INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS FLORIANO
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

IZAQUE NICOLAS VIEIRA DE MELO

LIBRAS-VC: UMA PLATAFORMA WEB PARA RECONHECIMENTO E TRADUÇÃO EM LÍNGUA BRASILEIRA DE SINAIS POR VISÃO COMPUTACIONAL

FLORIANO – PI
2026

IZAQUE NICOLAS VIEIRA DE MELO

LIBRAS-VC: UMA PLATAFORMA WEB PARA RECONHECIMENTO E TRADUÇÃO EM LÍNGUA BRASILEIRA DE SINAIS POR VISÃO COMPUTACIONAL

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia do Piauí – Campus Floriano, como requisito parcial para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Me. Ronaldo Pires Borges

FLORIANO – PI
2026

LIBRAS-VC: UMA PLATAFORMA WEB PARA RECONHECIMENTO E TRADUÇÃO EM LÍNGUA BRASILEIRA DE SINAIS POR VISÃO COMPUTACIONAL

LIBRAS-VC: A Web Platform for Sign Language Recognition and Translation using Computer Vision

Izaque Nicolas Vieira de Melo[1]
Prof. Me. Ronaldo Pires Borges[2]

RESUMO

A comunicação entre pessoas surdas e ouvintes no Brasil é frequentemente comprometida pela escassez de ferramentas tecnológicas que possibilitem a tradução automática da Língua Brasileira de Sinais (LIBRAS). Este trabalho apresenta o LIBRAS-VC, uma plataforma web desenvolvida segundo a metodologia Design Science Research (DSR), voltada ao reconhecimento e à tradução de sinais LIBRAS em tempo real por meio de visão computacional. O sistema integra a biblioteca MediaPipe para extração de *landmarks* corporais das mãos, rosto e tronco superior, compondo um vetor de características de 156 dimensões por quadro. Para a classificação de sinais estáticos, emprega-se o algoritmo Random Forest; para sinais dinâmicos, utiliza-se o Dynamic Time Warping (DTW). Um módulo complementar de pós-processamento linguístico, baseado em Modelos de Linguagem de Grande Escala (LLM), converte sequências de tokens reconhecidos em frases em português natural. A arquitetura da solução é composta por um *backend* em FastAPI e um *frontend* em React, com 21 *endpoints* REST e padrão MVVM na interface. O *dataset* construído contém 426 amostras distribuídas em 17 classes do alfabeto e sinais especiais em LIBRAS. Os resultados evidenciam a viabilidade da abordagem computacional proposta para a promoção da acessibilidade digital e da inclusão social da comunidade surda.

[1] IFPI – Campus Floriano. E-mail: caflo.2024114tads0031@aluno.ifpi.edu.br
[2] IFPI – Campus Floriano. E-mail: ronaldo.borges@ifpi.edu.br

Palavras-chave: LIBRAS; Visão Computacional; Aprendizado de Máquina; Plataforma Web; Acessibilidade Digital.

ABSTRACT

Communication between deaf and hearing people in Brazil is frequently hindered by the lack of technological tools capable of automatically translating Brazilian Sign Language (LIBRAS). This paper presents LIBRAS-VC, a web platform developed following the Design Science Research (DSR) methodology, aimed at recognizing and translating LIBRAS signs in real time through computer vision. The system integrates the MediaPipe library to extract body landmarks from hands, face, and upper trunk, composing a 156-dimensional feature vector per frame. For static sign classification, the Random Forest algorithm is employed; for dynamic signs, Dynamic Time Warping (DTW) is used. A complementary linguistic post-processing module, based on Large Language Models (LLM), converts sequences of recognized tokens into natural Portuguese sentences. The system architecture consists of a FastAPI backend and a React frontend, featuring 21 REST endpoints and the MVVM pattern in the interface. The constructed dataset contains 426 samples distributed across 17 classes of the LIBRAS alphabet and special signs. Results demonstrate the feasibility of the proposed computational approach for promoting digital accessibility and social inclusion of the deaf community.

Keywords: LIBRAS; Computer Vision; Machine Learning; Web Platform; Digital Accessibility.

Data de aprovação: 09/07/2026

1 INTRODUÇÃO

A comunicação é um direito fundamental do ser humano e condição indispensável para o exercício pleno da cidadania. Para a comunidade surda brasileira, esse direito é exercido primordialmente por meio da Língua Brasileira de Sinais (LIBRAS), reconhecida como língua oficial da comunidade surda no Brasil pela Lei Federal nº 10.436, de 24 de abril de 2002 (Brasil, 2002), e regulamentada pelo Decreto nº 5.626, de 22 de dezembro de 2005 (Brasil, 2005). De acordo com o Censo Demográfico de 2010, realizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), aproximadamente 9,7 milhões de brasileiros possuem algum grau de deficiência auditiva, dos quais cerca de 344 mil são surdos profundos (IBGE, 2012). Essa parcela

expressiva da população enfrenta cotidianamente barreiras de comunicação que comprometem seu acesso à saúde, à educação, ao emprego e aos serviços públicos.

A barreira comunicacional entre surdos e ouvintes se perpetua, em grande medida, pela ausência de ferramentas tecnológicas acessíveis que possibilitem a tradução automática de sinais LIBRAS para a língua portuguesa em tempo real. Embora existam dicionários visuais e plataformas de ensino de LIBRAS, sistemas capazes de reconhecer sinais automaticamente por meio de visão computacional e traduzi-los para o português natural ainda são escassos, especialmente no contexto de aplicações web de amplo acesso (Rastgoo; Kiani; Escalera, 2021).

Diante desse cenário, o presente trabalho propõe o desenvolvimento do LIBRAS-VC, uma plataforma web voltada ao reconhecimento e à tradução de sinais LIBRAS em tempo real. A solução integra a biblioteca MediaPipe (Lugaresi et al., 2019) para extração de *landmarks* corporais das mãos, rosto e tronco superior, compondo um vetor de características de 156 dimensões por quadro de vídeo. Para a classificação de sinais estáticos, emprega-se o algoritmo Random Forest (Breiman, 2001); para sinais dinâmicos, utiliza-se o Dynamic Time Warping (DTW) (Sakoe; Chiba, 1978). Um módulo complementar de pós-processamento linguístico, baseado em Modelos de Linguagem de Grande Escala (LLM), converte as sequências de tokens reconhecidos em frases em português natural.

A pesquisa foi conduzida segundo a metodologia Design Science Research (DSR) (Peffers et al., 2007), que orienta a construção e a avaliação sistemática de artefatos computacionais como resposta a problemas reais identificados. A adoção dessa metodologia justifica-se pela natureza aplicada do trabalho, que visa não apenas propor uma solução técnica, mas também avaliar sua efetividade no contexto da promoção da acessibilidade digital e da inclusão social.

O objetivo geral deste trabalho é **desenvolver uma plataforma web para reconhecimento e tradução de sinais LIBRAS em tempo real, utilizando visão computacional e aprendizado de máquina**, buscando reduzir as barreiras de comunicação entre pessoas surdas e ouvintes. Para alcançar esse objetivo, foram definidos os seguintes objetivos específicos: (i) identificar e modelar os requisitos funcionais e não funcionais da plataforma; (ii) construir um pipeline de extração de *landmarks* corporais com o MediaPipe; (iii) desenvolver classificadores para sinais estáticos e dinâmicos; (iv) integrar um módulo de pós-processamento linguístico via LLM; (v) implementar uma interface web acessível e intuitiva; e (vi) avaliar o desempenho do sistema por meio de métricas de reconhecimento.

A relevância do trabalho justifica-se em três dimensões: **social**, ao contribuir para a redução das barreiras de comunicação da comunidade surda; **tecnológica**, ao aplicar técnicas avançadas de visão computacional e aprendizado de máquina a um problema de acessibilidade; e **científica**, ao produzir um artefato avaliado segundo os

critérios rigorosos da DSR, gerando conhecimento replicável para trabalhos futuros na área.

O presente artigo está organizado da seguinte forma: a Seção 2 apresenta o referencial teórico sobre LIBRAS, acessibilidade digital, visão computacional, aprendizado de máquina, DSR e trabalhos relacionados; a Seção 3 descreve a metodologia de pesquisa adotada; a Seção 4 detalha o desenvolvimento da solução proposta; a Seção 5 apresenta os resultados obtidos e a discussão; e a Seção 6 traz as considerações finais e perspectivas futuras.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Língua Brasileira de Sinais

A LIBRAS é uma língua natural, de modalidade gestual-visual, utilizada pela comunidade surda brasileira. Segundo Quadros e Karnopp (2004), como toda língua natural, ela possui gramática própria, com estrutura sintática, morfológica e fonológica independente da língua portuguesa. Sua gramática é organizada em torno de cinco parâmetros fonológicos: configuração de mão (CM), movimento (M), locação ou ponto de articulação (L), orientação da palma da mão (O) e expressão facial e/ou corporal (EF). A combinação desses parâmetros determina o significado de cada sinal, tornando a LIBRAS um sistema linguístico completo e autônomo. A Figura 1 apresenta o alfabeto manual da LIBRAS, que ilustra a diversidade de configurações de mão – e, em algumas letras, o movimento indicado pelas setas – parâmetros fonológicos que o reconhecimento automático precisa distinguir.

O reconhecimento da LIBRAS como língua oficial da comunidade surda pela Lei nº 10.436/2002 representou um marco histórico para a inclusão social no Brasil (Brasil, 2002). O Decreto nº 5.626/2005 aprofundou essa conquista ao estabelecer a obrigatoriedade do ensino de LIBRAS nos cursos de formação de professores, a garantia do direito dos surdos ao acesso a intérpretes e tradutores em serviços públicos e a inserção da LIBRAS como disciplina curricular (Brasil, 2005). Contudo, a implementação prática de tais direitos ainda enfrenta desafios significativos, especialmente no que se refere ao acesso de pessoas surdas a tecnologias de comunicação automatizada.

Do ponto de vista computacional, a LIBRAS apresenta características que tornam seu reconhecimento automático um problema complexo: a variabilidade interindividual na execução dos sinais, a coarticulação (influência de sinais vizinhos uns sobre os outros), a presença de sinais com e sem movimento, e a importância das expressões faciais para a diferenciação gramatical. Essas particularidades motivam a utilização de abordagens híbridas que combinem extração de *landmarks* com algoritmos de classificação adequados a cada tipo de sinal.

Figura 1 – Alfabeto manual da LIBRAS, evidenciando as diferentes configurações de mão e, em algumas letras, o movimento

Fonte: Clube de Libras da Universidade Federal do Ceará. Disponível em: <https://clubedelibras.ufc.br/wp-content/uploads/2021/04/facema-cartilha-libras-enfermagem.pdf>. Acesso em: jul. 2026.

2.2 Acessibilidade Digital e Inclusão Social

A acessibilidade digital diz respeito à garantia de que produtos e serviços digitais possam ser utilizados por pessoas com deficiência em igualdade de condições com as demais. O World Wide Web Consortium (W3C), por meio das Diretrizes de Acessibilidade para Conteúdo Web (WCAG 2.1), estabelece quatro princípios fundamentais: *perceptível*, *operável*, *compreensível* e *robusto* (W3C, 2018). O cumprimento desses princípios é essencial para que plataformas web atendam às necessidades de usuários com diferentes capacidades sensoriais e motoras.

A Convenção sobre os Direitos das Pessoas com Deficiência da Organização das Nações Unidas (ONU), adotada em 2006 e ratificada pelo Brasil em 2008, determina que os estados signatários devem promover o acesso das pessoas com deficiência à tecnologia da informação e comunicação, inclusive à internet (ONU, 2006). No âmbito nacional, a Lei Brasileira de Inclusão (Lei nº 13.146/2015) reforça esse compromisso ao estabelecer o direito à tecnologia assistiva como instrumento de promoção da autonomia, independência e qualidade de vida das pessoas com deficiência (Brasil, 2015).

Nesse contexto, sistemas de reconhecimento e tradução de língua de sinais representam uma contribuição direta à promoção da acessibilidade digital. A disponibilização desses sistemas como plataformas web amplia seu alcance, uma vez que elimina a necessidade de instalação de *software* específico e permite o acesso por

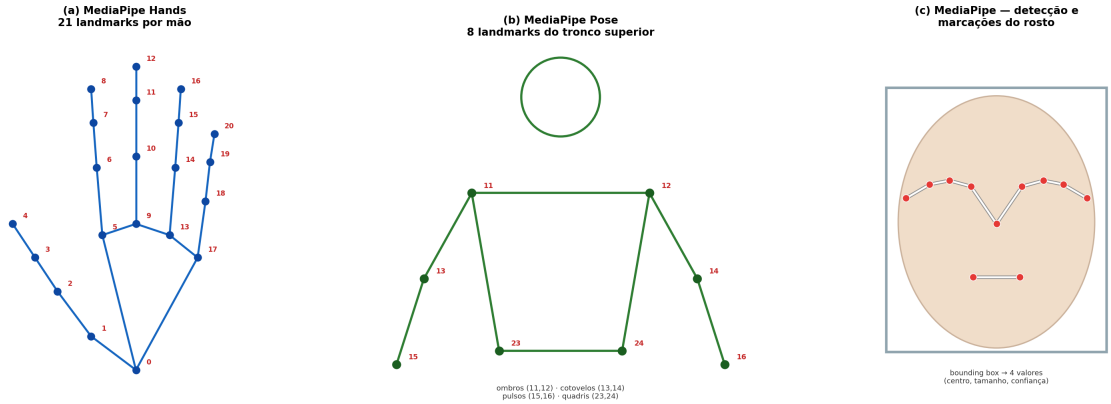
diferentes dispositivos e sistemas operacionais.

2.3 Visão Computacional Aplicada ao Reconhecimento de Gestos

A visão computacional é um campo interdisciplinar que habilita computadores a interpretar e compreender o conteúdo visual do mundo por meio de algoritmos capazes de processar imagens e vídeos digitais. O reconhecimento de gestos por visão computacional tem avançado significativamente com o advento de redes neurais profundas e de bibliotecas especializadas, como o MediaPipe, desenvolvido pelo Google Research (Lugaresi et al., 2019).

O MediaPipe é um framework de código aberto para processamento de mídia em tempo real, que fornece soluções pré-treinadas para detecção de *landmarks* corporais. No contexto deste trabalho, são utilizadas três soluções: *MediaPipe Hands*, capaz de detectar 21 pontos de referência (*landmarks*) por mão; *MediaPipe Face Detection*, que localiza o rosto por meio de *bounding box*; e *MediaPipe Pose*, que estima 33 *landmarks* do corpo humano. O modelo de detecção de mãos do MediaPipe, descrito por Zhang et al. (2020), adota uma abordagem de dois estágios: na primeira etapa, detecta a palma da mão; na segunda, localiza com precisão os 21 pontos articulares, mesmo em condições adversas de iluminação e oclusão parcial. A Figura 2 apresenta as representações extraídas das mãos, da pose e do rosto utilizadas neste trabalho.

Figura 2 – Representações extraídas pelo MediaPipe: (a) 21 *landmarks* da mão; (b) 8 *landmarks* do tronco superior da pose; e (c) detecção do rosto (*bounding box*) com as marcações faciais



Fonte: Elaborado pelo autor.

A utilização de *landmarks* como representação do sinal, em contraposição à análise direta de pixels, oferece vantagens importantes: a representação é compacta, robusta a variações de iluminação e cor de pele, e computacionalmente eficiente para processamento em tempo real (Papastratis et al., 2021). Essa abordagem tem sido amplamente empregada em sistemas recentes de reconhecimento de língua de

sinais, que combinam extração de *landmarks* com classificadores tradicionais ou redes neurais.

2.4 Representação e Normalização de Características

A conversão dos *landmarks* detectados em um vetor de características (*feature vector*) adequado à classificação exige uma etapa de **normalização**, cujo objetivo é remover fontes de variação irrelevantes ao significado do sinal. Duas dessas fontes são particularmente críticas no reconhecimento de línguas de sinais: a **posição** do sinalizante no quadro e a **escala** aparente das mãos, que varia conforme a distância à câmera. Sem normalização, um mesmo sinal executado em posições ou distâncias diferentes produziria vetores de características distintos, dificultando a generalização do modelo.

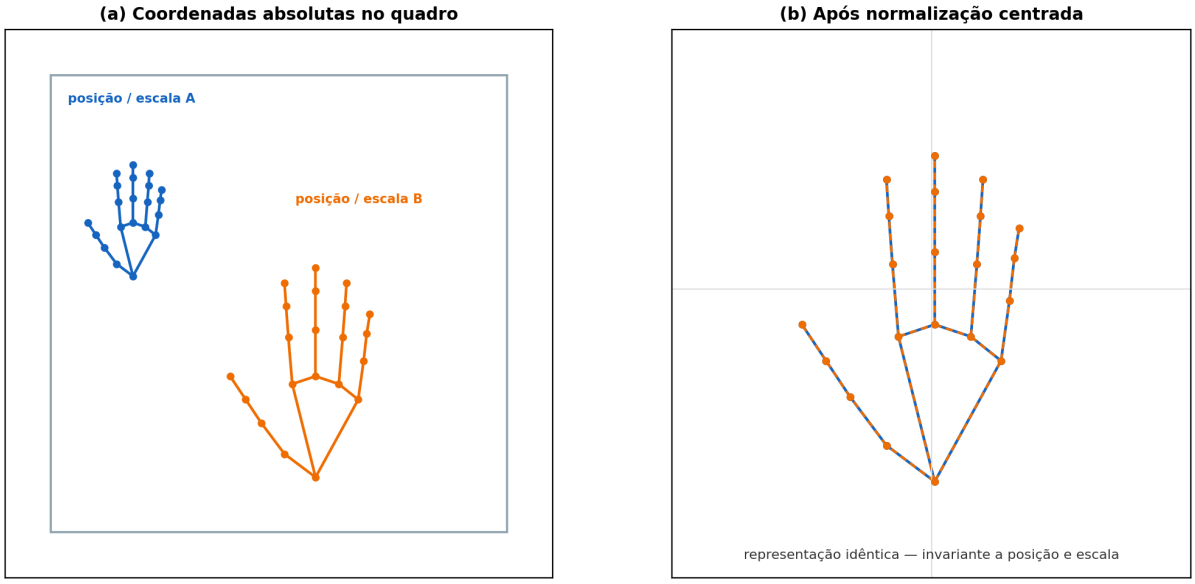
Uma estratégia amplamente empregada consiste em expressar as coordenadas dos *landmarks* de forma **relativa**, e não absoluta. Na normalização centrada, as coordenadas de cada mão são transladadas em relação ao seu centroide e reescaladas por um fator de dispersão, tornando a representação invariante à translação e à escala (Papastratis et al., 2021). Adicionalmente, a padronização das *features* – subtração da média e divisão pelo desvio padrão – antes do treinamento equaliza a contribuição de cada dimensão, evitando que atributos de maior magnitude dominem o aprendizado. Essas transformações reduzem a variabilidade intraclasse e favorecem o reconhecimento independente do sinalizante e das condições de captura. A Figura 3 ilustra esse efeito.

2.5 Segmentação Temporal e Detecção de Movimento

Diferentemente do reconhecimento de sinais isolados a partir de um único quadro, o reconhecimento de sinais que envolvem movimento requer o tratamento de **sequências temporais** de quadros. Isso impõe dois desafios adicionais: (i) determinar **quando** um gesto começa e termina – problema de **segmentação temporal** – e (ii) decidir se um trecho de vídeo corresponde a um sinal **estático** (sem trajetória) ou **dinâmico** (com trajetória), o que define qual classificador deve ser acionado.

Madeo et al. (2016) investigaram justamente a segmentação de unidades gestuais em sequências contínuas, evidenciando a importância de delimitar corretamente os limites de cada gesto para o reconhecimento em fluxo. Uma abordagem comum para essa delimitação é a **detecção de movimento**: mede-se a magnitude da variação das características entre quadros consecutivos e, a partir de um limiar, distinguem-se períodos de repouso de períodos de gesto ativo. Esse mesmo indicador pode ser utilizado para **rotear** a inferência entre o classificador de sinais estáticos e o de sinais dinâmicos. Na prática, os quadros são acumulados em um ***buffer*** temporal, cujo

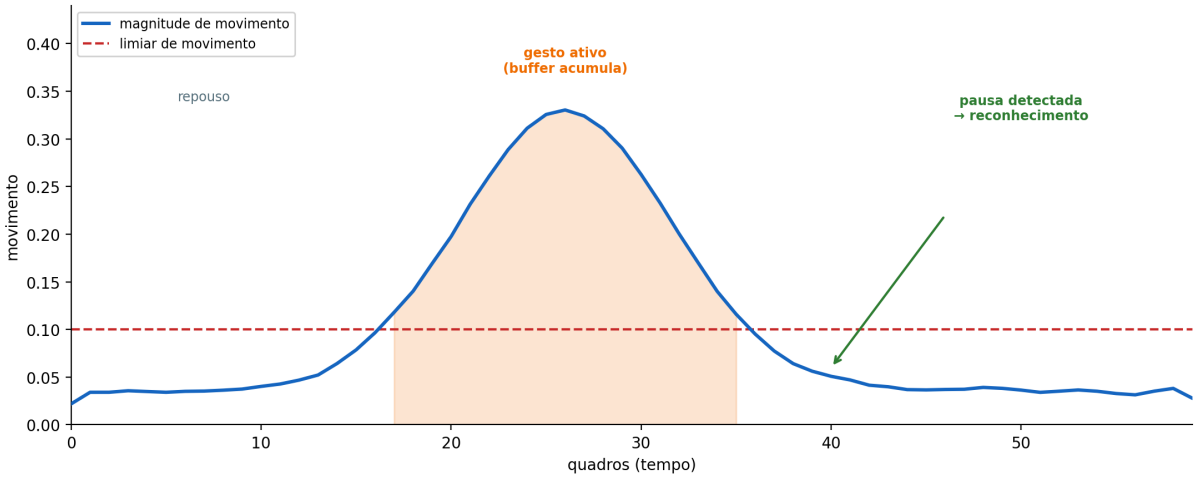
Figura 3 – Efeito da normalização dos *landmarks*: (a) uma mesma mão em posições e escalas distintas no quadro; (b) representação idêntica após a normalização centrada



Fonte: Elaborado pelo autor.

conteúdo é submetido ao reconhecimento quando se detecta uma pausa no movimento, sinalizando a conclusão do gesto. A Figura 4 ilustra esse mecanismo.

Figura 4 – Detecção de movimento e segmentação temporal do gesto: acúmulo de quadros no *buffer* durante o movimento e disparo do reconhecimento na pausa



Fonte: Elaborado pelo autor.

2.6 Aprendizado de Máquina para Reconhecimento de Padrões

O aprendizado de máquina (do inglês, *machine learning*) compreende um conjunto de técnicas computacionais que permitem que sistemas aprendam automaticamente a partir de dados, sem serem explicitamente programados para cada tarefa específica. No contexto do reconhecimento de sinais LIBRAS, dois algoritmos se destacam pela adequação a cada um dos tipos de sinal discutidos na seção anterior: o Random Forest, para os sinais estáticos, e o Dynamic Time Warping, para os sinais dinâmicos.

O **Random Forest**, proposto por Breiman (2001), é um método de aprendizado por conjunto (*ensemble*) que constrói múltiplas árvores de decisão sobre subamostras aleatórias do conjunto de treinamento. A predição final é determinada por votação majoritária entre todas as árvores. O algoritmo oferece alta resistência ao sobreajuste (*overfitting*), boa generalização em conjuntos de dados de dimensionalidade moderada e não exige pré-processamento extensivo das *features*. Essas características tornam o Random Forest adequado para a classificação de sinais estáticos, cujas *features* são extraídas de um único quadro de vídeo.

O **Dynamic Time Warping (DTW)**, originalmente proposto por Sakoe e Chiba (1978) para o reconhecimento de voz, é um algoritmo de alinhamento de séries temporais que mede a similaridade entre duas sequências de comprimentos potencialmente diferentes. Diferentemente da distância euclidiana, que exige correspondência ponto a ponto, o DTW encontra, por programação dinâmica, o alinhamento não linear de menor custo entre as sequências, sendo especialmente adequado ao reconhecimento de sinais dinâmicos, nos quais um mesmo sinal pode ser executado com velocidades e durações distintas.

Dadas duas sequências $X = (x_1, \ldots, x_n)$ e $Y = (y_1, \ldots, y_m)$, o DTW constrói uma matriz de custo acumulado D na qual cada célula é obtida pela relação de recorrência

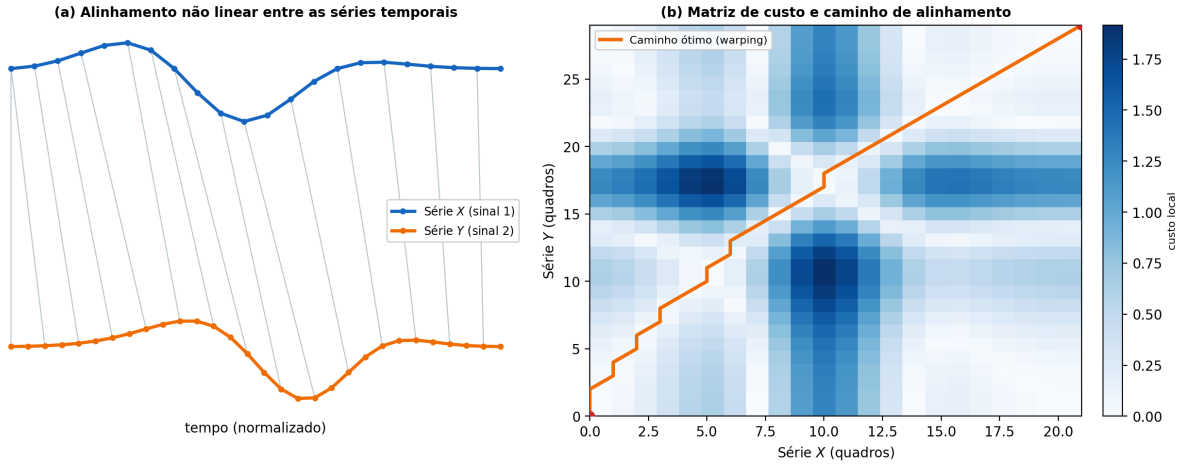
$$D(i, j) = d(x_i, y_j) + \min\{\, D(i-1, j),\ D(i, j-1),\ D(i-1, j-1)\,\}, \tag{1}$$

onde $d(x_i, y_j)$ é a distância local (euclidiana) entre os quadros x_i e y_j. O valor $D(n, m)$ corresponde ao custo do alinhamento ótimo, sujeito às restrições de contorno, monotonicidade e continuidade que garantem um caminho de *warping* válido (Müller, 2007).

Como o custo acumulado cresce com o comprimento do caminho de alinhamento, adota-se neste trabalho a **normalização pelo comprimento do caminho**, dividindo o custo total pelo número de passos do alinhamento ótimo, o que torna a distância comparável entre sequências de durações diferentes. A classificação é então realizada por **vizinho mais próximo** (*nearest-neighbor*): a sequência a reconhecer é comparada a um conjunto de *templates* de referência de cada classe, atribuindo-se a classe cujo *template* apresenta a menor distância DTW normalizada. Essa abordagem baseada

em *templates* dispensa grandes volumes de dados por classe, característica adequada ao estágio inicial de construção do *dataset* dinâmico. A Figura 5 ilustra o alinhamento não linear produzido pelo DTW entre duas séries temporais e o caminho de menor custo na matriz de alinhamento.

Figura 5 – Funcionamento do *Dynamic Time Warping*: (a) alinhamento não linear entre duas séries temporais e (b) caminho de menor custo na matriz de alinhamento



Fonte: Elaborado pelo autor.

2.7 Métricas de Avaliação de Classificadores

A avaliação de modelos de classificação apoia-se em métricas derivadas da comparação entre as classes previstas e as classes verdadeiras. A **acurácia** expressa a proporção de predições corretas sobre o total, mas pode ser enganosa em conjuntos de dados desbalanceados. Para uma análise mais criteriosa, empregam-se a **precisão** (proporção de predições positivas que estão corretas), a **revocação** (*recall*, proporção de exemplos positivos corretamente identificados) e o ***F1-score***, que é a média harmônica entre precisão e revocação, equilibrando ambas em um único valor (Powers, 2011). A **matriz de confusão** complementa essas medidas ao detalhar, para cada classe, os acertos e os tipos de erro cometidos, permitindo identificar pares de classes frequentemente confundidos.

Para que as métricas reflitam a capacidade de **generalização** do modelo – e não apenas seu desempenho nos dados de treino –, a avaliação deve ser conduzida sobre dados não vistos durante o treinamento. Técnicas como a divisão estratificada entre treino e validação e a **validação cruzada** (*k-fold cross-validation*) reduzem a dependência de uma única partição dos dados e fornecem estimativas mais estáveis do desempenho, sendo especialmente relevantes em conjuntos de dados de tamanho moderado (Kohavi, 1995). O acompanhamento do desempenho em função do volume

de dados de treino, por meio da **curva de aprendizado**, auxilia ainda no diagnóstico de sobreajuste (*overfitting*) e de subajuste (*underfitting*).

2.8 Modelos de Linguagem e Tradução de Glosas

Nas línguas de sinais, a representação escrita de um sinal é usualmente feita por meio de uma **glosa**: uma etiqueta em língua oral – por convenção, grafada em caixa alta – que identifica o sinal, mas que não reproduz a estrutura gramatical nem da língua de sinais nem da língua oral (Quadros; Karnopp, 2004). Dessa forma, a sequência de glosas produzida pelo reconhecimento (por exemplo, `EU CASA IR`) não constitui uma frase gramaticalmente correta em português, sendo necessária uma etapa de **pós-processamento linguístico** que a converta em texto fluente, com conjugação verbal, preposições e concordância de gênero e número. Ao longo deste trabalho, as glosas reconhecidas pelo sistema são também denominadas *tokens*, termo adotado na descrição da implementação.

Historicamente, essa conversão era realizada por sistemas baseados em regras, de difícil manutenção e cobertura limitada. Com o avanço do Processamento de Linguagem Natural (PLN), os **Modelos de Linguagem de Grande Escala** (*Large Language Models* – LLMs) tornaram-se o estado da arte para tarefas de geração e tradução de texto. Tais modelos baseiam-se na arquitetura *Transformer* (Vaswani et al., 2017), que emprega mecanismos de atenção para capturar dependências de longo alcance, e são pré-treinados em grandes corpora textuais, adquirindo conhecimento sintático e semântico que pode ser aplicado a novas tarefas por meio de instruções em linguagem natural (*prompting*) (Brown et al., 2020).

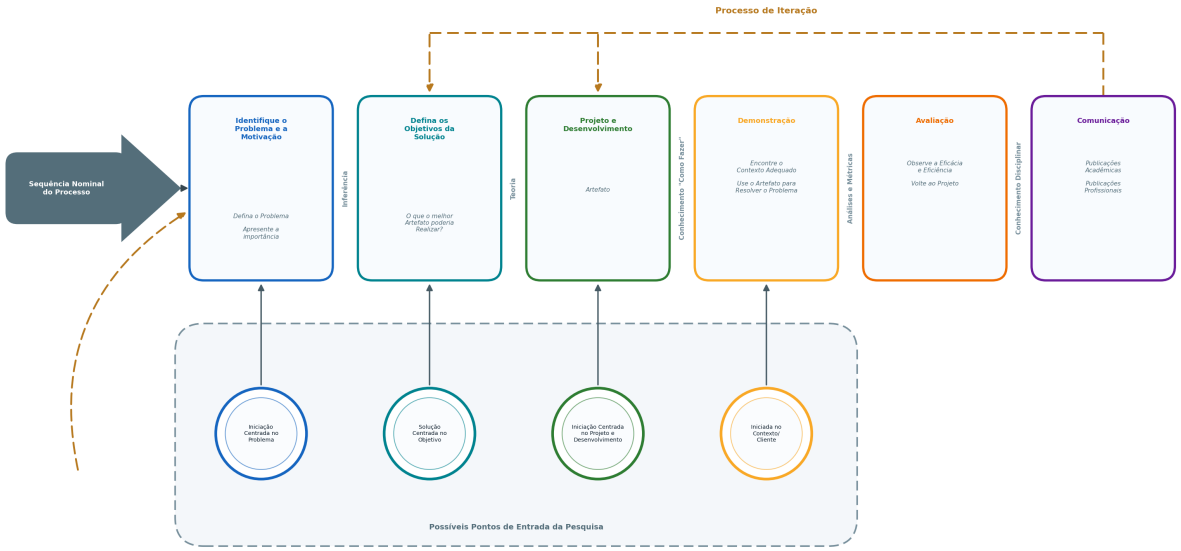
No contexto do reconhecimento de línguas de sinais, o emprego de modelos neurais na etapa de tradução – e não apenas no reconhecimento visual – tem sido explorado como forma de gerar frases mais naturais a partir das glosas reconhecidas (Camgoz et al., 2018). Neste trabalho, a tradução das glosas para o português é delegada a um LLM (Google Gemini, com alternativa OpenAI GPT), o que desloca a geração de linguagem natural de uma camada de regras fixas para um modelo generativo capaz de produzir frases contextualmente coerentes (Anil et al., 2023).

2.9 Design Science Research

A Design Science Research (DSR) é uma metodologia de pesquisa em sistemas de informação que tem como objetivo a criação e a avaliação rigorosa de artefatos computacionais como resposta a problemas relevantes identificados na prática. Proposta inicialmente por Hevner et al. (2004) e formalizada como processo metodológico estruturado por Peffers et al. (2007), a DSR orienta pesquisadores a construir soluções inovadoras, avaliá-las segundo critérios científicos e comunicar suas contribuições para os públicos técnico e acadêmico.

Peffers et al. (2007) propõem um processo de pesquisa composto por seis atividades iterativas: (1) **identificação do problema e motivação**; (2) **definição dos objetivos da solução**; (3) **projeto e desenvolvimento do artefato**; (4) **demonstração**; (5) **avaliação**; e (6) **comunicação**. Esse ciclo assegura que o artefato construído atenda efetivamente às necessidades identificadas e que sua avaliação seja conduzida de forma sistemática, distinguindo a DSR da prática de desenvolvimento de *software* convencional. A Figura 6 apresenta esse processo e suas seis atividades.

Figura 6 – Processo de *Design Science Research* e suas seis atividades



Fonte: Adaptado de Peffers et al. (2007).

Hevner et al. (2004) estabelecem sete diretrizes para a condução de pesquisas em DSR, entre as quais se destacam: a relevância do problema para a prática; a criação de um artefato inovador; a avaliação rigorosa do artefato com métodos apropriados; e a contribuição científica gerada pela pesquisa. Essas diretrizes orientam tanto o processo de desenvolvimento quanto a comunicação dos resultados, garantindo que a pesquisa em DSR produza conhecimento científico cumulativo e aplicável.

2.10 Trabalhos Relacionados

O reconhecimento automático de língua de sinais tem sido objeto de crescente atenção na literatura científica. Uma revisão abrangente conduzida por Rastgoo, Kiani e Escalera (2021) identificou tendências e desafios na área, destacando o uso de redes neurais convolucionais (CNNs), redes recorrentes (RNNs/LSTMs) e, mais recentemente, *transformers* para o reconhecimento de sinais estáticos e dinâmicos. Koller (2020) realizou um levantamento quantitativo do estado da arte, evidenciando

que as abordagens baseadas em *landmarks* têm ganho destaque pela sua eficiência computacional em aplicações de tempo real.

No contexto brasileiro, o trabalho de Dias et al. (2021) investigou o reconhecimento de movimentos manuais para LIBRAS utilizando redes neurais baseadas em distâncias, obtendo resultados promissores para o reconhecimento de configurações de mão. Trindade, Macêdo e Zanchettin (2022) propuseram uma abordagem baseada em MediaPipe e aprendizado de máquina para LIBRAS, utilizando grafos de mãos como representação, e alcançaram acurácias elevadas em conjuntos de dados controlados. Madeo et al. (2016) investigaram a segmentação de unidades gestuais com Máquinas de Vetores de Suporte (SVMs), contribuindo para o entendimento do reconhecimento de gestos em sequências contínuas.

A Tabela 1 apresenta um comparativo entre os trabalhos relacionados identificados e a proposta do LIBRAS-VC.

Tabela 1 – Comparativo entre trabalhos relacionados e o LIBRAS-VC

Trabalho	**Abordagem**	**Língua**	**Plataforma**	**Tradução LLM**
Dias et al. (2021)	Rede neural / distâncias	LIBRAS	Desktop	Não
Trindade e Macêdo (2022)	MediaPipe + Grafo	LIBRAS	Desktop	Não
Madeo et al. (2016)	SVM	LIBRAS	Desktop	Não
Papastratis et al. (2021)	CNN / Transformer	ASL/BSL	Desktop	Não
LIBRAS-VC (2026)	**MediaPipe + RF/DTW**	**LIBRAS**	**Web**	**Sim (Gemini)**

Fonte: Elaborado pelo autor.

Observa-se que o LIBRAS-VC se diferencia dos trabalhos relacionados por: (i) oferecer uma plataforma web de acesso direto, sem instalação local; (ii) combinar o reconhecimento de sinais estáticos e dinâmicos em um único sistema; (iii) integrar um módulo de tradução linguística via LLM para a geração de frases em português natural; e (iv) disponibilizar uma interface de gerenciamento de *dataset* integrada ao sistema de reconhecimento.

3 METODOLOGIA

3.1 Caracterização da Pesquisa

Esta pesquisa pode ser classificada, quanto à sua natureza, como **aplicada**, pois objetiva gerar conhecimento para aplicação prática na solução de um problema concreto de acessibilidade e comunicação. Quanto à abordagem, trata-se de uma pesquisa **qualiquantitativa**: qualitativa no que se refere à análise do problema e ao projeto da solução; quantitativa no que diz respeito à avaliação do desempenho do sistema por meio de métricas de reconhecimento. Quanto aos objetivos, a pesquisa é de caráter **exploratório e construtivo**, uma vez que explora as possibilidades da visão computacional aplicada ao reconhecimento de LIBRAS e constrói um artefato computacional funcional como resposta ao problema identificado (Gil, 2019).

O procedimento metodológico adotado é a **Design Science Research (DSR)**, que orienta a criação e a avaliação de artefatos de tecnologia da informação como forma de produção de conhecimento científico. Conforme Peffers et al. (2007), a DSR é especialmente adequada para pesquisas em sistemas de informação que visam desenvolver soluções inovadoras para problemas práticos relevantes, produzindo conhecimento científico generalizável a partir da experiência de construção e avaliação do artefato.

3.2 Aplicação da Design Science Research

A condução deste trabalho seguiu as seis atividades do processo de DSR propostas por Peffers et al. (2007), conforme descrito a seguir.

Atividade 1 – Identificação do problema e motivação. A partir da revisão da literatura e da análise do contexto social brasileiro, identificou-se o problema central da pesquisa: a escassez de plataformas web acessíveis para o reconhecimento e a tradução automática de sinais LIBRAS, que contribui para o isolamento comunicacional da comunidade surda e para a perpetuação de barreiras de acesso a serviços e oportunidades.

Atividade 2 – Definição dos objetivos da solução. Com base no problema identificado, estabeleceu-se como objetivo o desenvolvimento de uma plataforma web capaz de reconhecer sinais LIBRAS em tempo real por meio de visão computacional e de traduzi-los para o português natural, sendo acessível por navegadores comuns sem necessidade de instalação de *software* adicional.

Atividade 3 – Projeto e desenvolvimento do artefato. O artefato foi projetado como um sistema composto por um *backend* em Python com FastAPI e um *frontend* em React, integrados por uma API REST. O desenvolvimento foi conduzido de forma iterativa, com a construção progressiva dos módulos de visão computacional, extração de *features*, classificação, tradução e interface de usuário. Os detalhes do

desenvolvimento são apresentados na Seção 4.

Atividade 4 – Demonstração. O artefato desenvolvido foi demonstrado por meio da execução do sistema em ambiente local, com reconhecimento de sinais LIBRAS a partir de um *dataset* construído com 426 amostras de vídeo distribuídas em 17 classes (395 amostras estáticas e 31 dinâmicas). A demonstração permitiu verificar o funcionamento dos módulos de captura, processamento, classificação e tradução de forma integrada.

Atividade 5 – Avaliação. A avaliação do sistema foi conduzida em duas dimensões: (i) **avaliação técnica**, por meio do treinamento e validação dos modelos de reconhecimento estático com métricas de acurácia, precisão, revocação e F1-*score*; e (ii) **avaliação qualitativa** da interface, verificando a conformidade com os requisitos definidos e a usabilidade da plataforma para os perfis de usuário alvo. Os resultados são apresentados na Seção 5.

Atividade 6 – Comunicação. Os resultados, contribuições e limitações da pesquisa são comunicados por meio deste artigo científico, conforme os padrões exigidos pelo curso de Tecnologia em Análise e Desenvolvimento de Sistemas do IFPI – Campus Floriano.

3.3 Construção do *Dataset*

O *dataset* foi construído de forma incremental utilizando a própria plataforma como ferramenta de coleta, o que assegura que as amostras de treinamento sejam capturadas nas mesmas condições da inferência. Cada amostra corresponde a um vídeo do sinal, gravado por *webcam* a 30 quadros por segundo, e é rotulada quanto ao seu tipo (*sign_type*): **estático**, para sinais sem movimento significativo, ou **dinâmico**, para sinais que envolvem trajetória. Para cada classe, o usuário grava múltiplas repetições, de modo a capturar a variabilidade natural de execução.

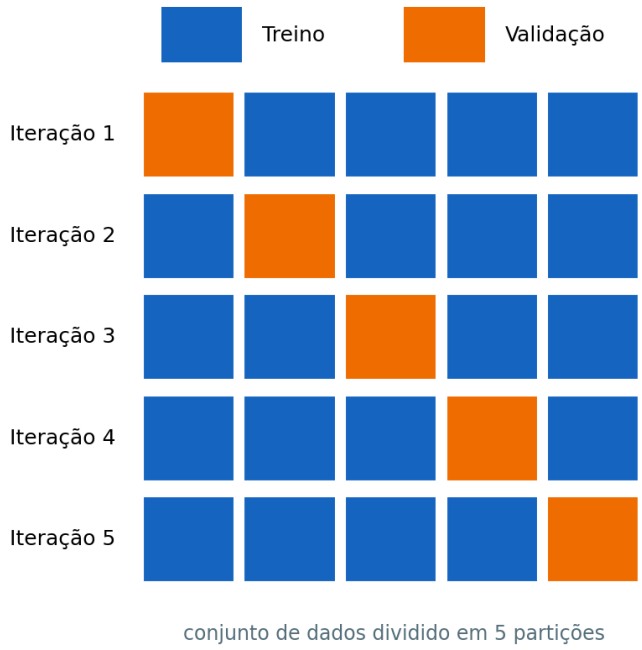
As amostras são organizadas por classe e persistidas com metadados (identificador, tipo, número de quadros e resolução), e a plataforma disponibiliza a exportação das características extraídas em formato CSV, viabilizando análises externas e a reprodutibilidade dos experimentos. A distribuição final de amostras por classe é apresentada na Seção 5. Ressalta-se que a coleta foi realizada com um único participante, o que constitui uma limitação reconhecida quanto à generalização para diferentes sinalizantes.

3.4 Protocolo de Treinamento e Avaliação

O classificador de sinais estáticos (Random Forest) foi treinado sobre o vetor de características extraído do quadro central de cada vídeo, com divisão **estratificada** de 80% para treino e 20% para validação, preservando a proporção entre classes. Os hiperparâmetros adotados foram 100 árvores de decisão e profundidade máxima de

20, com padronização das *features* (*StandardScaler*) previamente ao treinamento. A robustez do modelo foi adicionalmente avaliada por **validação cruzada com cinco dobras** (*5-fold cross-validation*) (Kohavi, 1995), sintetizada na curva de aprendizado. A Figura 7 ilustra esse procedimento.

Figura 7 – Validação cruzada com cinco dobras (*5-fold cross-validation*)



Fonte: Elaborado pelo autor.

O classificador de sinais dinâmicos (DTW) não requer treinamento paramétrico: as sequências de características extraídas dos vídeos rotulados como dinâmicos são armazenadas diretamente como *templates* de referência de cada classe. O desempenho de ambos os classificadores foi mensurado pelas métricas de classificação apresentadas na Seção 2.7 (acurácia, precisão, revocação e *F1-score*), complementadas pela matriz de confusão. A avaliação da interface seguiu abordagem qualitativa, fundamentada nos princípios da WCAG 2.1 (W3C, 2018) e nas heurísticas de usabilidade descritas por Preece, Rogers e Sharp (2013).

Cabe ressaltar que o protocolo de avaliação adotado é de natureza técnica e analítica – baseado em métricas quantitativas de classificação e na inspeção da interface segundo diretrizes de acessibilidade e usabilidade –, não tendo contado com a participação direta de pessoas surdas ou de profissionais de LIBRAS. Dessa forma, embora os resultados evidenciem a viabilidade da abordagem, a validação da adequação linguística dos sinais e da real utilidade da ferramenta para o público-alvo permanece como uma limitação deste trabalho. Recomenda-se, como desdobramento futuro, a realização de uma avaliação participativa que envolva usuários surdos e

intérpretes ou professores de LIBRAS, capaz de verificar a correção dos sinais coletados, aferir a usabilidade em contexto real de uso e orientar o aprimoramento do sistema.

3.5 Calibração do Classificador Dinâmico

Uma etapa metodológica específica foi conduzida para calibrar a camada de decisão do classificador dinâmico, cuja confiança é obtida pela transformação $conf = e^{-d/\tau}$, sendo d a distância DTW normalizada e τ um parâmetro de escala. Para determinar τ e o limiar de aceitação de forma criteriosa, adotou-se um protocolo de validação ***leave-one-out***: cada amostra dinâmica foi classificada por vizinho mais próximo contra todos os demais *templates*, excluindo o seu próprio, evitando o vício de avaliação (*data leakage*). Adicionalmente, cada amostra foi reprocessada a aproximadamente quatro quadros por segundo, replicando as condições reais de inferência.

A partir das distribuições de distância **intraclasse** (acertos) e **interclasse** (potenciais confusões), determinou-se a fronteira de decisão D^* como o ponto médio da janela que separa as duas distribuições; o valor de τ foi então definido de modo a mapear D^* ao limiar de confiança adotado. Esse procedimento, detalhado quantitativamente na Seção 5.4, constitui uma contribuição metodológica do trabalho para a calibração de classificadores dinâmicos baseados em DTW.

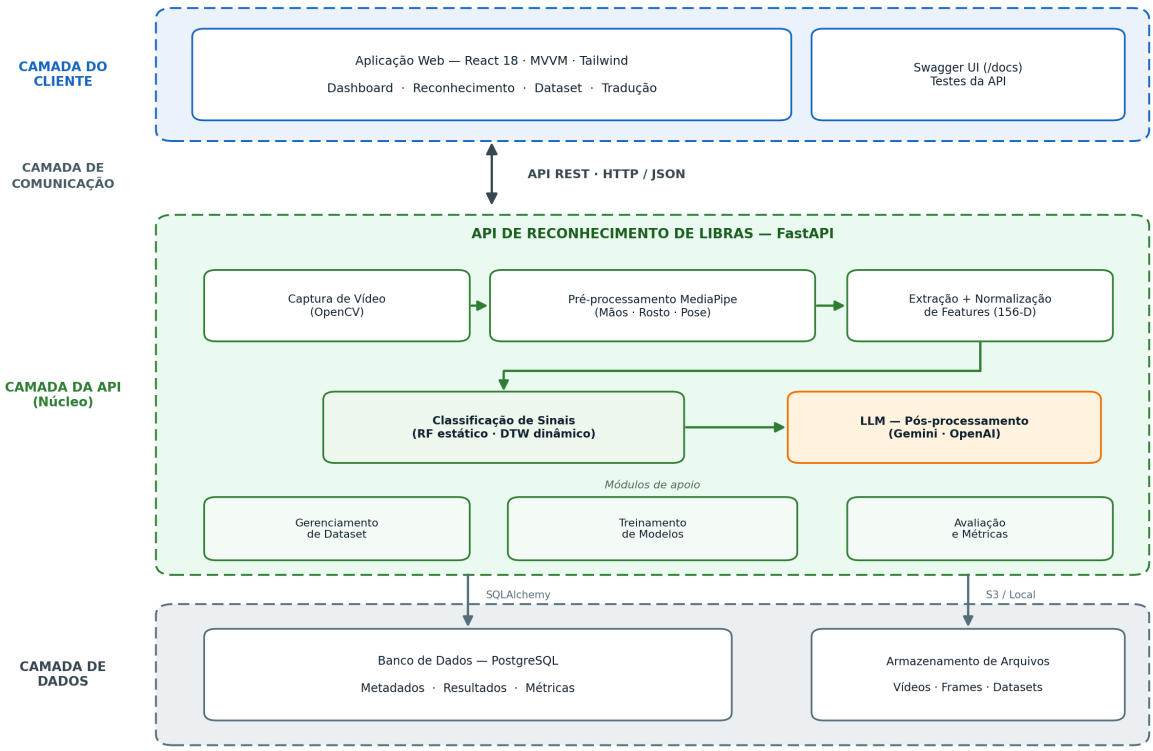
4 DESENVOLVIMENTO DA SOLUÇÃO

4.1 Visão Geral da Arquitetura

O LIBRAS-VC é um sistema de software composto por dois subsistemas principais: um *backend* responsável pelo processamento de vídeo, extração de *features*, classificação de sinais e tradução linguística; e um *frontend* responsável pela interface de usuário. A comunicação entre os dois subsistemas ocorre por meio de uma API REST com protocolo HTTP e formato de dados JSON, garantindo independência entre as camadas e facilitando a manutenção e a evolução do sistema. A Figura 8 apresenta a visão geral dessa arquitetura.

O *backend* é desenvolvido em Python 3.10 e estruturado em módulos coesos: `app/capture` (captura de vídeo com OpenCV), `app/features` (extração de *features* de mãos, rosto e pose), `app/preprocessing` (normalização e combinação de *features*), `app/recognition` (classificadores estático e dinâmico), `app/training` (treinamento dos modelos), `app/translation` (módulo de tradução via LLM) e `app/routes` (definição dos *endpoints* REST). O *frontend* é desenvolvido em React 18.2 com Vite como ferramenta de *build* e Tailwind CSS para estilização, adotando o padrão arquitetural MVVM (*Model-View-ViewModel*).

Figura 8 – Arquitetura Geral do Sistema LIBRAS-VC



Fonte: Elaborado pelo autor.

4.2 Módulo de Visão Computacional e Extração de *Features*

O módulo de visão computacional é responsável por processar os quadros de vídeo capturados pela câmera e extrair as representações numéricas dos sinais LIBRAS. O processamento de cada quadro segue um pipeline sequencial composto pelas seguintes etapas:

(1) Captura de vídeo: o quadro de vídeo é capturado pela câmera e convertido do espaço de cor BGR (padrão do OpenCV) para RGB, formato requerido pelo MediaPipe.

(2) Detecção de *landmarks*: o quadro é processado pelo MediaPipe Hands, pelo MediaPipe Face Detection e pelo MediaPipe Pose, que detectam, respectivamente, os *landmarks* das mãos, a região do rosto e os *landmarks* corporais.

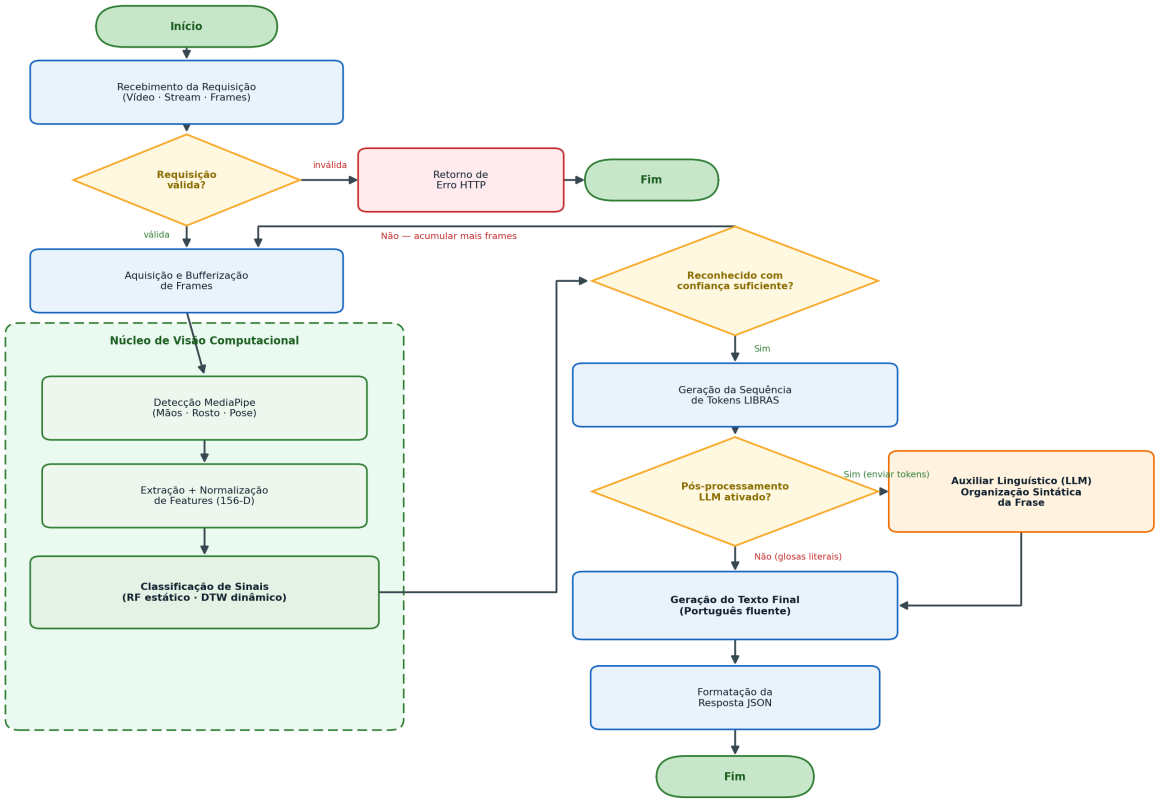
(3) Extração e normalização de *features*: os *landmarks* detectados são extraídos e normalizados, compondo um vetor de características de dimensão fixa de 156 valores por quadro.

(4) Classificação: o vetor de *features* é encaminhado ao classificador correspondente (estático ou dinâmico) para a inferência do sinal.

A Figura 9 apresenta o fluxo completo de execução da aplicação, desde o

recebimento da requisição até a resposta traduzida.

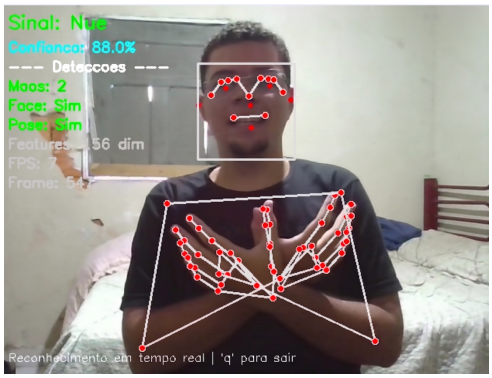
Figura 9 – Fluxo de execução da aplicação: do recebimento da requisição à resposta traduzida



Fonte: Elaborado pelo autor.

A Figura 10 ilustra o resultado da etapa de detecção sobre um usuário real: os *landmarks* das mãos – com as conexões que formam o esqueleto de cada mão –, a região do rosto e os pontos do tronco superior são identificados e sobrepostos ao quadro de vídeo, compondo o vetor de 156 dimensões utilizado na classificação.

Figura 10 – Marcações (*landmarks*) detectadas sobre o usuário durante o reconhecimento



Fonte: Elaborado pelo autor.

4.3 Composição do Vetor de *Features*

O vetor de características (*feature vector*) é a representação numérica de cada quadro de vídeo utilizada pelos classificadores. Optou-se por um vetor de dimensão fixa de 156 valores, garantindo compatibilidade com todos os classificadores e facilitando o armazenamento em formato CSV para análise e reuso. A composição do vetor é apresentada na Tabela 2 e ilustrada graficamente na Figura 11.

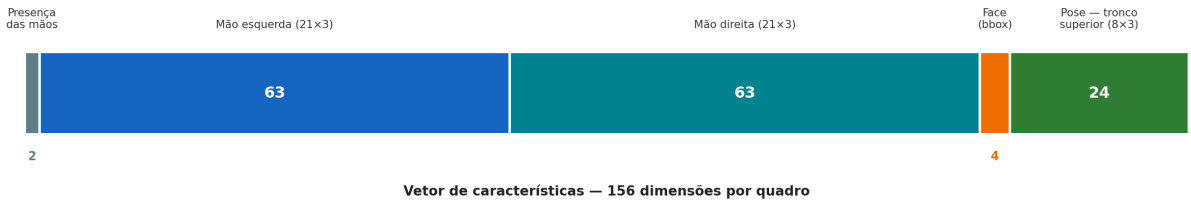
Tabela 2 – Composição do vetor de características (156 dimensões)

Componente	***Landmarks***	**Coords.**	**Dims.**	**Descrição**
Presença das mãos	–	2	2	Indicadores binários de presença da mão esquerda e direita
Mão esquerda	21	3	63	x, y, z normalizados de cada ponto articular
Mão direita	21	3	63	x, y, z normalizados de cada ponto articular
Rosto (*bbox*)	1	4	4	Centro (c_x, c_y), tamanho e confiança da detecção
Pose (tronco superior)	8	3	24	Ombros, cotovelos, pulsos e quadris
Total	**51**	–	**156**	–

Fonte: Elaborado pelo autor.

A normalização dos *landmarks* é realizada pelo método *centered*: as coordenadas de cada mão são centradas pela média dos 21 pontos e escalonadas pela

Figura 11 – Composição do vetor de características de 156 dimensões por quadro



Fonte: Elaborado pelo autor.

distância máxima, tornando a representação invariante à posição e à escala da mão na imagem. Para a pose, são selecionados apenas 8 dos 33 *landmarks* disponíveis – ombros (índices 11 e 12), cotovelos (13 e 14), pulsos (15 e 16) e quadris (23 e 24) –, descartando os *landmarks* de membros inferiores, irrelevantes para o reconhecimento de LIBRAS. Essa seleção reduziu o vetor de pose de 99 para 24 dimensões, resultando em uma redução total do vetor de 231 para 156 dimensões (redução de 32,5%), sem perda de informação relevante para o reconhecimento de sinais.

4.4 Modelos de Reconhecimento de Sinais

O sistema implementa dois classificadores independentes, selecionados automaticamente por um módulo de roteamento que analisa o nível de movimento nos quadros recentes do *buffer* dinâmico.

Classificador estático (Random Forest): utilizado para sinais executados sem movimento significativo, como a maior parte das letras do alfabeto em LIBRAS. O modelo é treinado com o vetor de 156 *features* extraído do quadro central de cada vídeo de treinamento. Os hiperparâmetros adotados foram: 100 árvores de decisão, profundidade máxima de 20 níveis e critério de impureza *Gini*. Um *StandardScaler* é aplicado às *features* antes do treinamento para garantir o balanceamento das escalas. O classificador rejeita predições com confiança inferior a 80%, limitando a ocorrência de falsos positivos.

Classificador dinâmico (DTW): utilizado para sinais que envolvem movimento, como a letra J do alfabeto LIBRAS (que exige o traçado de um J no espaço). O modelo opera por comparação de sequências: a sequência de *features* do sinal a ser reconhecido é comparada com os *templates* armazenados de cada classe por meio da distância DTW normalizada. A classe cujo *template* apresenta menor distância é retornada como resultado, com confiança calculada por $conf = e^{-d/\tau}$, onde d é a distância DTW normalizada e τ é um parâmetro de escala. O valor de τ foi calibrado empiricamente em $\tau = 3{,}0$ (Seção 5.4), de modo a alinhar a confiança dos acertos ao limiar de aceitação, fixado em 50%. O *buffer* dinâmico acumula entre 15 e 90 quadros

por sinal, com detecção automática de pausa pelo limiar de movimento calibrado (*motion threshold* = 0,326).

4.5 Módulo de Tradução Linguística

O módulo de tradução recebe como entrada uma sequência de tokens reconhecidos pelo sistema (por exemplo: `["EU", "ESTUDAR", "LIBRAS"]`) e produz como saída uma frase em português natural (por exemplo: *"Eu estudo LIBRAS"*). O processo de tradução é composto por duas etapas principais.

Na **etapa de pré-processamento**, letras individuais sequenciais são combinadas em palavras (por exemplo: `["L", "I", "B", "R", "A", "S"]` é convertido em `"LIBRAS"`), e os tokens são preparados para envio ao modelo de linguagem.

Na **etapa de pós-processamento via LLM**, os tokens pré-processados são enviados a um modelo de linguagem de grande escala, como o Google Gemini ou o GPT-4o-mini da OpenAI (Anil et al., 2023), que realiza a conversão para o português natural com conjugação verbal correta e concordância de gênero e número. O sistema inclui um mecanismo de *fallback*: caso o serviço LLM esteja indisponível, os tokens são simplesmente concatenados com capitalização da primeira letra, garantindo a continuidade do serviço. Importante ressaltar que os dados visuais do usuário não são enviados ao serviço LLM em nenhuma circunstância, apenas os tokens textuais reconhecidos, preservando a privacidade do usuário.

Um exemplo real desse processo é apresentado na Figura 13 (Seção 4.6), no qual o token reconhecido (`Nue`, um dos sinais experimentais do *dataset*) é convertido pelo LLM na palavra correspondente em português (*Pássaro*), acompanhado das métricas de confiança e de qualidade da tradução.

4.6 Interface Web

O *frontend* do LIBRAS-VC foi desenvolvido em React 18.2 (Meta Platforms; Inc., 2013) com Vite como ferramenta de *build* e Tailwind CSS para estilização, adotando o padrão arquitetural MVVM. A interface é composta por quatro telas principais, acessíveis por meio de um menu de navegação lateral.

A **tela de *Dashboard*** apresenta o status dos subsistemas (API, reconhecimento estático, reconhecimento dinâmico, tradução e modelo LLM) por meio de cartões com indicadores visuais coloridos (verde para operacional, amarelo para parcialmente disponível, vermelho para indisponível).

A **tela de Reconhecimento** exibe o vídeo da câmera em tempo real, com os *landmarks* desenhados sobre as mãos, rosto e corpo do usuário. O resultado do reconhecimento é exibido ao lado, com o token reconhecido, o nível de confiança e a sequência acumulada de sinais. A tela permite selecionar entre os modos estático

e dinâmico, e acionar a tradução automática da sequência acumulada. A Figura 12 apresenta essa interface em operação.

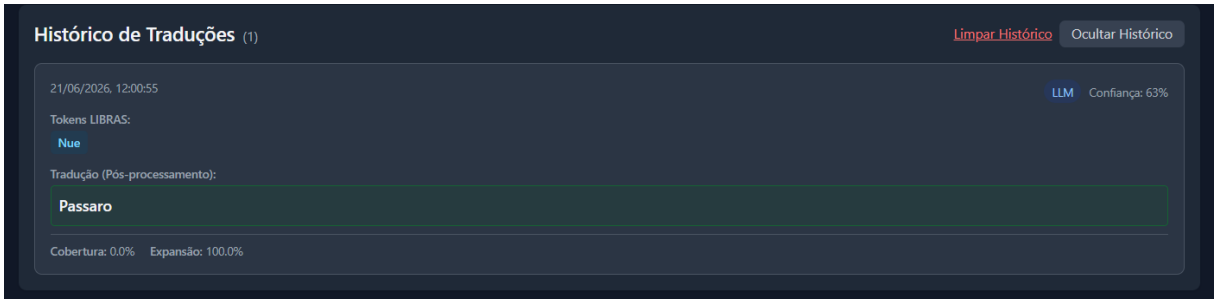
A **tela de *Dataset*** permite ao usuário criar classes de sinais, fazer o *upload* de vídeos de treinamento e visualizar as amostras organizadas por classe e tipo (estático ou dinâmico). A tela apresenta estatísticas do *dataset* e permite a exclusão de amostras individuais, conforme ilustra a Figura 14.

A **tela de Tradução** permite ao usuário inserir manualmente uma sequência de tokens LIBRAS e obter a tradução para o português, com métricas de qualidade da tradução e comparação entre o resultado com e sem LLM.

Figura 12 – Tela de Reconhecimento em Tempo Real do LIBRAS-VC

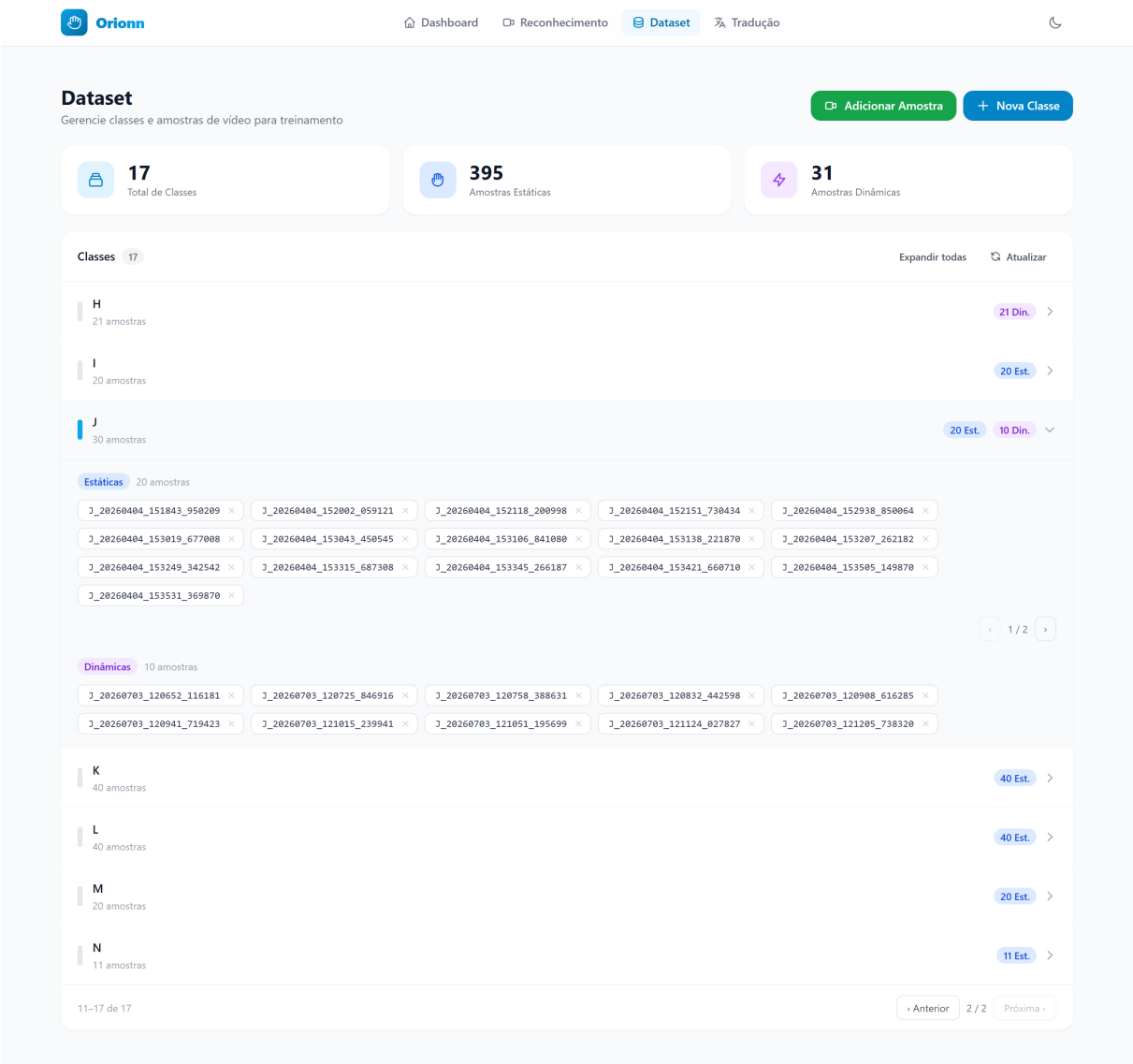
Fonte: Elaborado pelo autor.

Figura 13 – Exemplo de tradução do módulo linguístico: token LIBRAS reconhecido convertido para português pelo LLM



Fonte: Elaborado pelo autor.

Figura 14 – Tela de Gerenciamento de *Dataset*



Fonte: Elaborado pelo autor.

4.7 Tecnologias Utilizadas

A Tabela 3 apresenta as principais tecnologias empregadas no desenvolvimento do LIBRAS-VC.

Tabela 3 – Principais tecnologias utilizadas no sistema LIBRAS-VC

Componente	**Tecnologia**	**Versão**	**Finalidade**
Linguagem (*back*)	Python	3.10+	Lógica do servidor e ML
Framework web	FastAPI	0.110+	API REST assíncrona
Visão computacional	MediaPipe	0.10.21	Detecção de *landmarks*
Processamento de imagem	OpenCV	4.x	Captura e manipulação de vídeo
ML – classificação	Scikit-learn	1.4+	Random Forest e SVM
Banco de dados	PostgreSQL	12+	Persistência de dados
Linguagem (*front*)	JavaScript (JSX)	ES2022	Interface de usuário
Framework UI	React	18.2.0	Componentes reativos
Ferramenta de *build*	Vite	5.0.8	Empacotamento do *front*
Estilização	Tailwind CSS	3.3.6	*Design* responsivo
HTTP Client	Axios	1.6.2	Chamadas à API REST
LLM (*default*)	Google Gemini	Flash	Pós-processamento NLP
LLM (alternativo)	OpenAI GPT-4o-mini	–	Pós-processamento NLP

Fonte: Elaborado pelo autor.

4.8 Requisitos do Sistema

Os requisitos funcionais e não funcionais do LIBRAS-VC foram levantados com base na análise do problema e nos objetivos definidos na Atividade 2 da DSR. O Quadro 1 apresenta os requisitos funcionais e a Tabela 4 apresenta os requisitos não funcionais.

4.9 Casos de Uso

A Figura 15 apresenta o diagrama de casos de uso do sistema, enquanto o Quadro 2 sintetiza cada um deles, identificando os atores envolvidos e os resultados esperados em cada interação.

Tabela 4 – Requisitos não funcionais do sistema LIBRAS-VC

ID	Requisito	Métrica de aceitação	Categoria
RNF01	Processamento de quadro em tempo real	Latência $\leq$ 200 ms/quadro	Desempenho
RNF02	Interface acessível por navegador *web*	Compatível com Chrome/Firefox	Portabilidade
RNF03	Não transmitir dados visuais a serviços externos	Apenas tokens textuais ao LLM	Privacidade
RNF04	Código modular e documentado	Módulos com responsabilidade única	Manutenibilidade
RNF05	Interface intuitiva sem treinamento	Fluxo em no máximo 3 cliques	Usabilidade
RNF06	Funcionar sem LLM externo	*Fallback* automático ativado	Disponibilidade

Fonte: Elaborado pelo autor.

Figura 15 – Diagrama de casos de uso do sistema LIBRAS-VC

Fonte: Elaborado pelo autor.

Quadro 1 – Requisitos funcionais do sistema LIBRAS-VC

ID	Requisito	Prioridade	Módulo
RF01	Reconhecer sinais LIBRAS estáticos em tempo real via câmera	Alta	Reconhecimento
RF02	Reconhecer sinais LIBRAS dinâmicos por sequência de quadros	Alta	Reconhecimento
RF03	Traduzir sequências de tokens LIBRAS para português natural	Alta	Tradução
RF04	Permitir criação e gerenciamento de classes no *dataset*	Alta	*Dataset*
RF05	Permitir *upload* de vídeos de treinamento por classe	Alta	*Dataset*
RF06	Exibir status dos subsistemas no *dashboard*	Média	*Dashboard*
RF07	Manter histórico de sinais reconhecidos na sessão	Média	Reconhecimento
RF08	Exportar *features* do *dataset* em formato CSV	Média	*Dataset*
RF09	Detectar automaticamente o tipo de sinal (estático/dinâmico)	Média	Reconhecimento
RF10	Suportar *fallback* de tradução sem LLM	Baixa	Tradução

Fonte: Elaborado pelo autor.

5 RESULTADOS E DISCUSSÃO

5.1 Requisitos Computacionais e de Rede

Por ser uma aplicação web com processamento centralizado no servidor, o LIBRAS-VC impõe requisitos assimétricos entre cliente e servidor. No cliente, a exigência é mínima: basta um navegador moderno com suporte à API de captura de mídia (*getUserMedia*), uma *webcam* e conexão com a internet, dispensando instalação de *software* ou hardware especializado. O navegador é responsável apenas por capturar os quadros de vídeo (640×480 a 30 quadros por segundo) e exibir os resultados, tarefas leves que executam de forma fluida mesmo em equipamentos modestos.

A carga computacional concentra-se no servidor, que, a cada quadro recebido, realiza a decodificação da imagem, a extração de landmarks pelo MediaPipe (*Hands*, *Face Detection* e *Pose*) e a classificação pelos modelos Random Forest e DTW, além do pós-processamento linguístico. Como o *pipeline* de visão computacional opera sobre CPU – com o MediaPipe *Hands* configurado em *model_complexity* = 0 para priorizar

Quadro 2 – Casos de uso principais do sistema LIBRAS-VC

UC	**Caso de Uso**	**Ator**	**Resultado Esperado**
UC01	Reconhecer sinal estático	Usuário	Token LIBRAS exibido com nível de confiança
UC02	Reconhecer sinal dinâmico	Usuário	Token LIBRAS exibido após detecção de movimento
UC03	Traduzir sequência de tokens	Usuário	Frase em português natural gerada
UC04	Criar classe de sinal	Administrador	Nova classe adicionada ao *dataset*
UC05	Adicionar amostra de vídeo	Administrador	Vídeo armazenado e associado à classe
UC06	Visualizar status do sistema	Usuário	*Dashboard* com indicadores de saúde dos subsistemas
UC07	Exportar *dataset* em CSV	Administrador	Arquivo CSV com *features* extraídas para análise externa

Fonte: Elaborado pelo autor.

a velocidade –, não há exigência de GPU dedicada, mas o servidor demanda um processador com desempenho suficiente para manter o tempo de processamento por quadro abaixo do limiar de 200 ms definido no RNF01. Ressalta-se que os quadros são processados sequencialmente por uma instância compartilhada do MediaPipe, de modo que a capacidade de atendimento a usuários simultâneos é limitada pela CPU do servidor, característica a ser considerada em cenários de múltiplos acessos concorrentes.

Quanto à rede, a arquitetura transmite cada quadro de vídeo do cliente ao servidor por meio de requisições HTTP e recebe em resposta os landmarks e o sinal reconhecido. Para equilibrar responsividade e volume de tráfego, a inferência é acionada a aproximadamente 4 quadros por segundo (intervalo de 250 ms entre envios), e não à taxa plena de captura. Essa abordagem exige uma conexão estável e de baixa latência, uma vez que o atraso de rede soma-se ao orçamento de processamento por quadro; conexões lentas ou servidores geograficamente distantes podem, assim, comprometer a percepção de tempo. Na etapa de tradução, por fim, apenas os *tokens* textuais reconhecidos são enviados ao modelo de linguagem – e não os dados visuais do usuário –, o que reduz o volume trafegado e preserva a privacidade.

5.2 Dataset Construído

O *dataset* do LIBRAS-VC foi construído de forma incremental ao longo do desenvolvimento do sistema, utilizando a própria plataforma como ferramenta de coleta. Os vídeos foram gravados em resolução 640×480 pixels, a 30 quadros por segundo, com duração padrão de 20 segundos por amostra – intervalo suficiente para a captura estável do sinal, do qual se extrai o quadro central, no caso dos sinais estáticos, ou a subsequência de movimento, no caso dos dinâmicos. A Tabela 5 apresenta a distribuição de amostras por classe no *dataset* construído.

Tabela 5 – Distribuição de amostras por classe no *dataset* LIBRAS-VC

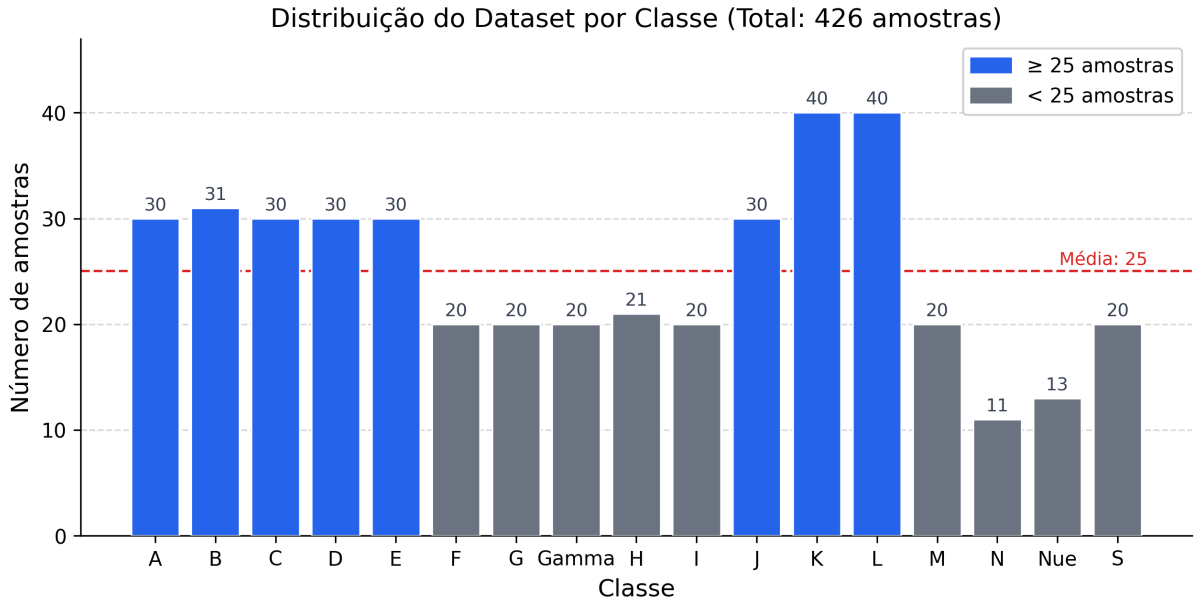
Classe	**Est.**	**Din.**	**Total**	**Observação**
A	30	–	30	–
B	31	–	31	–
C	30	–	30	–
D	30	–	30	–
E	30	–	30	–
F	20	–	20	–
G	20	–	20	–
H	–	21	21	Sinal dinâmico
I	20	–	20	–
J	20	10	30	Estático e dinâmico
K	40	–	40	–
L	40	–	40	–
M	20	–	20	–
N	11	–	11	–
S	20	–	20	–
Gamma (Γ)	20	–	20	Sinal experimental
Nue	13	–	13	Sinal experimental (menor volume)
Total	**395**	**31**	**426**	17 classes

Fonte: Elaborado pelo autor.

O *dataset* totaliza 426 amostras distribuídas em 17 classes, sendo 395 amostras de sinais estáticos e 31 de sinais dinâmicos (classes H e J). Observa-se desequilíbrio moderado entre as classes estáticas, com a mais populosa (L, com 40 amostras) tendo mais de três vezes o volume da menos populosa (N, com 11 amostras); esse desequilíbrio é um fator de atenção para o treinamento, uma vez que classificadores tendem a privilegiar classes com maior volume de amostras. O subconjunto dinâmico, ainda restrito a duas classes (H, com 21 amostras, e J, com 10), foi construído especificamente para viabilizar o treinamento e a avaliação quantitativa do classificador

DTW, apresentada na Seção 5.4. Ressalta-se que as classes `Gamma` (Γ) e `Nue` são sinais experimentais, criados pelo autor durante os testes iniciais do sistema e mantidos no *dataset*; não pertencem ao alfabeto nem ao vocabulário oficial da LIBRAS, servindo apenas para validar o funcionamento do pipeline de reconhecimento. A Figura 16 ilustra essa distribuição.

Figura 16 – Distribuição do *Dataset* por Classe



Fonte: Elaborado pelo autor.

Diferentemente de versões anteriores do *dataset*, os sinais que envolvem movimento já foram registrados com a rotulagem adequada: a classe J – que na gramática da LIBRAS exige o traçado de um "J" no espaço – passou a contar com 10 amostras dinâmicas, além das estáticas, e a classe H foi construída integralmente como dinâmica (21 amostras). Essa correção viabilizou o treinamento e a avaliação quantitativa do classificador DTW, apresentados na Seção 5.4.

5.3 Avaliação do Sistema

Desempenho do classificador estático: O modelo Random Forest foi treinado e avaliado sobre o subconjunto de 395 amostras estáticas (16 classes), utilizando o quadro central de cada vídeo como representação; os 31 vídeos dinâmicos (classes H e J) são avaliados separadamente pelo classificador DTW (Seção 5.4). Adotou-se divisão estratificada de 80% para treino e 20% para validação, cujas métricas agregadas são apresentadas na Tabela 6. Para uma análise mais robusta por classe – dado o pequeno número de amostras de validação de algumas classes –, as métricas por classe (Tabela 7) e a matriz de confusão (Figura 17) foram calculadas por validação cruzada

estratificada de 5 dobras. A importância das *features* por componente é apresentada na Figura 18 e a curva de aprendizado na Figura 19. Observa-se desempenho elevado na maioria das classes, com as menores pontuações em S (F1 de 0,84) e A (F1 de 0,89) – classes que apresentam confusão mútua, conforme a Figura 17. Ressalta-se que os valores agregados da Tabela 6 (protocolo 80/20) e os da Tabela 7 (validação cruzada 5-*fold*) decorrem de protocolos distintos, o que explica pequenas diferenças entre eles – como a precisão média ponderada de 0,99 e 0,98, respectivamente.

Tabela 6 – Métricas de desempenho do classificador estático (Random Forest)

Métrica	**Treino**	**Validação**	**Observação**
Acurácia	1,00	0,99	Validação estratificada 80/20
Precisão (média ponderada)	–	0,99	–
Revocação (média ponderada)	–	0,99	–
F1-*score* (média ponderada)	–	0,99	–

Fonte: Elaborado pelo autor a partir dos experimentos realizados.

Avaliação qualitativa da interface: A interface do LIBRAS-VC foi avaliada com base nos critérios de acessibilidade estabelecidos pela WCAG 2.1 (W3C, 2018) e nos princípios de usabilidade de Preece, Rogers e Sharp (2013). A análise identificou que o sistema atende aos critérios de *perceptibilidade* (contraste de cores adequado, feedback visual do reconhecimento), *operabilidade* (fluxo de reconhecimento em no máximo 3 interações), *compreensibilidade* (terminologia familiar ao contexto LIBRAS) e *robustez* (funcionamento em múltiplos navegadores modernos). A integração do mecanismo de *fallback* para a tradução sem LLM garante a disponibilidade contínua das funcionalidades essenciais do sistema.

Avaliação de latência: O pipeline de processamento de um único quadro – incluindo conversão de cor, detecção pelo MediaPipe (Hands, Face Detection e Pose) e extração do vetor de *features* – opera abaixo do limiar de 200 ms por quadro estabelecido no RNF01, mantendo a percepção de tempo real pelo usuário. Cabe distinguir as taxas envolvidas: a câmera captura o vídeo a 30 quadros por segundo, ao passo que a inferência de reconhecimento é acionada em uma taxa reduzida – de aproximadamente 4 quadros por segundo –, suficiente para acompanhar a execução dos sinais sem sobrecarregar o servidor; o orçamento de 200 ms refere-se ao tempo máximo de processamento de cada quadro submetido. Essa eficiência é obtida pelo uso do modelo de baixa complexidade do MediaPipe Hands (*model_complexity* = 0), que prioriza a velocidade de inferência em detrimento de uma margem de precisão adicional.

Tabela 7 – Métricas por classe do classificador estático (Random Forest, validação cruzada 5-*fold*)

Classe	**Precisão**	**Revocação**	**F1-score**	**Amostras**
A	0,96	0,83	0,89	30
B	0,97	1,00	0,98	31
C	1,00	0,97	0,98	30
D	1,00	0,97	0,98	30
E	0,97	0,97	0,97	30
F	1,00	0,95	0,97	20
G	1,00	1,00	1,00	20
Gamma (Γ)	1,00	1,00	1,00	20
I	1,00	1,00	1,00	20
J	0,95	1,00	0,98	20
K	1,00	1,00	1,00	40
L	1,00	1,00	1,00	40
M	0,95	1,00	0,98	20
N	1,00	0,91	0,95	11
Nue	1,00	1,00	1,00	13
S	0,76	0,95	0,84	20
Média ponderada	**0,98**	**0,97**	**0,97**	**395**
Acurácia global		**0,97**		

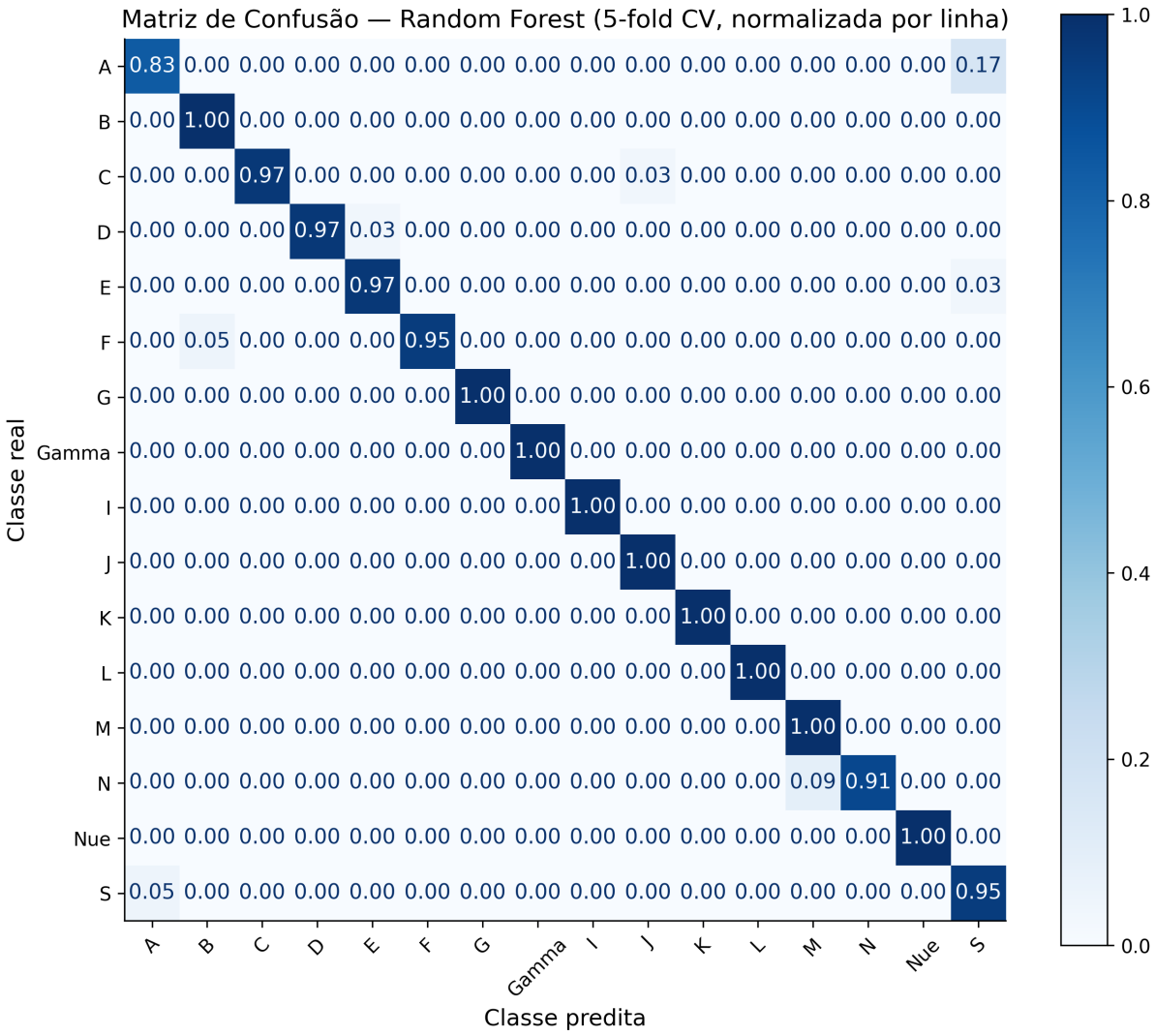
Fonte: Elaborado pelo autor a partir dos experimentos realizados.

5.4 Avaliação do Classificador Dinâmico e Calibração do Limiar

A avaliação do classificador dinâmico (DTW) requer um protocolo distinto do adotado para o classificador estático, uma vez que o algoritmo utiliza os próprios vídeos de treinamento como *templates* de referência. Para evitar o vício de avaliação (*data leakage*) decorrente de comparar uma amostra com o *template* dela mesma, adotou-se a validação *leave-one-out*: cada uma das 31 amostras dinâmicas (21 da classe H e 10 da classe J) foi classificada por vizinho mais próximo (*nearest-neighbor*) contra todos os demais *templates*, excluindo o seu próprio. Adicionalmente, cada amostra foi reprocessada a aproximadamente 4 quadros por segundo, replicando as condições reais de inferência – nas quais o *buffer* acumula sequências curtas –, enquanto os *templates* foram mantidos na taxa original de 30 quadros por segundo.

A avaliação revelou **separação perfeita entre as duas classes dinâmicas**: em 100% dos casos, o *template* mais próximo pertencia à classe correta, sem qualquer confusão entre H e J. A Tabela 8 sintetiza as métricas obtidas e a Figura 20 ilustra a separação das distâncias e o mapeamento para a confiança.

Figura 17 – Matriz de confusão do classificador estático (Random Forest, validação cruzada 5-*fold*, normalizada por linha)



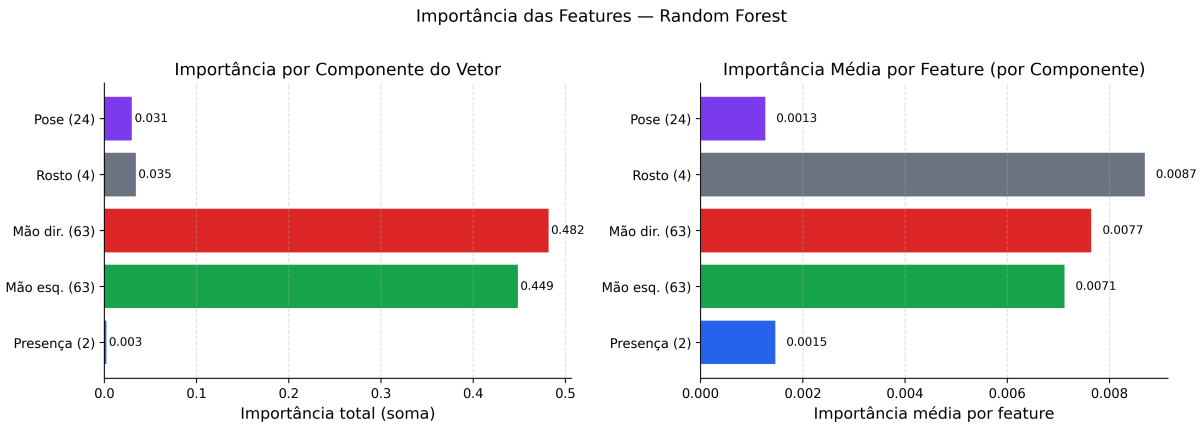
Fonte: Elaborado pelo autor.

Tabela 8 – Métricas de avaliação do classificador dinâmico (DTW, *leave-one-out*)

Métrica	**Valor**
Amostras dinâmicas avaliadas	31 (H: 21; J: 10)
Acurácia de classificação (H *vs.* J)	100%
Distância intra-classe (mediana / máximo)	0,94 / 1,59
Distância inter-classe (mínimo)	2,58
Fronteira de decisão D^*	2,08
Escala de confiança calibrada τ	3,0
Limiar de aceitação	0,50

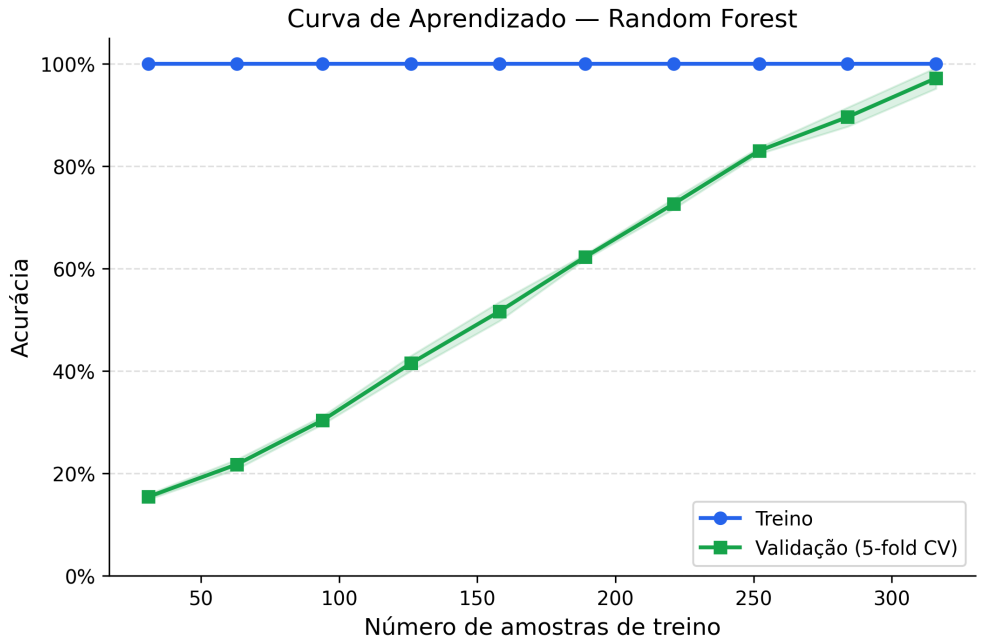
Fonte: Elaborado pelo autor a partir dos experimentos realizados.

Figura 18 – Importância das *Features* por Componente do Vetor — Random Forest



Fonte: Elaborado pelo autor.

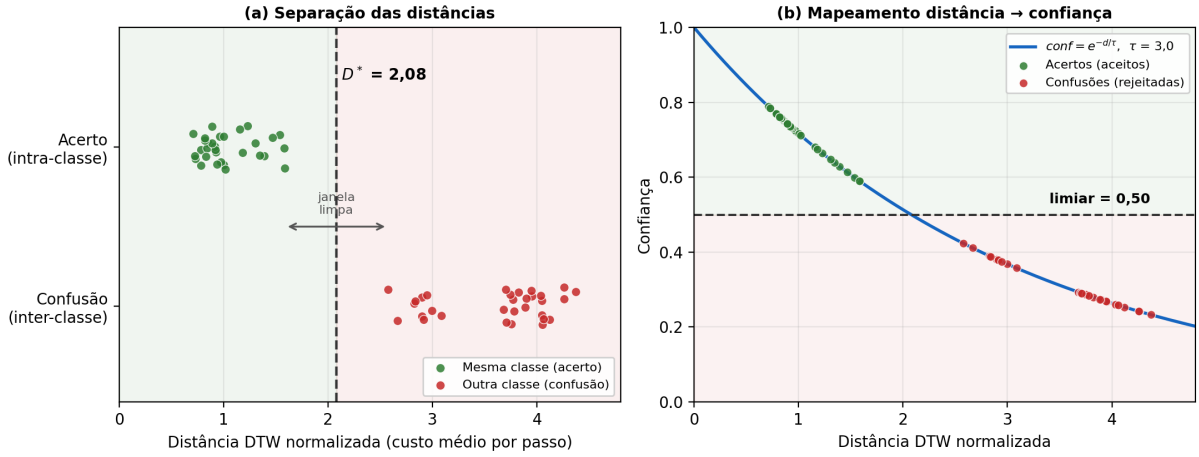
Figura 19 – Curva de Aprendizado do Classificador Estático (Random Forest, 5-*fold* CV)



Fonte: Elaborado pelo autor.

Um aspecto central da avaliação foi a **calibração do limiar de decisão**. A confiança do classificador é obtida pela transformação $conf = e^{-d/\tau}$, na qual a escala τ determina como as distâncias DTW são mapeadas para o intervalo $[0, 1]$. A distância intra-classe mediana observada foi de 0,94 (máximo de 1,59), enquanto a distância inter-classe mínima foi de 2,58, configurando uma *janela limpa* de separação no intervalo $[1{,}59;\ 2{,}58]$, cujo ponto médio define a fronteira de decisão $D^* = 2{,}08$. Com o valor de escala originalmente adotado ($\tau = 0{,}5$), mesmo os acertos produziam confiança inferior a 0,20 – muito abaixo do limiar então vigente de 0,80 –; como consequência, o classificador dinâmico não retornava sinal algum, ainda que a classe correta fosse corretamente identificada pelo alinhamento DTW. A recalibração de τ para 3,0 mapeia a fronteira D^* ao limiar de 0,50: os acertos passam a apresentar confiança entre 0,59 e 0,73 (aceitos), ao passo que as confusões potenciais permanecem em no máximo 0,42 (rejeitadas), com margem de separação confortável, conforme ilustrado na Figura 20(b).

Figura 20 – Calibração do Classificador Dinâmico: separação das distâncias DTW (a) e mapeamento distância–confiança (b)



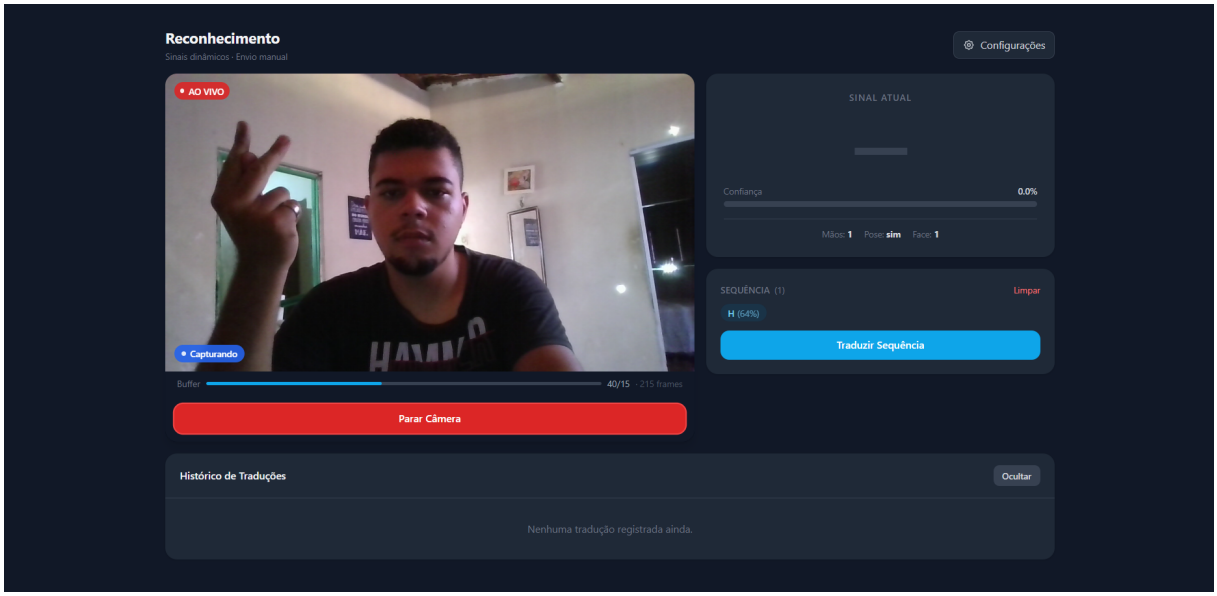
Fonte: Elaborado pelo autor.

Ressalta-se que essa calibração foi conduzida sobre o *dataset* reprocessado em condições próximas às de inferência, constituindo uma estimativa robusta da fronteira de decisão; a validação com capturas ao vivo de múltiplos participantes permanece como etapa recomendada para trabalhos futuros. A separação perfeita observada evidencia, contudo, que a representação de *features* adotada é suficientemente discriminativa para os sinais dinâmicos avaliados e que o principal fator limitante do reconhecimento dinâmico não residia na capacidade de distinção entre classes, mas sim na calibração da camada de decisão.

A Figura 21 demonstra o reconhecimento dinâmico em funcionamento na interface do sistema: enquanto o *buffer* temporal acumula os quadros do gesto (indicador “Buffer”), o sinal dinâmico H é reconhecido e adicionado à sequência com

64% de confiança – valor coerente com a faixa de 0,59 a 0,73 prevista para os acertos do classificador dinâmico após a calibração.

Figura 21 – Reconhecimento dinâmico em funcionamento: captura do gesto com o *buffer* temporal e o sinal H reconhecido pelo classificador DTW



Fonte: Elaborado pelo autor.

5.5 Discussão

Os resultados obtidos demonstram a viabilidade técnica da abordagem proposta para o reconhecimento e a tradução de sinais LIBRAS em ambiente web. A combinação de *landmarks* MediaPipe com o algoritmo Random Forest para sinais estáticos constitui uma solução eficiente e adequada ao tamanho atual do *dataset*: com cerca de 25 amostras por classe em média, métodos baseados em redes neurais profundas estariam sujeitos ao sobreajuste, enquanto o Random Forest oferece generalização satisfatória com esse volume de dados (Breiman, 2001).

A escolha do DTW para sinais dinâmicos é igualmente adequada ao estágio atual do projeto: o algoritmo não requer um grande volume de amostras por classe para operar de forma efetiva, pois utiliza os próprios vídeos de treinamento como *templates* de referência (Sakoe; Chiba, 1978). À medida que o *dataset* dinâmico for construído, o DTW permitirá o reconhecimento imediato de novos sinais sem necessidade de retreinamento.

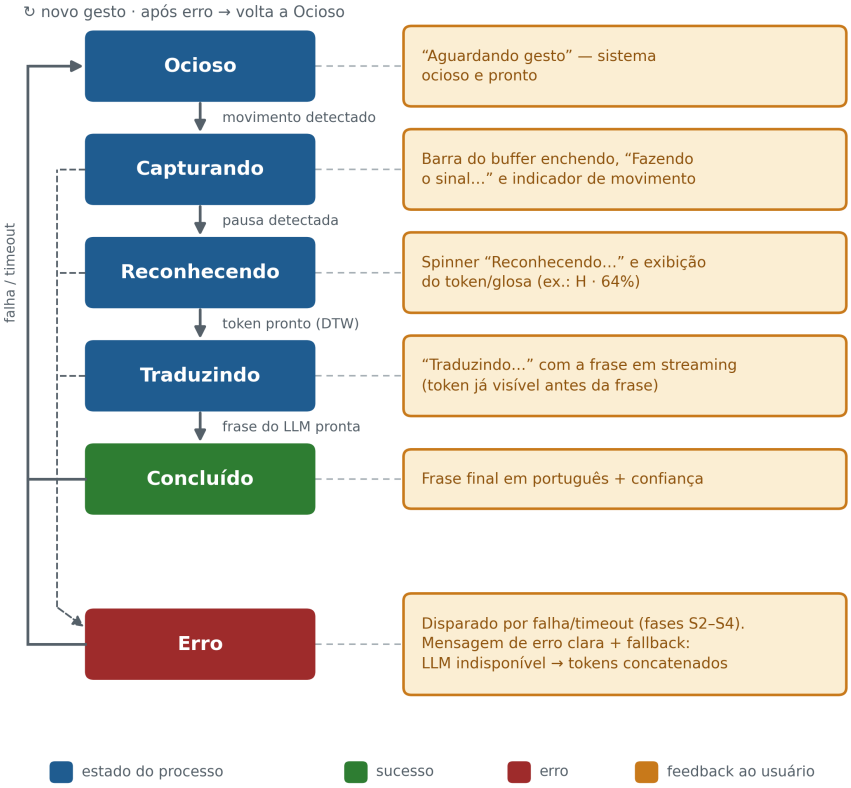
A integração do módulo LLM representa uma contribuição diferencial do LIBRAS-VC em relação aos trabalhos relacionados, uma vez que desloca a responsabilidade da geração de linguagem natural de uma camada de regras fixas para um modelo generativo capaz de conjugar verbos, ajustar concordâncias e produzir frases

contextualmente coerentes. A arquitetura de privacidade adotada – na qual apenas tokens textuais são enviados ao LLM, e não dados visuais do usuário – é uma decisão de projeto relevante do ponto de vista ético e legal, alinhada aos princípios da Lei Geral de Proteção de Dados (LGPD).

Entre as limitações identificadas, destacam-se: (i) o vocabulário dinâmico ainda restrito a duas classes (H e J) que, embora perfeitamente separáveis na avaliação realizada, não esgotam a variedade de sinais dinâmicos da LIBRAS e (ii) a dependência de uma câmera funcional e de boa iluminação para o reconhecimento adequado. Tais limitações constituem oportunidades de desenvolvimento para trabalhos futuros.

Para reduzir a percepção de espera durante o reconhecimento dinâmico, a interface do LIBRAS-VC estrutura o ciclo de reconhecimento como uma **máquina de estados** com pontos de *feedback* dedicados a cada fase, ilustrada na Figura 22. O mecanismo apoia-se em dois princípios: a **divulgação progressiva**, que exibe o *token*/glosa reconhecido pelo DTW imediatamente, antes da conclusão da tradução; e a **sinalização de estado**, que informa o usuário sobre a fase corrente por meio de indicadores visuais, em conformidade com o princípio de operabilidade da WCAG 2.1 (W3C, 2018). Cada estado do ciclo – ocioso, capturando (com barra de progresso do *buffer*), reconhecendo, traduzindo e concluído – possui um retorno visual associado, e as falhas são tratadas pelo mecanismo de *fallback*. Na etapa de tradução, a frase gerada pelo LLM é apresentada em ***streaming*** – token a token, por meio de *Server-Sent Events* –, reforçando a sensação de resposta imediata em vez de uma espera até o texto completo.

Figura 22 – Máquina de estados do reconhecimento dinâmico e os pontos de *feedback* ao usuário em cada fase



Fonte: Elaborado pelo autor.

6 CONSIDERAÇÕES FINAIS

Este trabalho apresentou o LIBRAS-VC, uma plataforma web para o reconhecimento e a tradução de sinais da Língua Brasileira de Sinais (LIBRAS) em tempo real, desenvolvida sob a metodologia Design Science Research (DSR). O sistema emprega a biblioteca MediaPipe para a extração de *landmarks* corporais, compondo um vetor de características de 156 dimensões por quadro de vídeo, sobre o qual operam um classificador Random Forest para sinais estáticos e um algoritmo DTW para sinais dinâmicos. Um módulo de pós-processamento baseado em LLM realiza a conversão das sequências de tokens reconhecidos para frases em português natural.

As principais contribuições deste trabalho podem ser sintetizadas em três dimensões. Do ponto de vista **social**, o sistema contribui para a redução das barreiras de comunicação enfrentadas pela comunidade surda brasileira, alinhando-se às diretrizes da Lei nº 10.436/2002 e da Lei Brasileira de Inclusão. Do ponto de vista **tecnológico**, a solução demonstra a viabilidade da construção de sistemas de reconhecimento de língua de sinais de código aberto, baseados em *landmarks* e

acessíveis via navegador *web*, sem dependência de *hardware* especializado. Do ponto de vista **científico**, a condução da pesquisa segundo a DSR produz conhecimento metodológico replicável sobre o processo de desenvolvimento, avaliação e comunicação de artefatos de tecnologia da informação voltados à acessibilidade.

Como perspectivas para trabalhos futuros, destacam-se: (i) a ampliação do *dataset* com um volume maior de amostras por classe e com diversidade de participantes (diferentes idades, gêneros e contextos de iluminação); (ii) a ampliação do vocabulário dinâmico para além das classes H e J já avaliadas, incluindo os demais sinais da LIBRAS que envolvem movimento; (iii) a realização de testes de usabilidade com usuários surdos e intérpretes de LIBRAS, para validar a efetividade do sistema na perspectiva dos usuários-alvo; (iv) a expansão do vocabulário reconhecido para além do alfabeto, incluindo palavras e frases de uso cotidiano; e (v) a avaliação de modelos neurais profundos (GRU, LSTM, *Transformers*) à medida que o *dataset* cresce e se torna suficientemente representativo para o treinamento dessas arquiteturas.

O LIBRAS-VC representa um passo concreto em direção à democratização da comunicação acessível no Brasil, demonstrando que a convergência entre visão computacional, aprendizado de máquina e modelos de linguagem pode gerar soluções tecnológicas socialmente relevantes, desenvolvidas no contexto da educação tecnológica pública.

REFERÊNCIAS

ANIL, R. et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. Disponível em: <https://arxiv.org/abs/2312.11805>. Acesso em: 20 jan. 2026.

BRASIL. **Lei nº 10.436, de 24 de abril de 2002**. Dispõe sobre a Língua Brasileira de Sinais – LIBRAS e dá outras providências. Brasília, DF, 2002. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/2002/l10436.htm>. Acesso em: 10 jan. 2026.

BRASIL. **Decreto nº 5.626, de 22 de dezembro de 2005**. Regulamenta a Lei nº 10.436, de 24 de abril de 2002. Brasília, DF, 2005. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2005/decreto/d5626.htm>. Acesso em: 10 jan. 2026.

BRASIL. **Lei nº 13.146, de 6 de julho de 2015**. Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência). Brasília, DF, 2015. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm>. Acesso em: 10 jan. 2026.

BREIMAN, L. Random forests. *Machine Learning*, v. 45, n. 1, p. 5–32, 2001.

BROWN, T. B. et al. Language models are few-shot learners. In: *Advances in Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2020.

CAMGOZ, N. C. et al. Neural sign language translation. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2018.

DIAS, D. B. et al. Hand movement recognition for Brazilian Sign Language: A study using distance-based neural networks. In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. Shenzhen, China: [s.n.], 2021.

GIL, A. C. **Métodos e técnicas de pesquisa social**. 7. ed. São Paulo: Atlas, 2019. 248 p.

HEVNER, A. R. et al. Design science in information systems research. *MIS Quarterly*, v. 28, n. 1, p. 75–105, 2004.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA (IBGE). **Censo Demográfico 2010**: características gerais da população, religião e pessoas com deficiência. Rio de Janeiro: IBGE, 2012. Disponível em: <https://censo2010.ibge.gov.br/resultados.html>. Acesso em: 15 jan. 2026.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *International Joint Conference on Artificial Intelligence (IJCAI)*. [S.l.: s.n.], 1995.

KOLLER, O. Quantitative survey of the state of the art in sign language recognition. *arXiv preprint arXiv:2008.09918*, 2020. Disponível em: <https://arxiv.org/abs/2008.09918>. Acesso em: 25 jan. 2026.

LUGARESI, C. et al. MediaPipe: A framework for perceiving and processing reality. In: *Third Workshop on Computer Vision for AR/VR at IEEE Winter Conference on Applications of Computer Vision*. Long Beach, CA: [s.n.], 2019.

MADEO, R. C. B. et al. Gesture unit segmentation using support vector machines: segmenting gestures from rest positions. *SAC '16: Proceedings of the 31st Annual ACM Symposium on Applied Computing*, p. 184–191, 2016.

META PLATFORMS, INC. *React*: A JavaScript library for building user interfaces. 2013. Disponível em: <https://react.dev/>. Acesso em: 20 jan. 2026.

MÜLLER, M. Dynamic time warping. In: *Information Retrieval for Music and Motion*. [S.l.]: Springer, 2007. p. 69–84.

ORGANIZAÇÃO DAS NAÇÕES UNIDAS (ONU). **Convenção sobre os Direitos das Pessoas com Deficiência**. Nova York, 2006. Disponível em: <https://www.un.org/disabilities/documents/convention/convoptprot-e.pdf>. Acesso em: 15 jan. 2026.

PAPASTRATIS, I. et al. Artificial intelligence technologies for sign language. *Sensors*, v. 21, n. 17, p. 5843, 2021.

PEFFERS, K. et al. A design science research methodology for information systems research. *Journal of Management Information Systems*, v. 24, n. 3, p. 45–77, 2007.

POWERS, D. M. W. Evaluation: From precision, recall and F-Measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, v. 2, n. 1, p. 37–63, 2011.

PREECE, J.; ROGERS, Y.; SHARP, H. **Design de interação**: além da interação humano-computador. 3. ed. Porto Alegre: Bookman, 2013. 585 p.

QUADROS, R. M. d.; KARNOPP, L. B. **Língua de sinais brasileira**: estudos linguísticos. Porto Alegre: Artmed, 2004. 221 p.

RASTGOO, R.; KIANI, K.; ESCALERA, S. Sign language recognition: A deep survey. *Expert Systems with Applications*, v. 164, p. 113794, 2021.

SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 26, n. 1, p. 43–49, 1978.

TRINDADE, P. R.; MACÊDO, D.; ZANCHETTIN, C. Reconhecimento de LIBRAS com MediaPipe e aprendizado de máquina: uma abordagem baseada em grafos de mãos. **Revista Brasileira de Computação Aplicada**, v. 14, n. 2, p. 45–57, 2022.

VASWANI, A. et al. Attention is all you need. In: *Advances in Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2017.

WORLD WIDE WEB CONSORTIUM (W3C). *Web Content Accessibility Guidelines (WCAG) 2.1*. 2018. Disponível em: <https://www.w3.org/TR/WCAG21/>. Acesso em: 20 jan. 2026.

ZHANG, F. et al. MediaPipe Hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020. Disponível em: <https://arxiv.org/abs/2006.10214>. Acesso em: 20 jan. 2026.