, o trabalho contribui ao disponibilizar um conjunto de dados pré-processado contendo *landmarks* das mãos e do corpo, visando facilitar estudos futuros na área.

Este artigo está organizado da seguinte forma: na Seção 2, é apresentada a fundamentação teórica com base em uma revisão sistemática da literatura sobre reconhecimento gestual por meio da inteligência artificial. A Seção 3 descreve a metodologia empregada, incluindo a construção do conjunto de dados e o algoritmo proposto. A Seção 4 apresenta e discute os resultados obtidos a partir da avaliação do modelo treinado. Por fim, a Seção 5 apresenta as conclusões e as sugestões para trabalhos futuros.

## II. Fundamentação teórica

A aplicação da inteligência artificial no reconhecimento de sinais em línguas de sinais tem se intensificado nos últimos anos, motivada pela necessidade de automatizar a tradução gestual e promover a inclusão de pessoas surdas em ambientes digitais e sociais. Uma revisão sistemática da literatura (RSL), apresentada em [12], identificou e analisou 93 estudos publicados entre 2019 e 2024, com foco na utilização de técnicas de IA para o reconhecimento gestual da Libras e de outras línguas de sinais.

Entre as técnicas de IA mais recorrentes, destacam-se as CNNs, com 35% de uso, seguidas pelas redes LSTM (22%) e pelas SVMs, com 15%. Essas técnicas têm se mostrado particularmente eficientes no processamento de dados visuais e temporais, aspectos essenciais na identificação precisa dos sinais.

O estudo também evidenciou o uso de diferentes tipos de atributos de entrada nos modelos. As imagens estáticas representaram a maior parte (30,2%), seguidas por dados RGB ou segmentações em escala de cinza (16,8%), vídeos dinâmicos (12,6%), dados de sensores de movimento (9,5%) e atributos extraídos de expressões faciais e articulações corporais por ferramentas como o *MediaPipe* (8,7%). Além disso, 7,9% dos trabalhos analisaram sinais temporais, e 14,3% se enquadraram em abordagens híbridas ou variadas.

As métricas de avaliação mais comuns incluíram acurácia, precisão, *recall* e *F1-score*, refletindo uma preocupação recorrente com o desempenho geral dos modelos. No entanto, diversos desafios foram identificados: sensibilidade à iluminação, baixa qualidade ou ausência de bases de dados públicas voltadas à Libras, dificuldade de distinguir sinais com gestos semelhantes e alto custo computacional de alguns modelos — especialmente em aplicações que demandam respostas em tempo real.

Adicionalmente, a RSL revelou uma lacuna significativa: a maioria dos estudos analisados concentra-se em línguas de sinais estrangeiras, como ASL (*American Sign Language*) e ArSL (*Arabic Sign Language*), havendo uma carência expressiva de pesquisas voltadas especificamente para a Libras. Essa lacuna reforça a importância de iniciativas que explorem soluções baseadas em IA para o reconhecimento de sinais da Libras, utilizando recursos acessíveis e replicáveis — como é o caso do presente estudo, que opta pelo uso de coordenadas espaciais extraídas via *MediaPipe* para compor a base de treinamento dos modelos.

## III. Metodologia

Foi adotada uma abordagem experimental com análises quantitativas e descritivas, em que o objetivo foi desenvolver e avaliar um algoritmo de aprendizado de máquina para reconhecer sinais da Libras a partir de coordenadas espaciais do corpo e das mãos. Dessa forma, a pesquisa foi dividida nas duas etapas a seguir.

### A. Construção do conjunto de dados

Para construir o conjunto de dados, foram utilizados três *datasets* públicos: *Brazilian Sign Language - Words Recognition* [13], *Libras-10* [14] e *MINDS-Libras* [15]. O primeiro *dataset* foi extraído da plataforma *Kaggle*, enquanto os dois últimos foram extraídos da *Zenodo*.

A Tabela I apresenta a quantidade de sinais e vídeos que cada *dataset* utilizado possui. Entretanto, os *datasets Brazilian Sign Language - Words Recognition* [13] e *MINDS-Libras* [15] possuem um sinal em comum: o sinal de "Banco". Dessa forma, o conjunto de dados final possui 44 sinais.

Tabela I: Informações dos *datasets* utilizados

| ***Dataset*** | **Sinais** | **Vídeos** |
|---|---|---|
| *Brazilian Sign Language - Words Recognition* [13] | 15 | 140 |
| *Libras-10* [14] | 10 | 100 |
| *MINDS-Libras* [15] | 20 | 1.200 |

*1) Extração dos atributos:* Para formular um sinal na Libras, são considerados cinco parâmetros fundamentais: configuração de mãos, ponto de articulação (local do corpo ou espaço), movimento, orientação das palmas das mãos e expressão facial e corporal [16]. Dessa forma, para que esses parâmetros fossem considerados, foram utilizados os componentes *Hands* e *Pose*, que empregam modelos de aprendizado de máquina internamente para realizar tarefas como detecção e rastreamento. Ambos fazem parte do módulo *Holistic* do *framework MediaPipe*, uma solução desenvolvida pela Google que combina modelos baseados em CNNs otimizadas para detectar e rastrear, em tempo real, poses corporais, mãos e face. O *MediaPipe Holistic* realiza a detecção inicial por meio de segmentação baseada em CNNs e, em seguida, aplica regressão para estimar a posição tridimensional dos pontos-chave (*landmarks*). Esses *landmarks*, por sua vez, são utilizados como entrada para classificadores tradicionais ou outras etapas do *pipeline* de reconhecimento.

O componente *Hands* é responsável por detectar, classificar e extrair as coordenadas espaciais (3D) das mãos presentes nos *frames* de cada vídeo. Quando uma mão é detectada,

é realizada a classificação da sua lateralidade (esquerda ou direita) e, depois, é realizada a extração das coordenadas 3D dos 21 dos pontos de referência (*landmarks*) daquela mão. A Figura 1 apresenta os *landmarks* que são extraídos por este modelo.

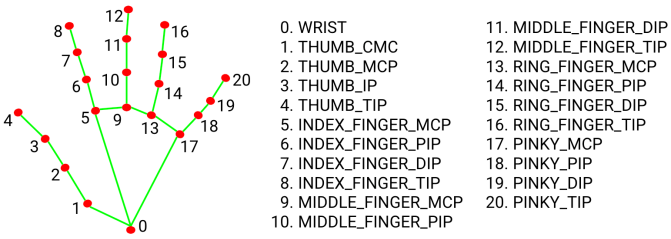



Figura 1: *Landmarks* do modelo *Hands* [17].

Já o modelo *Pose* detecta e extrai as coordenadas 3D dos *landmarks* do corpo presentes nos *frames* dos vídeos. Diferente do modelo *Hands*, o modelo *Pose* extrai 33 *landmarks* do corpo, como exibido na Figura 2.

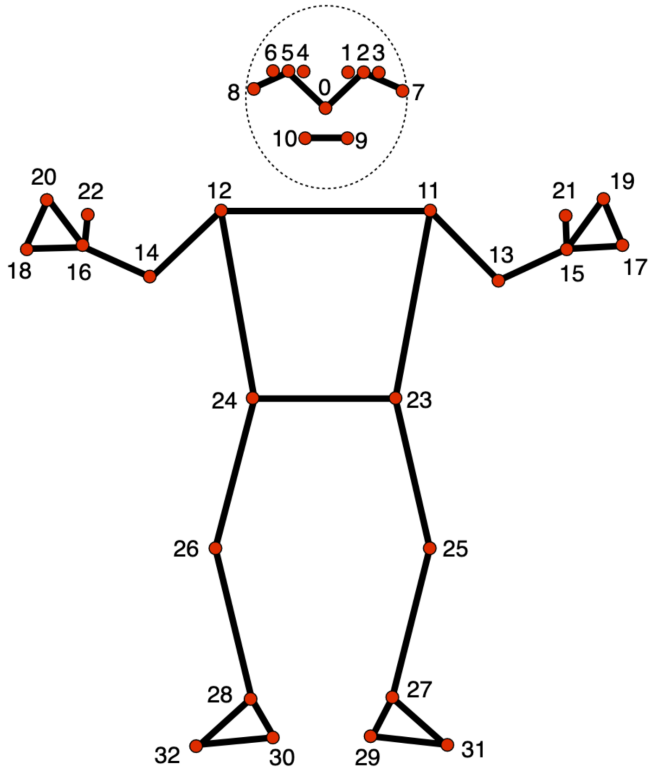


Figura 2: *Landmarks* do modelo *Pose* [17].

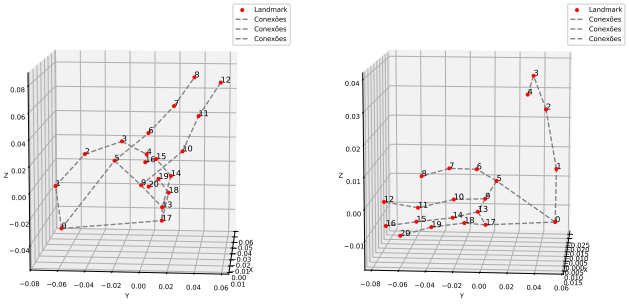
Dessa forma, utilizando os modelos *Hands* e *Pose*, foi possível criar um algoritmo para extrair as coordenadas espaciais das mãos e do corpo a partir dos *frames* de cada vídeo dos três *datasets* anteriormente mencionados. Esse algoritmo armazena as coordenadas X, Y e Z de todos os *landmarks* das mãos e dos 17 primeiros *landmarks* do corpo. Além disso, para contemplar tanto sinalizadores canhotos quanto destros, os *frames* são processados duas vezes, sendo uma delas com espelhamento horizontal.

Adicionalmente, devido à variedade de resoluções presentes nos vídeos dos diferentes *datasets*, foi adotada uma estratégia de normalização relativa das coordenadas, utilizando como referência o centro do corpo (região do tronco). Considerando que os sinais da Libras são predominantemente realizados da cintura para cima, essa centralização permite capturar de forma mais efetiva os movimentos relevantes, mantendo a consistência espacial entre os *frames*, independentemente da resolução original dos vídeos. Para isso, utilizou-se a função *world_landmark* do modelo *Pose* do *MediaPipe*, que fornece coordenadas tridimensionais $(X, Y, Z)$ em metros, estimadas por uma arquitetura do tipo *BlazePose* treinado com dados em 3D [18]. Esse sistema é centrado no centro do quadril, assumindo esse ponto como origem $(0, 0, 0)$ do sistema de coordenadas, sendo que valores negativos de Z indicam posições atrás da câmera.

A Figura 3 mostra um exemplo de como os modelos *Hands* e *Pose* detectam e exibem os *landmarks* das mãos e do corpo em um *frame* do vídeo do sinal "Acontecer", do *dataset MINDS-Libras* [15].

Figura 3: Exemplo de *landmarks* com o sinal "Acontecer".

A Figura 4 apresenta as coordenadas espaciais extraídas dos *landmarks* das mãos do sinal "Acontecer" do exemplo da Figura 3.



(a) (b)

Figura 4: Coordenadas espaciais dos *landmarks* das mãos da Figura 3: (a) Coordenadas espaciais da mão esquerda e (b) Coordenadas espaciais da mão direita.

O uso das coordenadas espaciais das mãos permite considerar dois dos parâmetros fundamentais na construção dos sinais em Libras: a configuração das mãos e a orientação das palmas. Já na Figura 5, são apresentadas as coordenadas espaciais extraídas dos *landmarks* detectados pelo modelo *Pose*, conforme ilustrado na Figura 3. As coordenadas do corpo possibilitam considerar as expressões faciais e o local do corpo onde o sinal é realizado. Vale ressaltar que o algoritmo de extração apresenta comportamentos distintos dependendo da quantidade de mãos detectadas em um determinado *frame*. Quando duas mãos são detectadas com lateralidades diferentes, o processo ocorre normalmente, extraindo as coordenadas de todos os *landmarks*. Se apenas uma mão for detectada, realiza-se a classificação de sua lateralidade, extraem-se suas coordenadas, e são atribuídos zeros às coordenadas da mão ausente. Dessa forma, os únicos casos em que o *frame* não é processado ocorrem quando nenhuma mão é detectada ou quando duas mãos são classificadas com a mesma lateralidade.

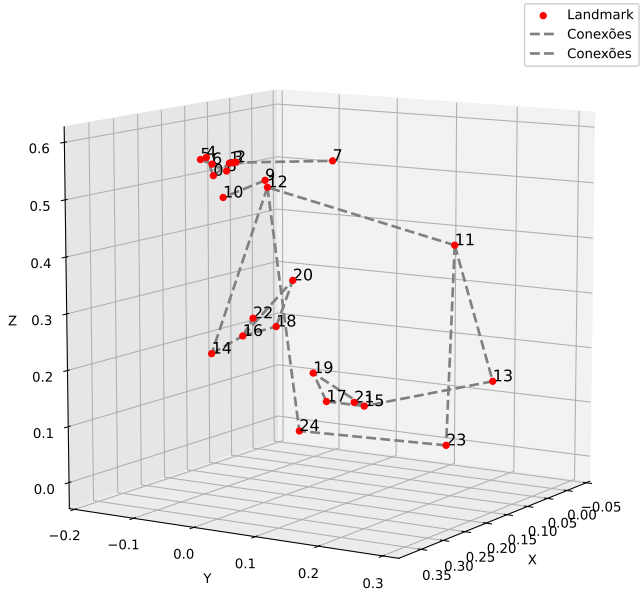



Figura 5: Coordenadas espaciais dos *landmarks* do corpo da Figura 3.

*2) Divisão e balanceamento de dados:* Depois que as coordenadas espaciais dos 21 *landmarks* das duas mãos e dos 17 primeiros *landmarks* do corpo foram extraídas, esses atributos foram armazenados em um *DataFrame* da biblioteca *Pandas*. Dessa forma, o *DataFrame* se constitui de 63 colunas destinadas às coordenadas da mão esquerda, outras 63 colunas para as da mão direita, 51 colunas para as coordenadas do corpo e uma última que corresponde a qual sinal aqueles atributos pertencem. Portanto, o *DataFrame* com os atributos possui 178 colunas e 69.936 linhas, onde as linhas representam o total de *frames* dos três *datasets* de onde os atributos foram extraídos.

Ao fim da construção do conjunto de dados, os dados do *DataFrame* foram divididos em 80% para treino e 20% para teste. Entretanto, devido ao uso de três *datasets* diferentes para a construção do conjunto de dados final, houve um desbalanceamento entre a quantidade de classes (sinais).

A Figura 6 apresenta a distribuição de cada classe presente no conjunto de dados de treino. Nota-se que a classe do sinal "Formação" é a que possui a maior quantidade de dados, enquanto a classe "Seu" apresenta uma quantidade menor.

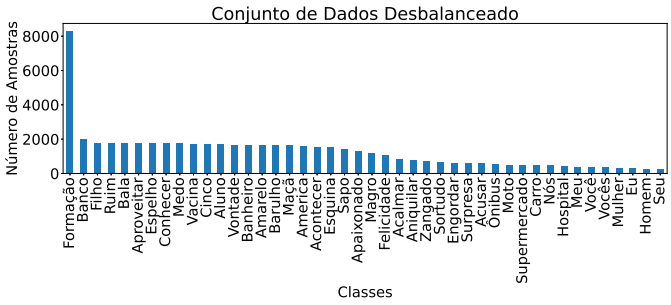



Figura 6: Conjunto de treino desbalanceado.

Assim, foi utilizada a técnica SMOTE (*Synthetic Minority Oversampling Technique*) [19] para que fosse possível balancear as classes do conjunto de dados de treino. O SMOTE gera amostras sintéticas para as classes que possuem um menor quantitativo de dados, fazendo com que a distribuição dos dados seja equilibrada em relação à classe majoritária [20].

A Figura 7 mostra o estado do conjunto de dados de treino após a aplicação da técnica, onde foram gerados dados sintéticos para as classes com um menor número de dados, deixando o conjunto de dados balanceado.

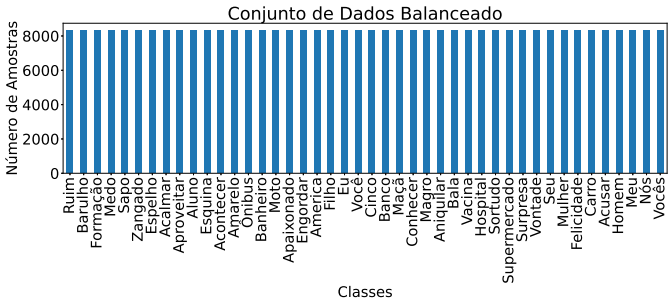



Figura 7: Conjunto de treino balanceado.

## B. Algoritmo proposto

O *framework Pycaret* foi utilizado para encontrar o modelo de aprendizado de máquina com o maior desempenho para o conjunto de dados. Dentre os modelos implementados, o classificador *Random Forest* foi o que teve o melhor desempenho. O *Random Forest* é um *ensemble* de árvores de decisão que utiliza o método *bagging* (*bootstrap aggregating*) para gerar árvores diferentes, aumentando a precisão e reduzindo o sobreajuste [7], [21].

No *Random Forest*, cada árvore $T_i$ é treinada com um subconjunto $D_i$ do conjunto de dados $D$, obtida por *bootstrap*

*sampling*. Ou seja, cada amostra $D_i$ é gerada aleatoriamente com reposição e pode existir dados duplicados [7].

$$D_i = \{(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \ldots, (x_{in}, y_{in})\}. \quad (1)$$

Como são treinadas sobre subconjuntos diferentes, as árvores de decisão do *Random Forest* resultam em modelos distintos [22], podendo gerar predições variadas. Assim, a predição final $\hat{y}$ do modelo é obtida pelo voto majoritário das predições individuais das $k$ árvores $T_i$:

$$\hat{y} = \text{mode}(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_k), \quad (2)$$

onde *mode* representa a classe mais frequente entre as predições individuais.

Para o modelo proposto, foram utilizadas 150 árvores de decisão. A pureza dos nós foi qualificada por meio da entropia, utilizada como critério de divisão. O parâmetro de profundidade máxima das árvores foi definido como *None*, permitindo que as árvores cresçam sem qualquer limitação pré-estabelecida. O número mínimo de amostras exigido para que um nó seja considerado folha foi fixado em 1. Já o número mínimo de amostras requerido para que um nó seja dividido foi configurado como 2. Por fim, o número máximo de características avaliadas em cada divisão foi definido como $\log_2$, o que significa que o parâmetro *max_features* corresponde ao logaritmo na base 2 do total de características do conjunto de treinamento (177).

## IV. Resultados e Discussões

Para treinar os modelos de classificação, foram utilizados os *frameworks Pycaret* (versão 3.3.2) e *Scikit-learn* (versão 1.4.2), juntamente com a linguagem *Python* (versão 3.11). Os modelos foram treinados e testados em um ambiente computacional composto por uma CPU QEMU Virtual version 2.5+ (14) @ 3.312GHz, uma GPU NVIDIA GeForce GTX 1660 Ti, 30 Gi de memória RAM e o sistema operacional Ubuntu 22.04.3 LTS x86_64.

Os modelos foram avaliados com base nas suas métricas de desempenho no conjunto de dados de teste. As métricas foram: acurácia, AUC, *recall*, precisão e *F1 Score*, em que maiores valores indicam um melhor desempenho do modelo em suas predições. Além disso, também foi avaliado o tempo de inferência médio (em segundos) para as mesmas 100 amostras, a fim de medir a velocidade de predição dos modelos.

A Tabela II exibe os resultados das métricas dos modelos e o tempo de treinamento para o conjunto de dados de 44 classes. Os modelos avaliados foram RF (*Random Forest*) , KNN (*K-Nearest Neighbors*) [23], DT (*Decision Tree*), QDA (*Quadratic Discriminant Analysis*), LDA (*Linear Discriminant Analysis*) [24], LR (*Logistic Regression*), Ridge (*Ridge Classifier*) [25], LightGBM (*Light Gradient Boosting Machine*) [26], SVM (*Support Vector Machine*) [27], NB (*Naive Bayes*) [28], AdaBoost (*Adaptive Boosting*) [29] e DC (Dummy Classifier). Para o KNN, 5 vizinhos com pesos uniformes e distância Euclidiana foram usados. Para a DT, o critério Gini com melhor divisão e sem limite de profundidade foi adotado. Para o QDA, utilizou-se regularização zero e covariância específica por classe. Para o LDA, o solver padrão "*svd*" foi aplicado. O LR empregou penalização L2, solver "*lbfgs*" e até 100 iterações. O RC usou regularização *alpha* igual a 1. Para o LightGBM, foram configurados 31 folhas, profundidade ilimitada e taxa de aprendizado 0,1. O SVM usou kernel RBF com penalização C igual a 1. O NB calculou média e variância diretamente dos dados. Por fim, o AdaBoost utilizou 50 estimadores baseados em árvores rasas com taxa de aprendizado 1.0.

Entre os modelos avaliados, o RF foi o que apresentou os melhores resultados em suas métricas de desempenho, confirmando que o modelo generaliza bem, isto é, o modelo classifica corretamente quase todas as amostras do conjunto de teste balanceado. Apesar de o modelo KNN ter alcançado métricas próximas as do modelo RF, o seu tempo de inferência mais elevado demonstra uma menor eficiência em tempo-real.

Tabela II: Performance dos modelos de classificação

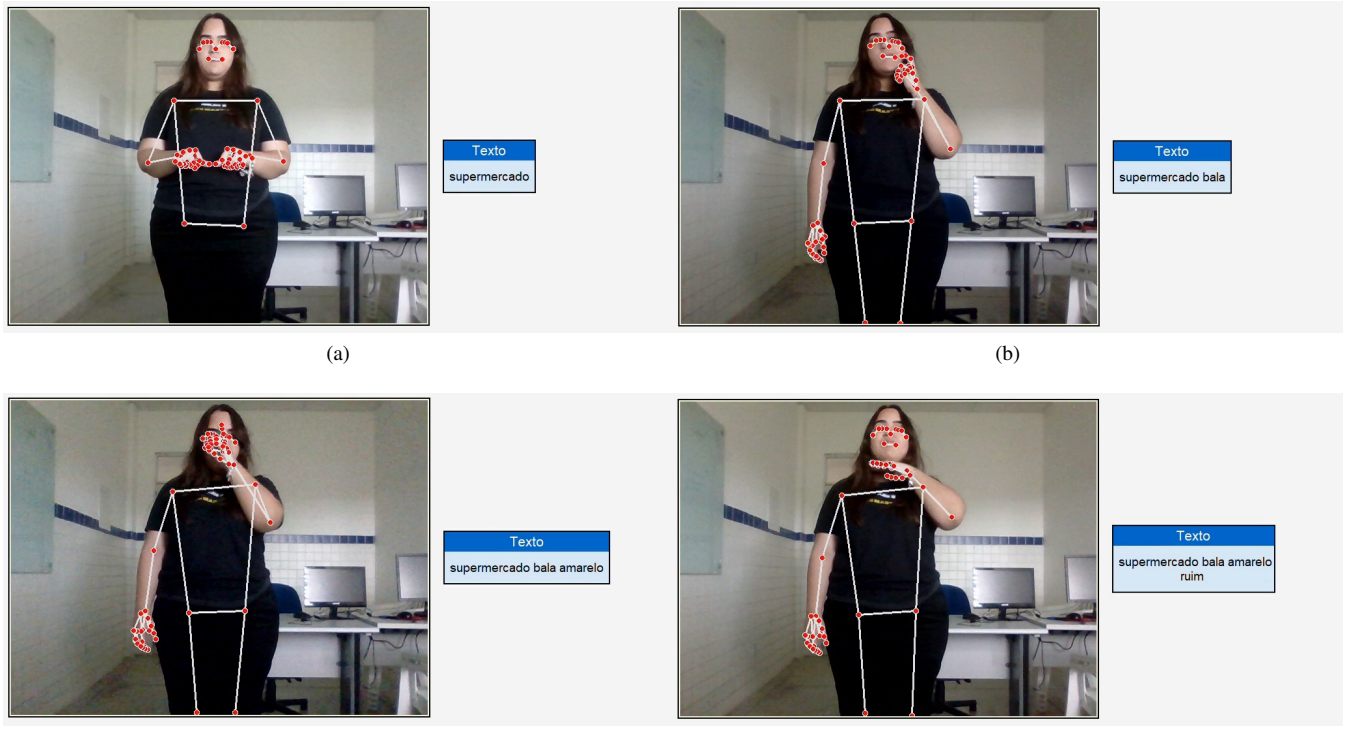
| Modelo | Acurácia | AUC | *Recall* | Precisão | *F1 Score* | Treinamento (s) | Inferência (s) |
|---|---|---|---|---|---|---|---|
| RF | **0.9956** | **1.0000** | **0.9956** | **0.9956** | **0.9956** | 83.783 | 0.002289 |
| KNN | 0.9893 | 0.9993 | 0.9893 | 0.9894 | 0.9893 | 77.46 | 0.005451 |
| DT | 0.9347 | 0.9666 | 0.9347 | 0.9347 | 0.9346 | 21.672 | **0.001553** |
| QDA | 0.9288 | 0.0000 | 0.9288 | 0.9348 | 0.9289 | 2.735 | 0.001856 |
| LDA | 0.6762 | 0.0000 | 0.6762 | 0.7297 | 0.6866 | 2.243 | 0.001847 |
| LR | 0.6693 | 0.0000 | 0.6693 | 0.6745 | 0.6667 | 52.775 | 0.001901 |
| Ridge | 0.5951 | 0.0000 | 0.5951 | 0.56 | 0.5598 | 0.581 | 0.001855 |
| LightGBM | 0.5759 | 0.7882 | 0.5759 | 0.6073 | 0.575 | 1511.311 | 0.001670 |
| SVM | 0.5498 | 0.0000 | 0.5498 | 0.693 | 0.5473 | 5.392 | 0.001832 |
| NB | 0.5192 | 0.9302 | 0.5192 | 0.6176 | 0.5264 | 1.724 | 0.001624 |
| Ada Boost | 0.0630 | 0.0000 | 0.063 | 0.0593 | 0.0286 | 59.789 | 0.001715 |
| DC | 0.0227 | 0.5000 | 0.0227 | 0.0005 | 0.001 | **0.555** | 0.001559 |

Além disso, mesmo que os tempos de inferência dos demais modelos sejam um pouco mais rápidos que RF, suas métricas de avaliação foram significativamente mais baixas.

A Figura 8 exibe as predições do modelo *Random Forest* para 4 sinais do dataset *Libras-10* e 4 sinais do *dataset MINDS-Libras* [15].


Figura 8: Predições do modelo *Random Forest* para as amostras dos datasets *Libras-10* [14] e *MINDS-Libras* [15].

Na Figura 8, nota-se que, apesar de o modelo *Hands* não conseguir detectar uma das mãos nos sinais de "Maçã" e "Amarelo", o modelo *Pose* consegue detectar o corpo normalmente, permitindo com que o modelo *Random Forest* realize as predições mesmo em sinais em que partes das mãos são ocultadas. Além disso, os *landmarks* da face obtidas pelo modelo *Pose* permitem que as expressões faciais de sinais que




(a) (b) (c) (d)

Figura 9: Teste com vídeos externo à base de dados e resultados das predições do modelo RF em tempo real para: (a) Sinal de “Supermercado”, (b) Sinal de “Bala”, (c) Sinal de “Amarelo” e (d) Sinal de “Ruim”.

precisam delas sejam captadas pelo modelo *Random Forest*, o que acontece, por exemplo, para os sinais de “Engordar”, “Zangado”, “Acusar” e “Bala”.

A Figura 9 apresenta as predições do modelo *Random Forest* para uso em tempo real. São exibidas as predições dos sinais “Supermercado”, “Bala”, “Amarelo” e “Ruim”. Os *landmarks* dos modelos *Hands* e *Pose* são mostradas ao usuário, e as predições são concatenadas à medida que um sinal é executado. Além disso, foi implementada a funcionalidade de conversão de texto em fala, permitindo que, sempre que um novo sinal for detectado, uma voz sintética anuncie a predição. A execução do programa em tempo real teve duração de 33,48 segundos, com um total de 202 *frames* processados. O tempo médio de extração dos *landmarks* pelos modelos do *MediaPipe* foi de 0,16 segundos, com um intervalo de 2 segundos entre cada predição realizada pelo modelo *Random Forest*.

Apesar dos bons resultados obtidos com o modelo *Random Forest* para uso em tempo real, é importante destacar uma limitação relevante: a ausência de informações temporais na modelagem dos sinais. Como o modelo considera apenas os *landmarks* extraídos de quadros individuais, sem levar em conta a sequência ou o tempo entre os *frames*, a predição torna-se insensível a sinais que dependem de movimento. Essa abordagem estática pode dificultar a distinção entre sinais semelhantes em posição, mas diferentes na dinâmica, como deslocamentos, gestos com trajetórias específicas ou sinais com transições entre poses. Em outras palavras, sinais compostos por gestos contínuos ao longo do tempo podem ser interpretados de forma incorreta ou incompleta.

A Figura 10 apresenta a matriz de confusão normalizada das predições do modelo *Random Forest* para o conjunto de dados de teste. Essa matriz permite analisar a precisão do modelo para cada classe do conjunto, onde valores maiores na diagonal principal indicam uma maior precisão do modelo para aquelas classes. Dessa forma, valores fora da diagonal principal indicam predições erradas do modelo.

Diante dos resultados na diagonal principal, pode-se afirmar que o modelo *Random Forest* obteve um bom desempenho, conseguindo predizer corretamente 100% das amostras de 20 classes. Entretanto, para as classes de sinais que necessitam de movimentos, o modelo atingiu uma precisão inferior entre 80% a 90%. São exemplos desse caso os sinais de “Filho”, “Vacina”, “Banheiro” e “Bala”. Essa diferença evidencia a limitação do modelo ao considerar apenas informações espaciais estáticas dos *landmarks*, sem levar em conta a progressão temporal dos gestos. Como resultado, sinais que envolvem trajetórias ou transições ao longo do tempo tendem a ser confundidos ou mal interpretados.

## V. Conclusões

Neste trabalho, foram analisadas abordagens de visão computacional utilizando o *framework* MediaPipe para o desenvolvimento de um algoritmo de aprendizado de máquina de baixo custo computacional para o reconhecimento da Libras.

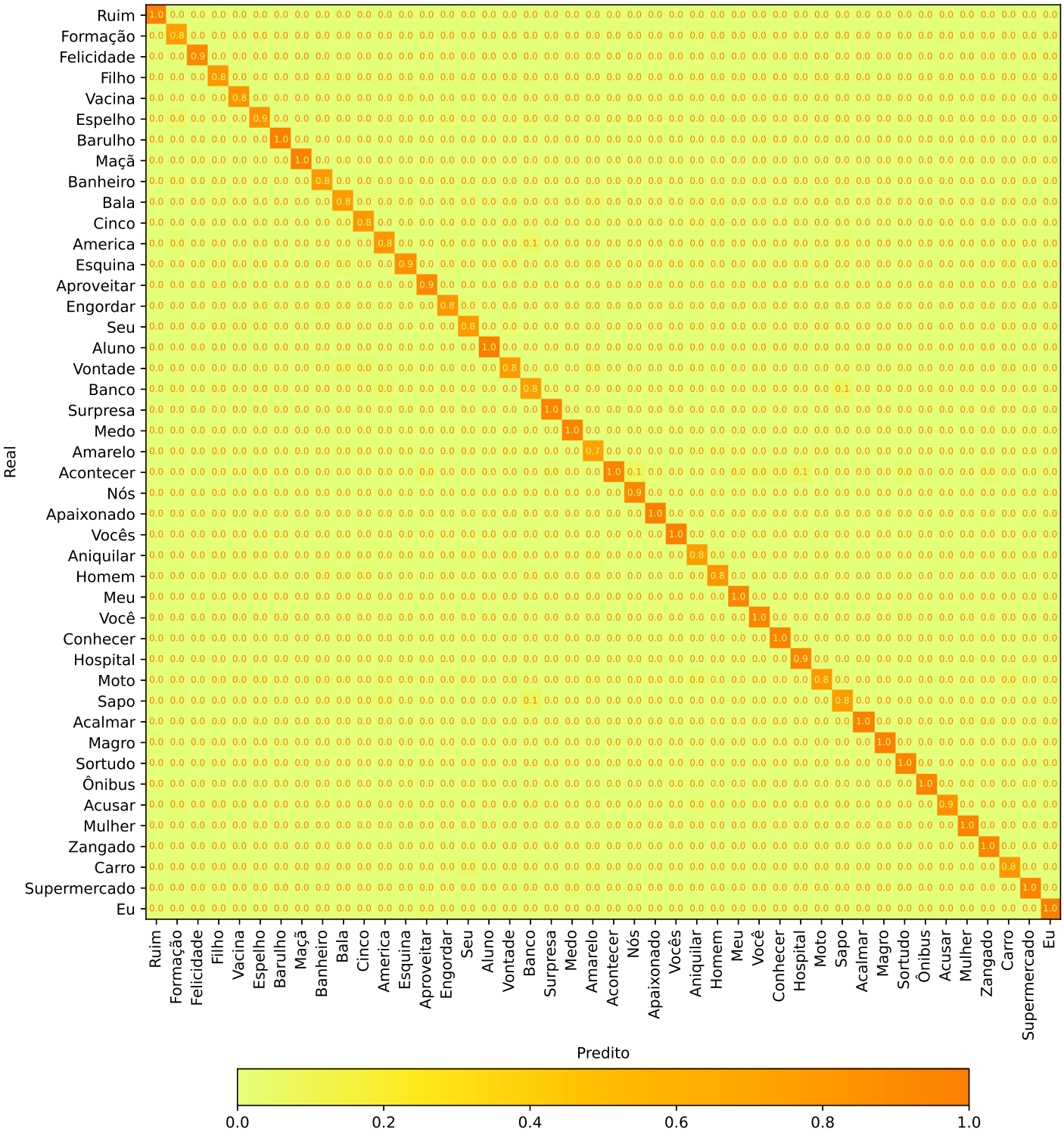



Figura 10: Matriz de confusão normalizada do modelo *Random Forest* com atenção especial a confusão entre os sinais "Surpresa", "América", "Sapo", "Nós" e "Acontecer", bem como entre "Banco" e "Sapo", e "Nós" e "Hospital", que apresentaram valor de 0.1 na matriz, indicando que o modelo teve dificuldade em distingui-los corretamente.

Os resultados obtidos indicam que o modelo *Random Forest* apresentou um alto desempenho nas predições, conforme evidenciado pelas métricas avaliadas. A matriz de confusão mostra que o modelo conseguiu predizer 27 das 44 classes com mais de 98% de precisão, o que demonstra sua capacidade de identificar corretamente a maioria dos sinais, capturando os padrões das configurações das mãos e suas posições no espaço. Entretanto, para sinais que envolvem movimentos mais complexos ou que ocultam parcialmente braços ou mãos, como os sinais "Sapo", "Seu" e "Vontade", as taxas de acerto foram inferiores. Isso se deve tanto à limitação do modelo em lidar com informações temporais, quanto às restrições dos modelos *MediaPipe Hands* e *Pose*, que, por realizarem estimativas tridimensionais a partir de imagens bidimensionais, apresentam dificuldades quando partes do corpo estão fora do campo de visão da câmera.

Ainda assim, este trabalho demonstrou que a combinação da visão computacional e do aprendizado de máquina pode contribuir significativamente para a redução de barreiras de comunicação entre pessoas ouvintes e surdas, ao desenvolver um algoritmo capaz de detectar e classificar sinais da Libras para texto. No entanto, a ausência de informações temporais no modelo, como a sequência ou o tempo entre os *frames*, limita a capacidade do sistema de capturar as dinâmicas temporais dos sinais não estáticos, que são essenciais para interpretar corretamente gestos compostos e expressões com significados dependentes da ordem dos movimentos. Ao tratar os *frames* de forma isolada, o modelo ignora padrões sequenciais importantes para a fluidez e o contexto dos sinais. Como proposta para trabalhos futuros, pretende-se implementar redes neurais recorrentes, como a LSTM, aliadas a modelos de linguagem, com o objetivo de desenvolver um sistema capaz de captar movimentos mais complexos e traduzir a sintaxe da Libras para o português de forma mais precisa e contextualizada.

## AGRADECIMENTOS

Ao Laboratório de Inteligência Computacional (CILab) da UFERSA pelo suporte à pesquisa e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro por meio da bolsa de Iniciação Científica através do Programa Institucional de Bolsas de Iniciação Científica (PIBIC/CNPq) UFERSA edital PROPPG 21/2024. Este trabalho foi financiado pelo CNPq, Código de Financiamento 0001.

## REFERÊNCIAS

[1] Governo Federal do Brasil, "VLibras – Plataforma de Acessibilidade em Língua Brasileira de Sinais," 2025, acesso em: 21 jul. 2025. [Online]. Available: https://www.gov.br/governodigital/pt-br/acessibilidade-e-usuario/vlibras

[2] R. Castro, "Níveis linguísticos da libras," 2019.

[3] L. M. de Morais, W. M. Almeida, and R. C. Rego, "Machine learning approaches for efficient recognition of brazilian sign language," in *Simp'osio Brasileiro de Sistemas de Informaç ao (SBSI)*. SBC, 2025, pp. 49–56.

[4] C. S. Rodrigues and F. Valente, *Aspectos linguísticos da LIBRAS*. IESDE Brasil SA, 2012.

[5] B. Passos, "Reconhecimento de gestos do alfabeto da língua brasileira de sinais utilizando técnicas de visão computacional," *Trabalho de Conclusão de Curso, Universidade do Vale do Itajaí*, 2019.

[6] European Sign Language Centre, "SpreadTheSign – Dicionário internacional de línguas de sinais," 2025, acesso em: 21 jul. 2025. [Online]. Available: https://www.spreadthesign.com/pt.br/

[7] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[8] A. P. G and A. P. K, "Design of an integrated learning approach to assist real-time deaf application using voice recognition system," *Computers and Electrical Engineering*, vol. 102, p. 108145, 2022.

[9] N. F. d. A. Junior *et al.*, "Brazilian sign language translation: Ai for the inclusion of deaf people," in *Anais do Congresso Brasileiro Interdisciplinar em Ciência e Tecnologia*, Diamantina, MG, Brasil, 2024.

[10] T. M. Rezende, S. G. M. Almeida, and F. G. Guimarães, "Development and validation of a brazilian sign language database for human gesture recognition," *Zenodo*, 2021.

[11] D. Lima, A. Salvador Neto, E. Santos, T. Araujo, and T. Do Rêgo, "Using convolutional neural networks for fingerspelling sign recognition in brazilian sign language," in *International Conference on Pattern Recognition Applications and Methods*, 2019.

[12] A. K. N. Costa, "Reconhecimento gestual da língua de sinais com técnicas de inteligência artificial: uma revisão sistemática da literatura," 2025.

[13] A. C. Carneiro, L. B. Silva, and D. P. Salvadeo, "Efficient sign language recognition system and dataset creation method based on deep learning and image processing," in *Thirteenth International Conference on Digital Image Processing (ICDIP 2021)*, vol. 11878. SPIE, 2021, pp. 11–19.

[14] S. G. M. Almeida, T. M. Rezende, A. C. R. Toffolo, and C. L. de Castro, "Libras-10 dataset," May 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3229958

[15] S. G. M. Almeida, T. M. Rezende, G. T. B. Almeida, A. C. R. Toffolo, and F. G. Guimarães, "Minds-libras dataset," Jul. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2667329

[16] C. L. M. Ferraz, *Dicionário de Configurações das Mãos em Libras*. UFRB, 2019. [Online]. Available: https://www.ufrb.edu.br/editora/titulos-publicados

[17] G. A. Edge, "Guia de detecção de pontos de referência manuais," https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker?hl=pt-br, 2025.

[18] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "Blazepose: On-device real-time body pose tracking," *arXiv preprint arXiv:2006.10204*, 2020.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[20] K. W. Bowyer, N. V. Chawla, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *CoRR*, vol. abs/1106.1813, 2011. [Online]. Available: http://arxiv.org/abs/1106.1813

[21] A. Géron, *Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow*. Alta Books, 2019. [Online]. Available: https://books.google.com.br/books?id=Z0mvDwAAQBAJ

[22] J. Grus, *Data Science Do Zero: noções fundamentais com python*. Alta Books, 2021. [Online]. Available: https://books.google.com.br/books?id=DG0rEAAAQBAJ

[23] E. Fix, *Discriminatory analysis: nonparametric discrimination, consistency properties*. USAF school of Aviation Medicine, 1985, vol. 1.

[24] R. Fisher, "The use of multiple measurements in taxonomic problems, annals of eugenics 7, 179-188," *Fisher1797Annals Eugenics1936*, 1936.

[25] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[26] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[27] C. Cortes and V. Vapnik, "Support-vector networks machine learning 20 (3): 273–297," 1995.

[28] B. P. Carlin and T. A. Louis, "Bayes and empirical bayes methods for data analysis," 1997.

[29] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.