UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

PEDRO FERREIRA CALIMAN

DESENVOLVIMENTO DE UM SISTEMA DE RECONHECIMENTO DE SINAIS DO ALFABETO MANUAL DE LIBRAS UTILIZANDO MEDIAPIPE HANDS E REDE LSTM

BAURU
Novembro/2024

PEDRO FERREIRA CALIMAN

DESENVOLVIMENTO DE UM SISTEMA DE RECONHECIMENTO DE SINAIS DO ALFABETO MANUAL DE LIBRAS UTILIZANDO MEDIAPIPE HANDS E REDE LSTM

Trabalho de Conclusão do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. Antonio Carlos Sementille

BAURU
Novembro/2024

C153d

Caliman, Pedro Ferreira

Desenvolvimento de um sistema de reconhecimento de sinais do alfabeto manual de Libras utilizando MediaPipe Hands e rede LSTM / Pedro Ferreira Caliman. -- Bauru, 2024

44 p. : il., tabs., fotos

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru

Orientador: Antonio Carlos Sementille

1. Língua Brasileira de Sinais. 2. Rede LSTM. 3. Detecção de Pontos de Referência. 4. Reconhecimento de Linguagem de Sinais. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

Pedro Ferreira Caliman

Desenvolvimento de um sistema de reconhecimento de sinais do alfabeto manual de Libras utilizando MediaPipe Hands e rede LSTM

Trabalho de Conclusão do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Antonio Carlos Sementille
Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Profa. Dra. Simone das Graças Domingues Prado
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Profa. Dra. Juliana da Costa Feitosa
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Bauru, 14 de Novembro de 2024.

Resumo

A Língua Brasileira de Sinais (Libras) é uma língua visual que, assim como muitas outras linguagens de sinais, ainda não possui uma infraestrutura suficientemente desenvolvida. Isso é algo que se torna ainda mais agravante quando considerada a sua importância como meio de comunicação para a comunidade surda. Por muito tempo, a análise dos sinais de Libras tem sido uma tarefa difícil para os computadores, em parte devido às dependências espaciais-temporais envolvidas no reconhecimento de gestos que utilizam a movimentação das mãos. Porém, os avanços recentes nas tecnologias de Visão Computacional e Aprendizado de Máquina permitem cada vez mais a superação destes obstáculos. Portanto, o objetivo deste trabalho consiste na implementação de um *pipeline* completo para o reconhecimento dos sinais do alfabeto manual de Libras, incluindo a coleta de fontes para a elaboração do conjunto de dados, a extração dos pontos de referência das mãos, e o treinamento de uma Rede *Long Short-Term Memory* capaz de analisar os sinais individuais.

Palavras-chave: Língua Brasileira de Sinais; Rede LSTM; Detecção de Pontos de Referência; Reconhecimento de Linguagem de Sinais.

Abstract

Brazilian Sign Language (Libras) is a visual language that, much like many other sign languages, still does not possess a sufficiently developed infrastructure to support it. This is further aggravated when considering its importance as a means of communication for the deaf community. For a long time, the analysis of signs in Libras has been a difficult task for computers, in part due to the spatial-temporal dependencies involved in the recognition of gestures that utilize hand movements. Recently, however, advances in Computer Vision and Machine Learning have increasingly allowed for the overcoming of these hurdles. Therefore, the goal of the work done here is to implement a complete pipeline for the recognition of signs from the Libras manual alphabet, including the gathering of sources for the development of the dataset, the extraction of landmarks from each hand, and the training of a Long Short-Term Memory network capable of analysing individual signs.
Keywords: Brazilian Sign Language; LSTM network; Landmark Detection; Sign Language Recognition

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

Adam	*Adaptive Moment Estimation*
ASL	*American Sign Language*
CNN	*Convolutional Neural Network*
FN	*False Negatives*
FNN	*Feedforward Neural Network*
FP	*False Positives*
fps	*frames per second*
IBGE	Instituto Brasileiro de Geografia e Estatística
Libras	Língua Brasileira de Sinais
LSTM	*Long Short-Term Memory*
ReLU	*Rectified Linear Unit*
RNN	*Recurrent Neural Network*
TP	*True Positives*

Sumário

1 Introdução

Historicamente, a qualidade de vida de pessoas com deficiência auditiva dificilmente foi comparável ao restante da população. Essa dissintonia pode ser atribuída a diversos fatores, entre eles a falta da aceitação das formas de linguagem não orais como meios válidos de comunicação e ensino (MOORES, 2010), algo que se reforçou graças a grande prevalência da comunicação verbal entre a população em geral. Desde a segunda metade do século XX, porém, a validade desta abordagem uniforme vem sendo questionada pela sociedade como um todo, e o interesse em integrar a linguagem de sinais ao nível institucional se torna cada vez mais prevalente.

No Brasil, as linguagens de sinais começaram a se legitimar apenas recentemente. Conforme a Lei Federal nº 10.436, a Língua Brasileira de Sinais (Libras) começou a ser reconhecida como meio legal de comunicação e expressão em 2002 (BRASIL, 2002). Porém, apesar de ser a língua de sinais mais difundida do país, uma pluralidade da população brasileira não é fluente em Libras ao ponto de manter uma conversa com outro falante. A baixa permeabilidade de Libras não é um problema que se manifesta somente entre pessoas ouvintes pois, até mesmo entre pessoas com surdez completa, apenas cerca de 22% desta parcela da população possui conhecimento na língua, segundo a Pesquisa Nacional de Saúde de 2019 (IBGE, 2021). Considerando essas estatísticas, é seguro dizer que, apesar do progresso obtido nos últimos tempos, ainda há muito a ser feito para informar a população, especialmente aqueles que demonstram a maior necessidade, a respeito desta língua.

Contribuindo para a dificuldade do aprendizado de Libras, podem ser identificadas, no sistema educacional brasileiro, diversas dificuldades que complicam o aprendizado pleno da língua, como a carência de profissionais adequados, a ausência de metologias específicas para o ensino de Libras, e a escassez de material didático apropriado (MELO; OLIVEIRA, 2012).

Considerando a falta de uma infraestrutura tradicional de acessibilidade voltada aos usuários de Libras, mostra-se uma oportunidade de explorar o potencial de métodos relacionados à Visão Computacional e às Redes Neurais Recorrentes (*Recurrent Neural Networks* - RNNs). Mais especificamente, a capacidade de desenvolvimento de um módulo em *software* capaz de capturar e reconhecer sinais automaticamente. A aplicação desta e outras tecnologias semelhantes tem o potencial de complementar o ensino de alunos surdos, que possuem maior facilidade de interação com meios de comunicação não-verbais, como imagens, animações e vídeos (TAVARES; OLIVEIRA, 2014).

1.1 Problemática

O reconhecimento de sinais de Libras é um problema complexo, visto que, segundo Capovilla et al. (2017) existem mais de 14 mil sinais diferentes dentro dessa linguagem. Observando-se a interseção entre a Visão Computacional e o campo do reconhecimento e tradução de língua de sinais tem-se que a mesma pode ser dividida em dois aspectos, de acordo com Sarmento (2023): i) análise de sinais estáticos (imagens) e ii) análise de sinais dinâmicos (vídeos). O sinal estático é aquele que não apresenta movimentação durante a execução, em contraposição ao sinal dinâmico, o qual apresenta movimentação em sua realização.

Sendo assim, especificamente o reconhecimento do alfabeto manual de Libras consiste em um desafio porque, apesar da maioria das letras serem sinais estáticos, existem algumas, tais como “Ç”, “H”, “J”, “X” e “Z”, que são representados por sinais dinâmicos.

1.2 Justificativa

Dada a falta de suporte infraestrutural para as linguagens de sinais, a criação de um sistema de reconhecimento de gestos apresenta uma possibilidade de complementar o ensino de Libras não apenas para a comunidade surda, mas para todos aqueles interessados na língua, que podem acabar sendo dissuadidos pela inacessibilidade do material didático atual. Além disso, o rápido progresso das redes neurais recorrentes cada vez mais viabiliza a construção destes sistemas.

Em específico para o desenvolvimento deste trabalho, propõe-se a utilização de uma rede *Long Short-Term Memory* para o reconhecimento dos gestos. Como o nome sugere, possui uma célula de memória capaz de guardar dependências temporais a curto e longo prazo, até diversas gerações, permitindo finalmente o processamento pleno das informações que possuem um forte contexto temporal, como é o caso dos gestos de Libras.

Considerando esses fatores, somados à problemática exposta na seção anterior, justifica-se o projeto e a implementação de um sistema capaz de reconhecer todos os sinais do alfabeto manual de Libras, utilizando-se, para isso, de técnicas de Visão Computacional, Aprendizado de Máquina e Aprendizado Profundo. Tal sistema, uma vez implementado de forma modular, poderá compor uma futura aplicação mais geral para reconhecimento e tradução de sinais de Libras.

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo principal deste trabalho foi o desenvolvimento e validação de um sistema capaz de reconhecer todos os sinais do alfabeto manual de Libras, sejam eles estáticos ou dinâmicos, excetuando-se os números. Para a extração dos pontos de referência da mão, foram utilizados os serviços da biblioteca MediaPipe *Hands* e para a classificação dos gestos, uma rede LSTM. O sistema desenvolvido poderá, em trabalhos futuros, fazer parte de um software mais completo visando o reconhecimento de Libras.

1.3.2 Objetivos Específicos

- Construir um sistema capaz de reconhecer todos os sinais do alfabeto manual de Libras, a partir de sequências de vídeos;
- Construir um *dataset* de amostras de sinais do alfabeto manual de Libras, a partir da extração dos pontos de referência da mão; e
- Realizar experimentos para validação do sistema implementado, a partir de sequências de vídeo dos sinais do alfabeto manual de Libras efetuados pelo usuário, em tempo real. A validação se dará a partir do uso de métricas quantitativas, como por exemplo, o cálculo da acurácia, precisão, sensibilidade e matriz de confusão, com relação ao reconhecimento dos gestos correspondentes às letras do alfabeto de Libras.

1.4 Organização do Trabalho

Os próximos capítulos estão organizados da seguinte forma: No Capítulo 2 são apresentados os conceitos mais relevantes para a compreensão plena do projeto. O Capítulo 3 discorre sobre os materiais e métodos implementados. No Capítulo 4 são apresentados e analisados os resultados obtidos. Por fim, o Capítulo 5 apresenta as conclusões e os potenciais desdobramentos futuros do trabalho.

2 Fundamentação Teórica

Este capítulo contém explicações a respeito dos temas essenciais para a compreensão do trabalho, como as Redes Neurais com foco nas Redes *Long Short-Term Memory*, o conceito de Pontos de Referência no contexto de Visão Computacional, as fontes utilizadas para criação do Conjunto de Dados, bem como os principais trabalhos correlatos encontrados por meio de um levantamento bibliográfico da literatura.

2.1 Língua Brasileira de Sinais

A linguagem de sinais, definida de maneira simples, é uma forma de comunicação marcada pelo uso de uma modalidade gestual-visual (MACHADO; WEININGER, 2018) que, assim como a linguagem oral, não constitui uma única língua internacional, com cada língua possuindo uma diferente estrutura e tradição, dependendo de suas raízes geográficas e culturais (LUCAS, 2001).

Nacionalmente, a língua com maior reconhecimento social e institucional é a Língua Brasileira de Sinais (BRASIL, 2002) (IBGE, 2021). Assim como muitas outras línguas de sinais, o conjunto de regras desta língua é próprio, e não depende das regras sintáticas impostas pela língua portuguesa. A Figura 1 mostra um exemplo de uma frase expressada em Libras. Analisando a imagem, é possível perceber a importância dos gestos combinados à expressão facial do falante, assim como a diferença entre a estrutura sintática de uma frase em Libras, cujos elementos podem assumir a ordem Sujeito-Objeto-Verbo, ou Sujeito-Verbo-Objeto como é a língua portuguesa (SCHLINDWEIN; AQUINO, 2021).

Figura 1 – Frase “O homem come a manga” em Libras.

Fonte: Adaptada de Schlindwein e Aquino (2021).

2.1.1 Alfabeto Manual

Dentre os sinais de Libras, um subconjunto importante diz respeito ao alfabeto manual (CAPOVILLA et al., 2017). O alfabeto manual de Libras é composto pelos gestos de mão que representam as letras e os números. A Figura 2 mostra os sinais do alfabeto manual de Libras correspondentes às letras. A datilologia é a soletração de uma palavra usando o alfabeto manual, sendo mais usada para expressar nome de pessoas, localidades e outras palavras que não possuem um sinal específico. Uma pessoa que não é surda pode usar a datilologia quando ela não sabe o sinal correspondente do que quer falar com outra pessoa surda e para que o surdo entenda do que se trata, deve-se soletrar usando o alfabeto manual. Do mesmo modo que algumas línguas orais possuem alfabetos diferentes, como é o caso da língua japonesa e chinesa, nas línguas de sinais, as formas de mãos para a formação do alfabeto manual também variam de país para país.

Figura 2 – Alfabeto manual de Libras – letras de A – Z

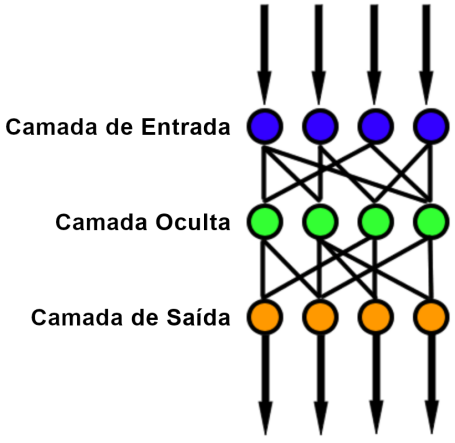
Fonte: Capovilla et al. (2017).

2.2 Redes Neurais Artificiais

As Redes Neurais Artificiais constituem um modelo muito usado no aprendizado de máquina inspirado pelo funcionamento do cérebro de organismos biológicos. Em termos práticos, isso se traduz para um modelo que é capaz de atingir um comportamento inteligente e complexo a partir de parâmetros iniciais simples e lógicos (GOODFELLOW; BENGIO; COURVILLE, 2016). Seguindo a analogia ao cérebro humano, redes neurais são compostas por diversos neurônios responsáveis pelo armazenamento de informação numérica. Apesar dos diferentes subgrupos que existem dentro do conceito de redes neurais, quase todos compartilham de uma arquitetura fundamental, definida pela presença de camadas que propagam os dados de entrada em diferentes maneiras.

A fim de transformar os dados recebidos na camada de entrada em uma previsão na camada de saída, as camadas intermediárias, também chamadas de camadas ocultas, empregam funções cujos parâmetros ajustáveis são chamados de pesos. Estes pesos são ajustados para minimizar ou maximizar uma função objetivo, que representa o erro entre os resultados atuais e os resultados esperados. É esse constante ajuste dos pesos que refina a performance do modelo neural ao longo do tempo. A Figura 3 apresenta uma simples rede neural cujo fluxo de dados é unidirecional, conhecida como Rede Neural de Propagação Direta (*Feedforward Neural Network* - FNN).

Figura 3 – Estrutura de uma Rede Neural de Propagação Direta (FNN)



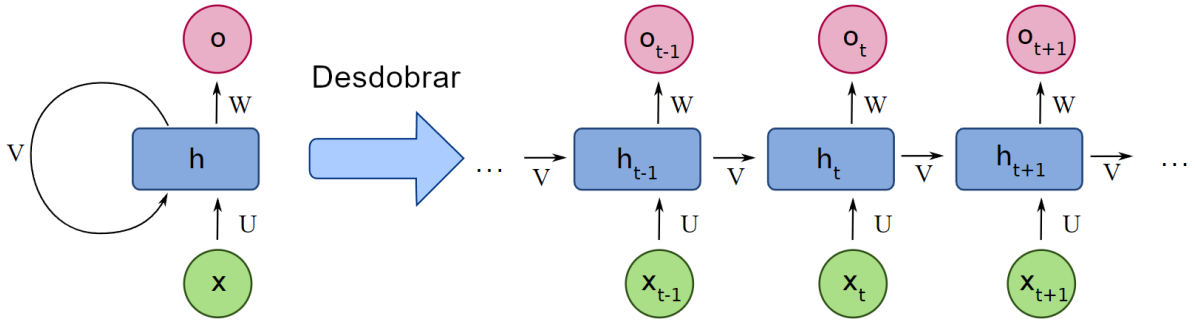
Fonte: Adaptado de Contributors (2006).

Entretanto, a rigidez do fluxo de dados encontrado nas FNNs gera uma grande dificuldade para a análise de informações que possuem dependências temporais, visto

que estas redes neurais não utilizam o contexto das características anteriores para o processamento de novas características. Isso não é um problema para a análise de dados estáticos, como imagens individuais por exemplo, mas no contexto da análise de sinais dinâmicos de Libras a implementação de uma FNN se torna inviável.

Uma solução para o reconhecimento de dados sequenciais é encontrada nas Redes Neurais Recorrentes *Recurrent Neural Networks* - RNNs). Estas redes neurais utilizam o resultado da previsão da última característica como contexto juntamente à nova entrada. Este processo de retroalimentação pode ser visto na Figura 4.

Figura 4 – Estrutura de uma Rede Neural Recorrente (RNN)



Fonte: Adaptado de Contributors (2020).

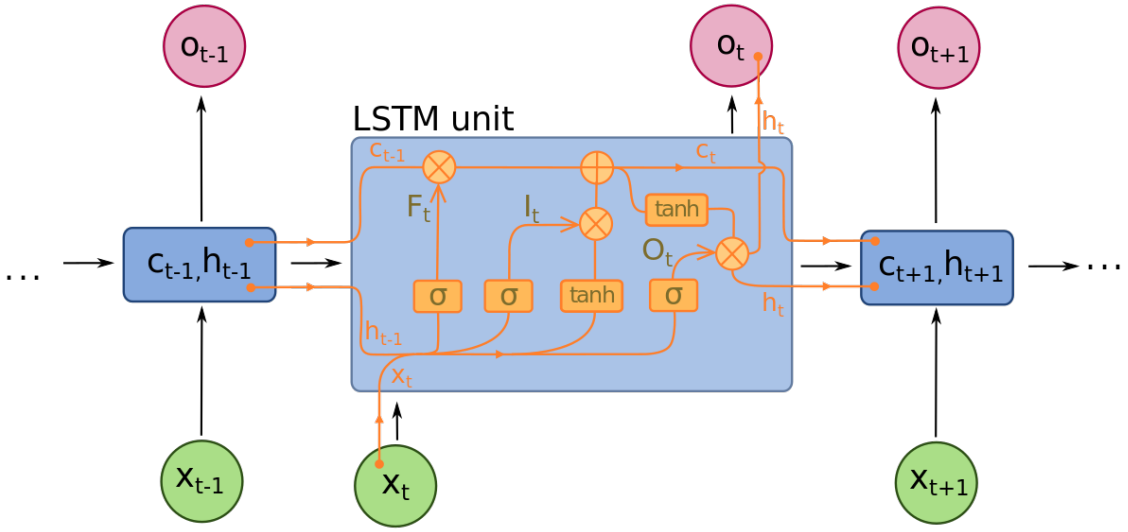
Esta arquitetura, porém, traz consigo um novo problema. A propagação do gradiente da função de perda é um método de otimização frequentemente aplicado para a atualização dos pesos nas camadas ocultas. Este gradiente, como derivada parcial da função de perda, tende a diminuir com cada propagação, o que pode resultar em um progresso proibitivamente lento, se não uma parada total na evolução da rede (HOCHREITER; SCHMIDHUBER, 1997).

2.2.1 Rede *Long Short-Term Memory*

A rede *Long Short-Term Memory* (LSTM) foi projetada para superar as limitações das Redes Neurais Recorrentes tradicionais, pois são capazes de aprender dependências de longo e curto prazo. A arquitetura da rede LSTM é composta por unidades de memória chamadas células. Cada célula possui três componentes principais: uma porta de esquecimento, uma porta de entrada e uma porta de saída. Essas portas permitem que a rede LSTM controle o fluxo de informações e o aprendizado de dependências temporais (HOCHREITER; SCHMIDHUBER, 1997).

A Figura 5 apresenta a arquitetura da rede LSTM. É possível observar que a célula possui duas linhas horizontais que atravessam toda a sua extensão. Isto resulta na transferência de informações da célula antiga para a subsequente, até o final da rede neural. Esta arquitetura permite a captura de informações relevantes em diferentes intervalos temporais, mantendo e atualizando a memória de acordo com as necessidades do problema. Por isso, as LSTMs consistem em uma ferramenta poderosa para modelar dependências temporais em dados sequenciais, sendo especialmente eficazes em tarefas que envolvem sequências longas como tradução de idiomas e reconhecimento de fala.

Figura 5 – Célula LSTM.



Fonte: Contributors (2020).

Observando a figura, é possível reparar três entradas para a célula de memória: O estado da célula anterior (c_{t-1}) equivalente à memória de longo prazo, o estado oculto da etapa anterior (h_{t-1}) equivalente à memória de curto prazo, e os dados de entrada (x_t) da etapa atual. Dentro da célula, os vetores h_{t-1} e x_t são concatenados em um novo vetor $[h_{t-1}, x_t]$. A porta de esquecimento (F_t) é responsável pela decisão de quais elementos serão esquecidos da memória de longo prazo. Já a porta de entrada (I_t) modifica os elementos que serão adicionados à esta memória. Por fim, a porta de saída (O_t) seleciona os elementos que serão mantidos no *output* da célula (o_t), que é equivalente ao vetor de estado oculto da etapa atual (h_t).

Todas as portas F_t, I_t e O_t são definidas pela sigmoide (σ) do vetor $[h_{t-1}, x_t]$, e são dadas pelas Equações 2.1, 2.2 e 2.3, enquanto os estados da célula c_t e h_t são dados pelas Equações 2.4 e 2.5 conforme Huang e Ye (2021). As funções sigmoide e tangente hiperbólica frequentemente utilizadas servem para mapear qualquer valor

para um número entre os intervalos $[0, 1]$ e $[-1, 1]$ respectivamente, e são dadas pelas Equações 2.6 e 2.7.

$$F_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.1}$$

$$I_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.2}$$

$$O_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.3}$$

$$c_t = F_t \cdot c_{t-1} + I_t \cdot tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.4}$$

$$h_t = O_t \ \cdot \ tanh(c_t) \tag{2.5}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.6}$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.7}$$

Nas equações anteriores, as variáveis W e b representam, respectivamente, os pesos e o viés utilizados pela rede para o ajuste das previsões.

2.2.2 Métricas Avaliativas

No contexto do Aprendizado de Máquina, a fim de avaliar o desempenho de um modelo de forma concreta, são utilizadas métricas que representam a performance deste em diferentes situações. Especificamente neste trabalho, foram utilizadas as seguintes métricas: Precisão, Revocação (*Recall*) e F-medida, definidas segundo as Equações 2.8, 2.9 e 2.10.

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2.8}$$

$$\text{Revocação} = \frac{TP}{TP + FN} \tag{2.9}$$

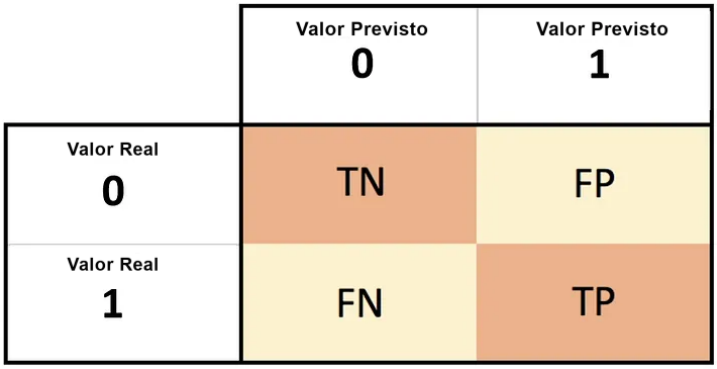
$$\text{F-Medida} = \frac{2 \cdot \text{Precisão} \cdot \text{Sensibilidade}}{\text{Precisão} + \text{Sensibilidade}} \tag{2.10}$$

Em que TP, FP e FN representam o total de verdadeiros positivos, falsos positivos e falsos negativos, respectivamente. Cada métrica pode assumir um valor entre 0 e 1, no qual 1 representa o cenário ideal em que os resultados falsos são mantidos a um mínimo.

A precisão é equivalente à porcentagem dos resultados positivos que representam a classe esperada, portanto, representa a capacidade do modelo em “focar” em uma classe específica no momento de fazer uma previsão. Já a revocação é equivalente à porcentagem das amostras de uma classe que foram corretamente identificadas, representando o quão sensível o modelo é àquela classe específica. Por esse motivo, essa métrica também pode ser chamada de sensibilidade. Por fim, a F-medida é a média harmônica de ambas as métricas, representando o desempenho combinado de ambas as métricas no reconhecimento da classe.

Para melhor visualizar as previsões do modelo em relação à cada classe, é utilizada uma tabela para visualizar a incidência de TP, FP e FN, chamada Matriz de Confusão. Na Figura 6 é apresentada uma matriz de confusão com apenas uma classe.

Figura 6 – Exemplo de Matriz de Confusão



Fonte: Adaptado de Zuhaib (2019).

Em situações em que o conjunto de dados possui múltiplas classes, como é o caso neste trabalho, a coluna à esquerda contém as classes esperadas, e a linha do topo contém as classes previstas. A diagonal principal da matriz representa os casos em que a previsão coincide com a classe esperada. Para uma célula específica contida na diagonal principal, esta representa os verdadeiros positivos da classe, as células na mesma linha representam os falsos negativos, os valores na mesma coluna representam os falsos positivos, e as células restantes representam os verdadeiros negativos.

Para os propósitos deste trabalho, todas as ferramentas necessárias para a elaboração da rede neural e monitoramento das métricas estão inclusas na biblioteca Tensorflow para o Python, com destaque especial às funcionalidades da biblioteca Keras.

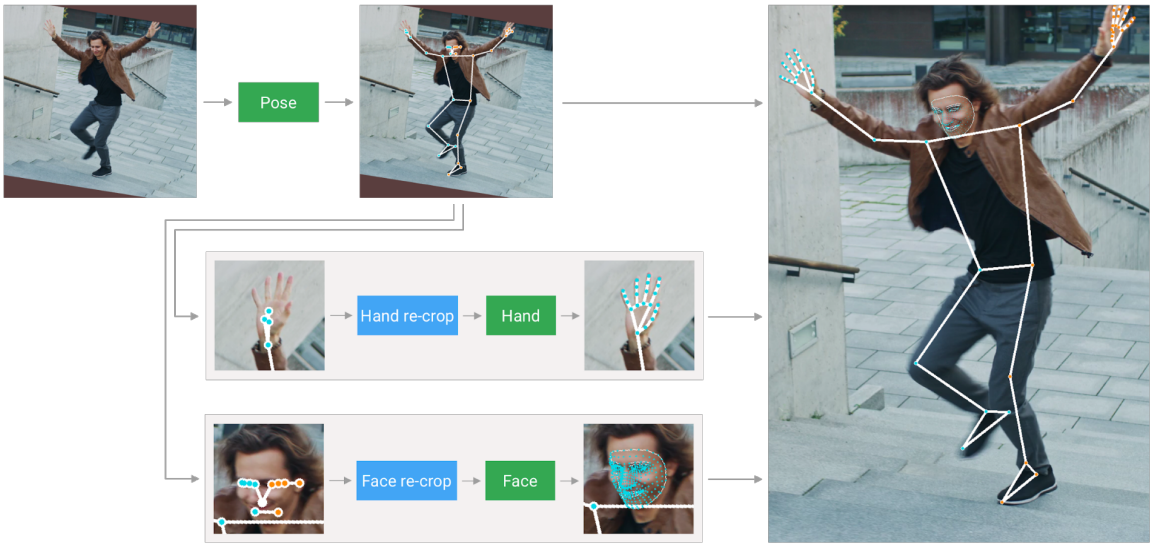
2.3 *Landmarks*

Os pontos de referência, também chamados de *landmarks* ou *keypoints*, são marcadores que identificam pontos de importância no corpo humano. Dependendo da tarefa, estes pontos podem representar diversos locais distintos do corpo humano, como o rosto, busto, e as juntas de braços, mãos e pés. Estes locais formados por um conjunto de pontos de referência são frequentemente chamados de regiões de interesse (*Regions of Interest - ROI*).

Uma das ferramentas mais populares para estimação dos pontos de referência foi disponibilizada recentemente pela empresa Google através da biblioteca de código aberto MediaPipe, a qual é capaz de realizar a estimação dos pontos de referência do esqueleto humano a partir de uma sequência de vídeo. Isso é possível através de uma identificação inicial custosa das ROIs encontradas em um quadro, seguida pelo rastreamento contínuo da movimentação dos *landmarks* utilizando inferência contextual (MediaPipe Hands, 2024).

O módulo mais completo desta ferramenta é chamado MediaPipe *Holistic*, e é capaz de rastrear três ROIs distintas: As feições do rosto, a postura corporal, e as juntas das mãos (MediaPipe Holistic, 2024). A Figura 7 mostra os *landmarks* gerados pelo reconhecimento destas regiões.

Figura 7 – *Landmarks* estimados pelo MediaPipe *Holistic*

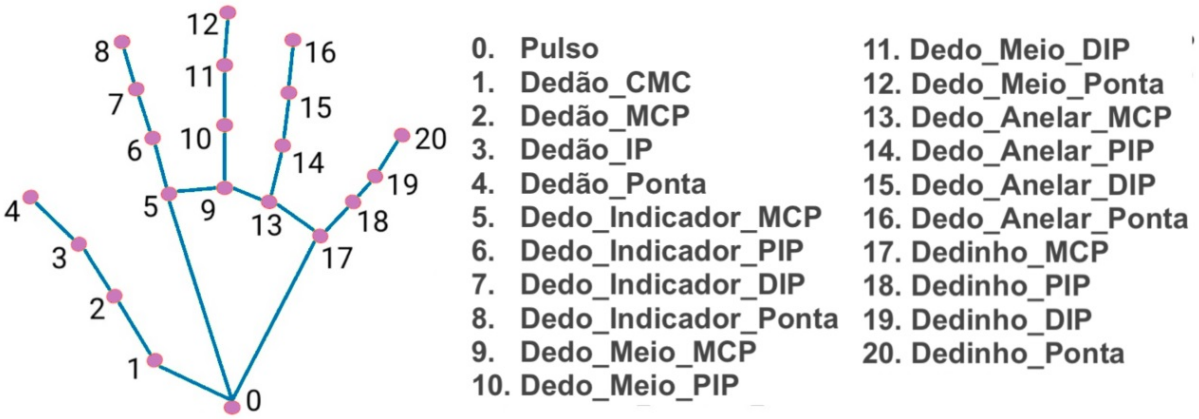


Fonte: Adaptado de MediaPipe Hands (2024).

Especificamente, para o propósito de reconhecimento dos sinais do alfabeto

manual de Libras, basta a estimação dos pontos de referência das mãos. O MediaPipe *Hands*, que faz parte desta biblioteca, é capaz de estimar 21 pontos de referência para cada uma das mãos. A Figura 8 apresenta a posição dos *landmarks* identificados pelo MediaPipe *Hands*, assim como a descrição de cada ponto.

Figura 8 – Pontos de referência da Mão



Fonte: Adaptado de MediaPipe Hands (2024).

Cabe ressaltar que, neste trabalho, utilizou-se o MediaPipe *Hands* através da linguagem Python para a extração dos pontos de referência da mão.

2.4 Conjunto de Dados

O conjunto de dados (*dataset*) representa toda a informação disponível para a execução da tarefa em mãos. No caso deste trabalho, é o coletivo de vídeos obtidos para extração das características que eventualmente foram empregadas no treinamento da rede LSTM.

Em função do escopo limitado deste trabalho, unido à grande dificuldade de criação de uma base de dados própria, foram reunidas diferentes fontes de dados pre-existentes para a elaboração de um único *dataset*. Estas fontes são as que seguem.

1. **Fonte V-LIBRASIL da Universidade Federal de Pernambuco (UFPE)**

   A base de dados do V-LIBRASIL faz parte da dissertação de mestrado de Rodrigues (2021), contém mais de 4000 vídeos de Libras realizados por múltiplos intérpretes, e foi criada com o expresso intuito de ampliar a disponibilidade de dados em Libras.

   A base de dados está disponível no site ⟨https://libras.cin.ufpe.br⟩

2. **Fonte do Canal “Dicionário Virtual LIBRAS”**

   O canal “Dicionário Virtual LIBRAS” contém um acervo com mais de 400 vídeos dedicados à criação de uma base de dados a fim de auxiliar profissionais do setor público.

   Os vídeos podem ser encontrados no canal ⟨https://www.youtube.com/@dicionariovisuallibras8919⟩

3. **Fonte do Canal “Dr. Mike!”**

   O canal “Dr. Mike!” é focado na realização de conteúdo educativo para crianças no estágio de primeira infância. Por ser um canal de língua inglesa, a linguagem de sinais utilizada é a Linguagem de Sinais Americana (*American Sign Language - ASL*). Porém, devido à coincidência de alguns sinais, esta foi utilizada para completar os dados de algumas classes.

   Os vídeos podem ser encontrados no canal ⟨https://www.youtube.com/@DrMikeEarlyLearning⟩

4. **Fonte do Canal “Gilbervan Soares”**

   O canal “Gilbervan Soares” no *YouTube* tem como objetivo a divulgação de materiais voltados ao aprendizado de Libras, como o ensino de palavras e frases cotidianas, e a tradução de diversos textos para a língua. Atualmente conta com mais de 200 vídeos.

   Os vídeos podem ser encontrados no canal ⟨https://www.youtube.com/@gilbervansoares⟩

5. **Fonte do Canal “Incluir Tecnologia”**

   O canal “Incluir Tecnologia” é outro canal que se propõe a distribuir conteúdo auxiliar à educação e disseminação do aprendizado de Libras. Atualmente, conta com mais de 1000 vídeos sobre a sinalização de diferentes palavras em Libras.

   Os vídeos podem ser encontrados no canal ⟨https://www.youtube.com/@incluirtecnologia⟩

2.5 Trabalhos Correlatos

Graças à correlação entre os projetos de reconhecimento de linguagem de sinais, independente de fatores específicos como a linguagem ou modelo de rede neural, foram reunidos alguns trabalhos para melhor compreender a metodologia aplicada. A Tabela 1 apresenta uma sumarização das principais diferenças entre os conjuntos de dados e o método empregado.

O trabalho de Huang e Ye (2021) envolve a implementação de um sistema baseado em *Boundary-Adaptive Encoders* utilizando rede LSTM para a detecção dos limites entre um sinal e outro durante o reconhecimento contínuo da linguagem chinesa de sinais. Esta língua possui uma diferença significativa quando comparada à Libras, visto que diversas palavras são compostas por outras sub-palavras. Os exemplo dados pelo autor consistem na palavra “Jardim de Infância” e “Posto de Gasolina”, que são compostas, respectivamente, pelas sub-palavras “Criança” e “Casa”, e “Carro” e “Casa”. Os autores realizam a arquitetura do *encoder* proposto utilizando duas camadas de rede LSTM bidirecionais e avaliam a segmentação utilizando camadas LSTM simples. Como os resultados foram descritos utilizando outras métricas de avaliação semântica, como BLEU e ROUGE-L, decidiu-se expor o melhor resultado na métrica METEOR, que é a mais comparável à F-medida. O valor obtido nesta métrica foi de 70.6%.

No trabalho de Machado (2018) foi criado um *dataset* próprio em parceria com o Núcleo de Tecnologia Assistiva do Instituto Federal do Amazonas. Este conjunto de dados é formado por 510 sinais dinâmicos. Apesar disso, o trabalho utiliza um conjunto de apenas 84 classes para reconhecimento, devido à dificuldade de segmentação dos sinais nos vídeos. Após o pré-processamento, o reconhecimento foi feito através de uma rede convolucional 3D e LSTM para análise direta das imagens. O melhor valor obtido para a F-medida foi de 89%.

O trabalho de Rezende (2021) utiliza o *dataset* próprio MINDS-Libras, que contém múltiplos vídeos de cada sinal utilizando a câmera e um sensor simultaneamente. Além disso, oferece dados relativos a 25 pontos de referência distribuídos pelas juntas do corpo inteiro. A metodologia inclui o treino de uma CNN3D (*Convolutional Neural Network*) para a análise dos vídeos e uma TCN (*Temporal Convolutional Network*) para o reconhecimento das coordenadas. O melhor valor obtido para a F-medida foi de 99%.

Ambos os trabalhos de Machado (2018) e Rezende (2021), foi utilizado o sensor de movimentos *Kinect v2* em conjunto ao método tradicional de captura de vídeo somente com uma câmera. Isso permite uma estimação direta dos pontos de referência através de um *hardware* dedicado para o reconhecimento de *keypoints* em espaço tridimensional, ao contrário da inferência por *software* oferecida pelo *MediaPipe*.

No trabalho de Koller et al. (2020), são avaliadas múltiplas modalidades diferentes referentes às regiões de interesse, sendo necessária a análise combinada e sincronização do fluxo de características labiais, manuais e de glossário em paralelo.

Segundo os autores, essa divisão do reconhecimento em múltiplos sub-problemas é vantajosa para o processamento computacional. Se destaca também o uso de uma arquitetura híbrida CNN-LSTM-HMM (HMM - *Hidden Markov Model*). O melhor valor obtido para a F-medida foi de 57%.

O trabalho de Sundar e Bagyammal (2022) apresenta a maior semelhança à proposta deste trabalho, visto que se trata do reconhecimento do alfabeto manual da Língua Americana de Sinais (*American Sign Language* - ASL). Por isso, é dado uma importância adicional à metodologia utilizada pelos autores, especialmente na arquitetura do modelo LSTM utilizada. Além disso, o modelo foi treinado utilizando as coordenadas dos 21 pontos de referência do MediaPipe *Hands*. O melhor valor obtido para a F-medida foi de 100%

Os melhores resultados obtidos em termos das métricas avaliativas de interesse para este trabalho são aqueles apresentados por Sundar e Bagyammal (2022). Isso pode se dar por vários motivos, entre eles a utilização das mesmas condições de ambiente e sinalizadores para o treino e para teste, mesmo com uma partição dedicada para cada. Os resultados obtidos por Rezende (2021) também são extremamente promissores, algo que pode ser atribuído à utilização de três tipos diferentes de dados de entrada para um mesmo sinal: Vídeos em RGB, Vídeos em profundidade e Coordenadas x-y-z dos *landmarks*.

Tabela 1 – Trabalhos relacionados ao Reconhecimento de Linguagem de Sinais

Referência	**LS**	**C**	**I**	**R**	**T**	**M**
Huang e Ye (2021)	Chinesa	500	50	10	250.000	LSTM
Koller *et al*. (2019)	Alemã	1.081	9	N/A	3.361	CNN,LSTM,HMM
Machado (2018)	Brasileira	510	7	6	21.420	CNN,LSTM
Rezende (2021)	Brasileira	20	12	5	1.200	CNN,TCN
Silva (2020)	Brasileira	50	10	10	5.000	CNN,LSTM
Sundar *et al*. (2022)	Americana	26	4	30	3.120	LSTM

Nota – LS = Linguagem de sinal, C = Número de classes, I = Número de intérpretes, R = Número de repetições, T = Número total de amostras, M = Método

Fonte: Elaborada pelo autor.

3 Método Proposto

Neste capítulo são apresentados uma visão geral do método proposto, os detalhes de sua implementação, bem como os materiais utilizados.

3.1 Pré-processamento do *dataset* de vídeos de sinais

Utilizando as bases de dados destacadas na Seção 2.4, foram obtidos 155 vídeos únicos para a criação do *dataset* a ser utilizado. Como o conjunto de dados foi agregado de diversas fontes distintas, naturalmente há uma variação nas propriedades relativas aos metadados de cada arquivo. A Tabela 2 mostra a variação nestes metadados através da Mediana, Moda, Mínimo e Máximo encontrado para cada propriedade.

Tabela 2 – Características dos vídeos constituintes do *dataset*

	Média	**Mediana**	**Moda**	**Mínimo**	**Máximo**
Quadros por segundo	33.05	29.97	29.97	24.00	60.00
Total de quadros	165.2	135	95,137	83	542
Resolução	N/A	1920x1080	1920x1080	854x480	1920x1080

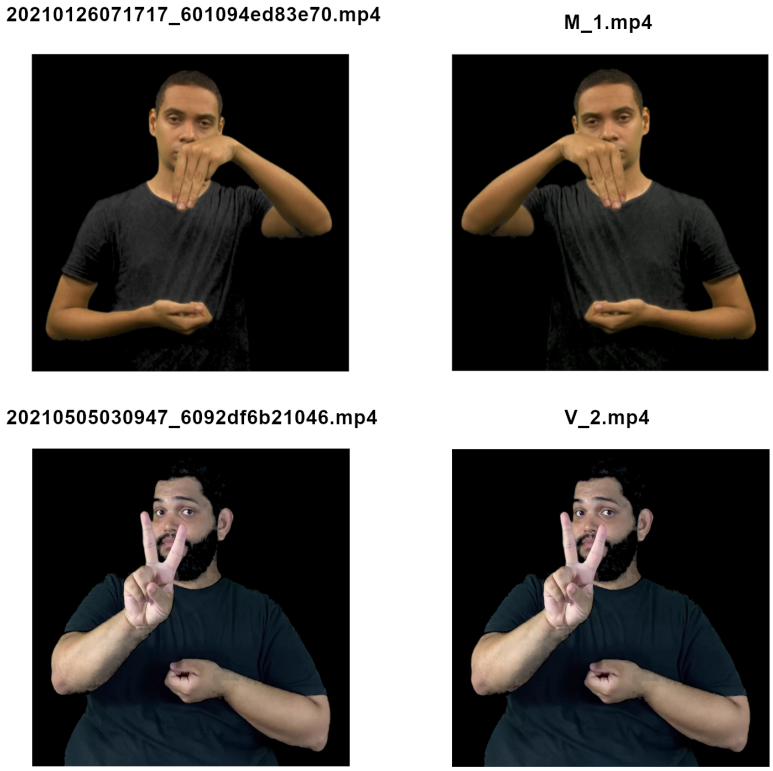
Fonte: Elaborada pelo autor.

Dada a grande varição entre essas características, justifica-se um conjunto de etapas preparatórias para normalização dos dados visando um ambiente mais controlado para treinamento da rede LSTM.

Primeiramente, cada vídeo é rotulado da seguinte maneira: “B_1.pm4”, em que “B” representa o sinal de Libras sendo realizado e “1” representa o intérprete que está o fazendo. Para isso, cada arquivo foi renomeado seguindo este mesmo padrão para que estes critérios possam ser recuperados no momento de captura dos *landmarks*. Então, todos os vídeos cujo intérprete realiza o movimento com a mão esquerda são espelhados horizontalmente. Essa etapa facilita a captura da mão correta durante a fase de coleta dos pontos de referência, dado que a mão a ser rastreada será sempre a mesma. A mão direita foi escolhida pois há uma grande predominância do uso desta por parte dos intérpretes. A Figura 9 mostra o resultado destes primeiros dois ajustes.

Em seguida, foi observada a taxa de quadros por segundo (*frames per second - fps*) de cada vídeo das bases de dados. Devido à maneira como os pontos de referência são coletados, é importante que este valor seja próximo para todas as amostras utilizadas. Para isso, foram consideradas duas opções: a) Reduzir o número de

Figura 9 – Amostras espelhadas e rotuladas



Fonte: Elaborada pelo autor.

quadros dos vídeos com uma taxa de fps alta ou b) Utilizar métodos de interpolação a fim de aumentar o número de quadros dos vídeos com uma taxa de fps baixa. Dada a prevalência dos valores padrões 29.97 e 30.00 no *dataset* e em mídias de vídeo no geral, faz-se conveniente a adoção destes valores como base. Portanto, o método utilizado consiste na remoção de metade dos quadros, intercaladamente, de todos os vídeos cuja taxa excede o valor de 30.00. Isso porque os únicos valores encontrados que atendem a esse critério são 59.94 e 60.00, o dobro dos valores mais presentes no conjunto de dados.

Por fim, é realizado um recorte uniforme dos quadros válidos centrais de cada vídeo. Os quadros são considerados válidos quando o reconhecimento dos *landmarks* é realizado com êxito. Este último passo garante que todos os vetores correspondentes aos dados capturados possuam as mesmas dimensões, o que é necessário para o treinamento do modelo, e que elementos externos como a posição neutra inicial e final sejam mantidas a um mínimo. As Figuras 10 e 11 mostram, respectivamente, as posições neutras no início e fim da amostra, e o tamanho das faixas de vídeo pré e pós o truncamento destes quadros.

Figura 10 – Sinal contido entre as posições neutras

Fonte: Elaborada pelo autor.

Figura 11 – Faixa de vídeo truncada

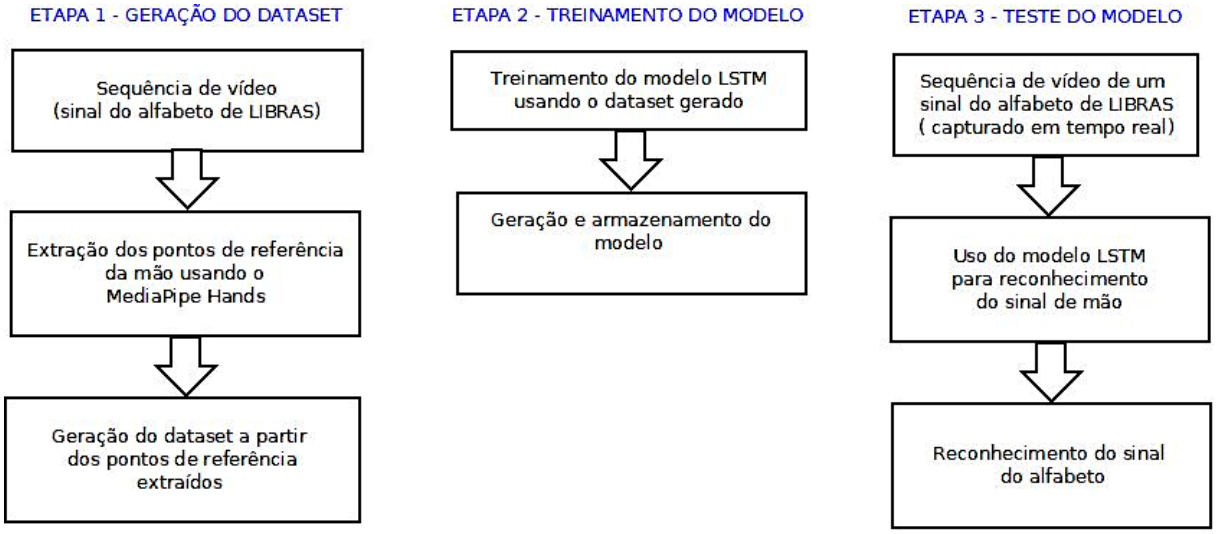
Fonte: Elaborada pelo autor.

A diferença na resolução entre os diferentes *datasets* foi mantida intacta, pois as coordenadas de cada ponto de referência detectado pelo *MediaPipe Hands* são normalizadas em relação às dimensões horizontal e vertical do vídeo. Dessa forma, em um primeiro momento, o valor específico deste metadado não resulta em nenhuma anomalia significativa na captura dos *landmarks*.

3.2 Visão Geral do Método

A Figura 12 ilustra a visão geral do método proposto. Cada uma das 3 etapas que compõem o método são descritas a seguir.

Figura 12 – Método Proposto



Fonte: Elaborada pelo autor.

3.2.1 Etapa 1 – Geração do *Dataset* de *Landmarks*

Após a padronização das propriedades, cada vídeo é processado quadro por quadro pelo detector de pontos de referência do *MediaPipe*. Para alguns quadros, especialmente aqueles cuja iluminação ou resolução não são elevadas, há a possibilidade de falha na detecção, ou seja, não são armazenadas coordenadas para cada *landmark* no quadro em questão. A fim de apurar a detecção nestas condições, foram alterados os valores padrões das variáveis *min detection confidence* e *min tracking confidence* para 0.6 e 0.4, respectivamente (originalmente 0.5). Assim, é necessário um nível de confiança inicial maior para a detecção das mãos, enquanto é dada maior leniência ao rastreamento contínuo de mãos já detectadas (MediaPipe Hands, 2024). A Figura 13 mostra os pontos de referência extraídos de um quadro em que houve êxito no reconhecimento.

Figura 13 – Pontos capturados

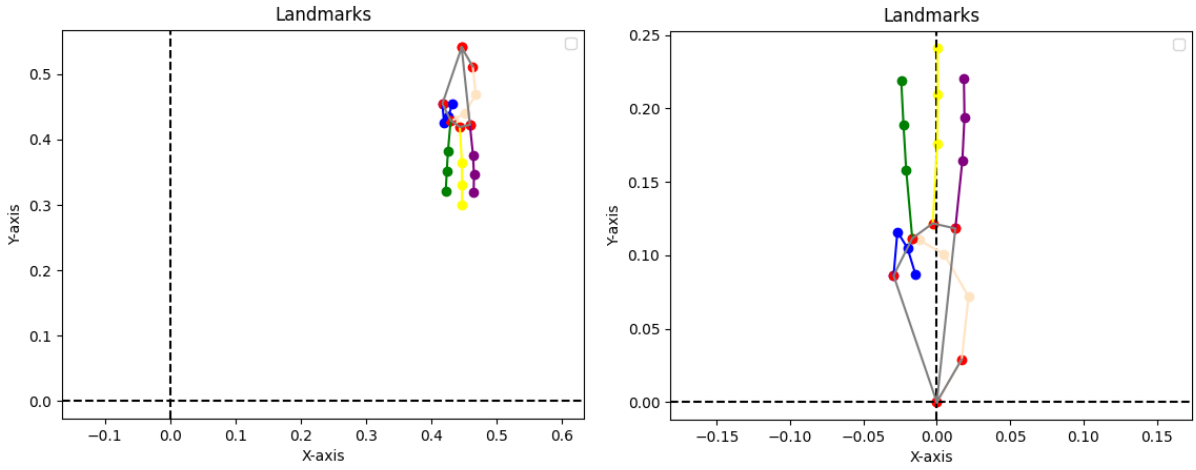
Fonte: Elaborada pelo autor.

Os dados extraídos pelo *MediaPipe* são equivalentes às coordenadas de cada ponto em um plano cartesiano. Cada detecção bem-sucedida gera 63 valores, equivalentes à posição no eixo X, Y e Z de cada um dos 21 pontos de referência. Estes valores são armazenados em um vetor tri-dimensional V, no qual $V[i]$ contém todos os quadros da amostra número i, e $V[i][j]$ contém todos os *landmarks* do quadro número j. Visando aumentar a capacidade de generalização do modelo, estes dados precisam ser normalizados antes de prosseguir para a fase de treino.

3.2.1.1 Normalização das coordenadas

Como foi mencionado brevemente na Seção 3.1 os valores numéricos das coordenadas X e Y calculados pelo *MediaPipe* são normalizados relativamente à posição do ponto ao longo das dimensões do quadro. Mais especificamente, pontos cujas coordenadas horizontal e vertical são próximas ao canto superior esquerdo do quadro aproximam-se do valor (0,0), enquanto pontos próximos ao canto inferior direito do quadro aproximam-se do valor (1,1). Esse método cria uma dependência entre o sinal realizado e a posição na tela na qual ele foi realizado que atrapalha a capacidade do modelo de generalizar as classes. Uma solução para esse problema é apresentada na Figura 14, na qual as coordenadas de todos os pontos são normalizadas em relação à um *landmark* específico. Por conveniência, escolheu-se o marcador número 0, localizado no pulso, para representar a origem do novo plano de coordenadas.

Figura 14 – Coordenadas normalizadas da letra “W”



Fonte: Elaborada pelo autor.

Ambos os conjuntos de coordenadas apresentados na figura anterior correspondem aos dados capturados em um mesmo quadro do sinal “W” sendo realizado.

Como os valores próximos ao teto do quadro convergem a zero na escala utilizada pelo *MediaPipe*, as coordenadas originais quando plotadas em gráfico mostram a mão invertida. Para melhor legibilidade dos resultados do reconhecimento, os dados foram espelhados ao longo do eixo X.

Os dados referentes à coordenada Z de cada *landmark* não sofrem alteração nesta etapa, visto que esta já utiliza a profundidade relativa para cálculo dos valores, e não uma distância absoluta como a resolução do vídeo.

3.2.1.2 Aumento de Dados

Considerando o baixo volume de dados obtidos para treinamento do modelo LSTM, decidiu-se realizar uma etapa intermediária adicional que consiste na utilização de um módulo de Aumento de Dados (*Data Augmentation* - *DA*). O Aumento de Dados é uma técnica implementada para melhorar a capacidade de generalização do modelo durante a fase de treino através de algoritmos que transformam o conjunto de dados original (WEI et al., 2022). Em virtude da natureza cartesiana dos dados, certas transformações geométricas podem ser utilizadas para gerar o conjunto de dados aumentado. Os dados obtidos a seguir são um produto da manipulação exclusiva da partição de treino.

A primeira transformação aplicada consiste na rotação de todos os pontos de referência em torno do eixo Z, simulando uma leve inclinação na mão realizando o sinal. As Equações 3.1, 3.2 e 3.3 foram utilizadas para obter este resultado. Em que x, y e z são as coordenadas originais, θ é o ângulo de rotação utilizado e x', y' e z' são as coordenadas resultantes.

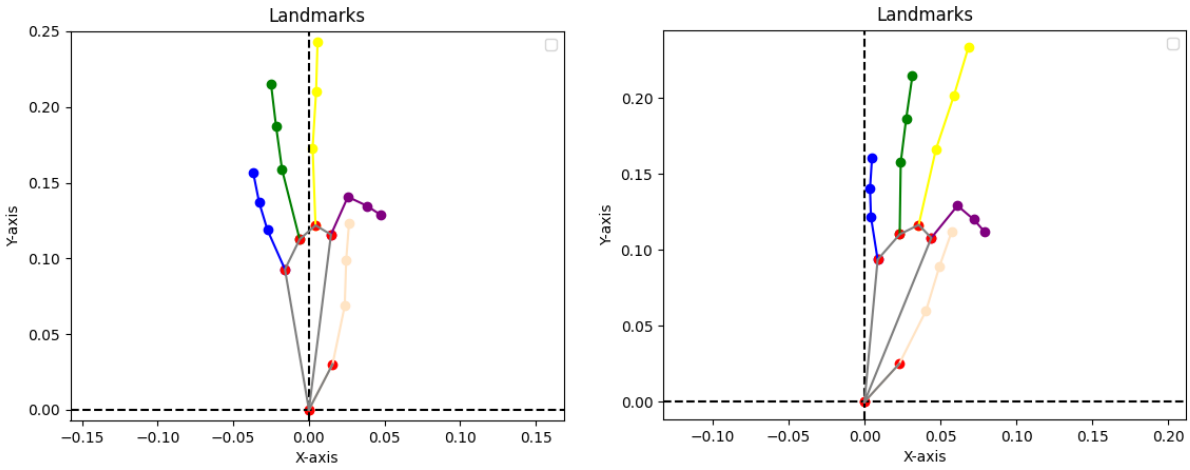
$$x' = xcos(\theta) + ysin(\theta) \tag{3.1}$$

$$y' = ycos(\theta) - xsin(\theta) \tag{3.2}$$

$$z' = z \tag{3.3}$$

Para evitar que os dados transformados se distanciem muito da maneira correta de realizar o sinal, o ângulo máximo de rotação foi limitado a 15 graus para ambos os sentidos, no qual cada 2 graus de rotação representam uma nova amostra. Um exemplo do resultado deste primeiro aumento é apresentado na Figura 15.

Figura 15 – Dados rotacionados da letra “T”



Fonte: Elaborada pelo autor.

A segunda transformação simula a aproximação e o afastamento da mão em relação à camera. Para isso, foi necessário considerar a profundidade de cada *landmark*, representada pela coordenada Z, e sua relação com a posição relativa de todos os outros pontos de referência. Mais especificamente, o aumento da distância entre dois pontos capturados representa uma redução inversamente proporcional da profundidade dos mesmos pontos. As Equações 3.4, 3.5 e 3.6 apresentam o método utilizado para calcular o novo conjunto de coordenadas. Em que x, y e z são as coordenadas originais, k é a constante de escala e x', y' e z' são as coordenadas resultantes.

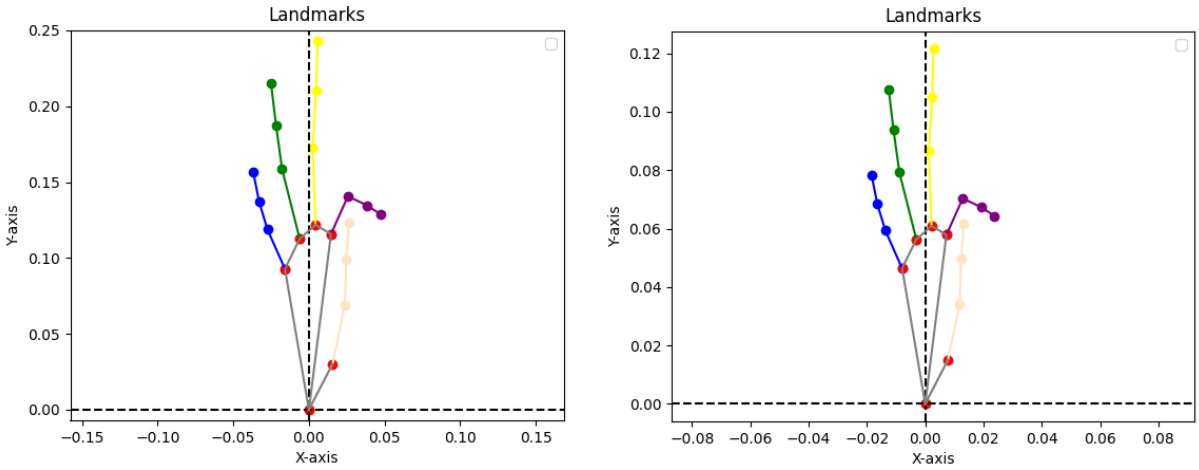
$$x' = kx \tag{3.4}$$

$$y' = ky \tag{3.5}$$

$$z' = \frac{z}{k} \tag{3.6}$$

No contexto do *dataset* final, foram substituídos para k valores dentro do intervalo $[0.5, 1.5]$ em incrementos de 0.25, ou seja, foram adicionadas ao conjunto de treino as amostras escaladas desde 50%, até 150% de seu valor original. A Figura 16 apresenta um exemplo das coordenadas obtidas através desta transformação.

Figura 16 – Dados reescalados da letra “T”



Fonte: Elaborada pelo autor.

Por fim, todas as amostras são duplicadas e espelhadas ao longo do eixo Y. Conforme mencionado na Seção 3.1, todos os vídeos constituintes do conjunto de dados apresentam os sinais sendo realizados com a mão direita. Esta última etapa é eficaz em simular o equivalente para a mão esquerda de todas as amostras geradas até então. As Equações 3.7, 3.8 e 3.9 mostram o resultado desta operação. Em que x, y e z são as coordenadas originais, e x', y' e z' são as coordenadas resultantes.

$$x' = -x \tag{3.7}$$

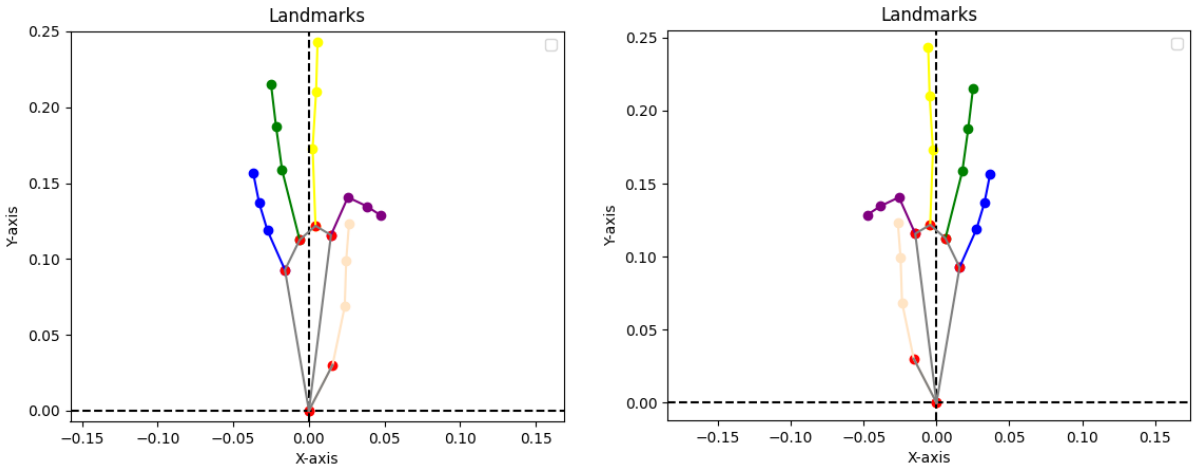
$$y' = y \tag{3.8}$$

$$z' = z \tag{3.9}$$

A Figura 17 apresenta as coordenadas relativas às mãos direita e esquerda de

uma mesma amostra inicial.

Figura 17 – Dados espelhados da letra “T”



Fonte: Elaborada pelo autor.

3.3 Etapa 2 – Treinamento do Modelo

Na etapa anterior, foram feitos todos os preparativos necessários para os dados que serão utilizados na entrada da rede sequencial LSTM. Portanto, conforme ilustrado na Figura 12, na etapa atual, efetuou-se o treinamento do modelo em si. Foi utilizado como base para a rede o trabalho de Sundar e Bagyammal (2022). A Tabela 3 apresenta a arquitetura do modelo utilizado.

Tabela 3 – Arquitetura base do modelo LSTM

Tipo de camada	**Dimensão de saída**	**Nº de parâmetros**
LSTM	(None,41,64)	32768
LSTM	(None,41,128)	98816
LSTM	(None,64)	49408
Dense	(None,64)	4160
Dense	(None,32)	2080
Dense	(None,27)	891

Fonte: Elaborada pelo autor.

Além desta estrutura básica, o sistema utiliza os seguintes hiper-parâmetros:

- **Número de épocas:** 2000
- ***Batch Size*:** 32

- **Função de ativação das camadas ocultas:** *ReLU*
- **Função de ativação da camada de predição:** *Softmax*
- **Função de perda:** *Sparse Categorical Cross-entropy*
- **Método de otimização:** *Adam*

A Etapa 3 do método proposto refere-se ao teste do modelo em si, sendo detalhada no próximo capítulo.

3.4 Materiais Utilizados

Neste projeto foram utilizados os seguintes componentes de hardware e software.

3.4.1 Componentes de *Hardware*

- CPU: AMD Ryzen 7 2700X @3,70GHz;
- GPU: NVIDIA GeForce RTX 3060 Ti (8 GB, GDDR6);
- RAM: 32 GB (DDR4);
- HDD: Seagate ST2000DM008 de 2TB (2 TB, 7200RPM);
- SSD: Kingston SA1000M8240G (240 GB);

3.4.2 Componentes de *Software*

- Sistema Operacional: Windows 11 64bit (Desktop);
- Ambiente de desenvolvimento: Microsoft Visual Studio Code;
- Biblioteca MediaPipe Hands;
- Biblioteca Tensorflow;

4 Resultados Experimentais

Neste capítulo estão expressos os resultados dos experimentos obtidos utilizando a metodologia estabelecida, assim como uma apresentação final da ferramenta desenvolvida. Além disso, são discutidos os possíveis motivos responsáveis pelos resultados finais, utilizando como base as métricas avaliativas definidas na Seção 2.2.2, bem como potenciais ajustes que poderiam ser feitos a fim de obter resultados mais satisfatórios no futuro.

4.1 Conjuntos de Treinamento e Teste

Para o *dataset* deste trabalho foi adotada a seguinte divisão de dados: aproximadamente 83% dos vídeos foram reservados para treinamento do modelo e os restantes 17% foram utilizados no teste dos resultados finais. Em números absolutos, essas partições resultam em, em média, 5 vídeos dedicados para o treinamento e 1 vídeo para o teste de cada sinal do alfabeto. A base de dados do canal Incluir Tecnologia foi utilizada para o conjunto de testes, dado que esta conta com todas as classes do experimento. A Tabela 4 apresenta informações relativas aos sinais do alfabeto (classes) obtidos por base de dados.

Tabela 4 – Volume de dados do *dataset*

Base de dados	**Classes**	**I**	**T**
V-LIBRASIL	24	3	71
Canal “Gilbervan Soares”	27	1	27
Canal “Dicionário Virtual LIBRAS”	27	1	27
Canal “Incluir Tecnologia”	27	1	27
Canal “Dr. Mike!”	3	1	3
Partição de Treino	27	5	128
Partição de Teste	27	1	27
Total	27	6	155

Nota – I = Número modal de intérpretes, T= Número total de amostras (vídeos)

Fonte: Elaborada pelo autor.

A diferença entre a soma do número de amostras para cada base de dados e o total obtido surge em função da falta de vídeos para alguns sinais específicos. No caso da base de dados V-LIBRASIL, por exemplo, não foram encontradas amostras relativas aos sinais “C”, “Ç” e “O”. Além disso, um dos vídeos em que o sinal “L” é realizado não

está disponível por inteiro. Por esses motivos, alguns sinais possuem apenas 4 vídeos por sinal, e a letra “Ç” em específico possui apenas 2 vídeos correspondentes.

4.2 Teste do Modelo e Análise dos Resultados

Devido à quantidade escassa de dados para teste, seguindo a partição destacada na Seção 2.4, foram utilizadas múltiplas sequências de quadros de um mesmo vídeo para cada sinal realizado. A utilização deste método se justifica pelo fato de um mesmo sinal poder ser representado por sequencias diferentes de quadros, permitindo uma análise mais significativa das predições do modelo ao longo da execução do sinal pelo intérprete. Deve ser mencionado, porém, que a implementação deste método implica um número desbalanceado de amostras baseado na quantidade de quadros de cada vídeo incluído na partição de testes. Apesar disso, as métricas continuam sendo consideradas representativas da performance da rede no reconhecimento de cada sinal. A Tabela 5 apresenta os valores obtidos para cada métrica, assim como a movimentação de cada sinal, definida na Seção 2.1.

Tabela 5 – Resultados obtidos no teste

Letra	Precisão	Sensibilidade	F-Medida	Estático
A	1	1	1	Sim
B	1	0.667	0.800	Sim
C	0.353	1	0.521	Sim
Ç	0	0	0	Não
D	0.600	1	0.750	Sim
E	0.636	1	0.777	Sim
F	1	0.857	0.922	Sim
G	1	1	1	Sim
H	0	0	0	Não
I	0	0	0	Sim
J	1	0.905	0.950	Não
K	0.348	1	0.516	Não
L	1	1	1	Sim
M	0.789	1	0.882	Sim
N	0.842	1	0.914	Sim
O	0.833	0.385	0.526	Sim
P	0.909	1	0.952	Sim
Q	0.917	1	0.956	Sim
R	1	0.750	0.857	Sim
S	1	0.688	0.815	Sim
T	1	1	1	Sim
U	0.600	1	0.75	Sim
V	1	1	1	Sim
W	0.400	1	0.5714	Sim
X	1	0.333	0.499	Não
Y	0.182	0.222	0.200	Não
Z	0.929	0.929	0.929	Não

Fonte: Elaborada pelo autor.

A princípio, a tabela revela resultados próximos ao ideal em diversos sinais para as três métricas avaliadas. Dos 27 sinais avaliados, aproximadamente 81% obtiveram uma pontuação acima de 0,9 em pelo menos uma das métricas.

Por outro lado, em alguns sinais dinâmicos como “Ç” e “H”, o reconhecimento demonstrou uma maior dificuldade de reconhecimento em geral. Esse resultado é esperado, pois o contexto necessário para reconhecer corretamente estes sinais adiciona uma camada de dificuldade para a rede neural. Entretanto, sinais estáticos como “I” e “O”, demonstraram um nível de reconhecimento extremamente baixo, enquanto outros sinais dinâmicos como “J” e “Z”, cuja detecção é mais complexa dado o contexto adicional, possuíram um grau de reconhecimento bem elevado, contrariando o resultado esperado. Uma possível explicação para essa discrepância é a similaridade entre elementos encontrados em cada sinal. Certos sinais possuem posicionamento semelhante, ou até mesmo idêntico, de diversos pontos de referência e, em casos mais extremos, dependem exclusivamente da movimentação para distingui-los. Para melhor compreender a discrepância entre sinais que possuíram má performance dos demais, é possível recorrer à Matriz de Confusão.

Tabela 6 – Matriz de Confusão

	A	**B**	**C**	**Ç**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0
C	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ç	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0
J	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
K	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	2	0	0	0	0	0
S	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	1	0	0	0	11	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
X	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0	1
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	4	3	0	0	0	0	0	0	0	0	0	0	2	0
Z	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13

Fonte: Elaborada pelo autor.

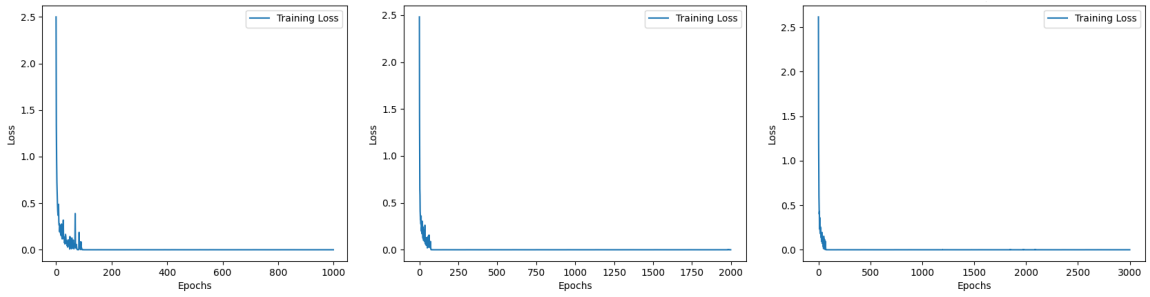
Para certos sinais, é possível perceber um padrão na distribuição dos sinais que não foram reconhecidos corretamente. O sinal “Ç”, por exemplo, foi confundido pela letra “C” em todos os testes realizados. Isso se dá pelo posicionamento seme-

lhante entre todos os pontos de referência compartilhados entre os dois sinais, cuja maior diferença está na presença de movimento no sinal dinâmico. Outros sinais que foram frequentemente tomados completamente por outro sinal, como as letras “H” e “I”, também apresentam diversos pontos semelhantes ao sinal incorretamente identificado.

Além disso, os sinais “S”, “X” e “Y” se destacam pela grande variedade na distribuição de predições feitas pelo modelo, não possuindo uma clara confusão com outro sinal específico. Entre as possíveis causas deste desvio, destacam-se a insuficiência de dados para o treinamento, e o método utilizado para recorte dos vídeos do conjunto de dados. Para verificar este último ponto, fez-se uma revisão manual dos vídeos truncados, limitada àqueles cujos sinais obtiveram má performance nas métricas, porém não foram encontrados erros significativos na seleção dos quadros que representam os sinais selecionados.

Para todos estes resultados, foram utilizadas 2000 épocas de treino, conforme o trabalho de Sundar e Bagyammal (2022), em cima do qual foi baseada a arquitetura do modelo utilizado. Porém, a fim de obter métricas mais completas, foram avaliados os valores obtidos para a perda utilizando 1000 e 3000 épocas. A comparação dos gráficos para todas as quantidades de épocas é apresentada na Figura 18.

Figura 18 – Gráficos de perda por # de épocas.



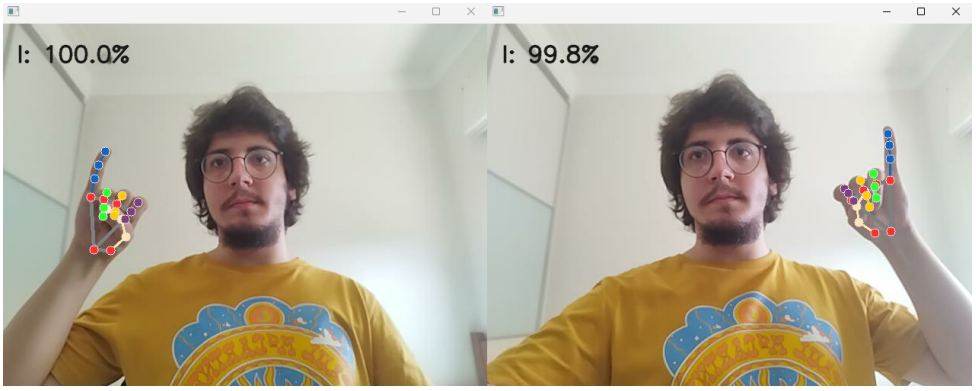
Fonte: Elaborado pelo autor.

Nos gráficos anteriores, é possível reparar uma rápida convergência à um valor de 0 para a perda. Este número se estabiliza por volta de 100 épocas e permanece mínimo pelo resto do treino, indicando uma quantia de épocas extremamente excessiva para a tarefa. Uma possível explicação para essa dissonância entre o número de épocas necessário para o treino deste modelo e a quantia utilizada pelos autores do trabalho original pode estar contida no volume de dados utilizados, visto que o conjunto de dados destes é aproximadamente vinte vezes maior.

4.3 Interface para Entrada de Sinais via *Webcam*

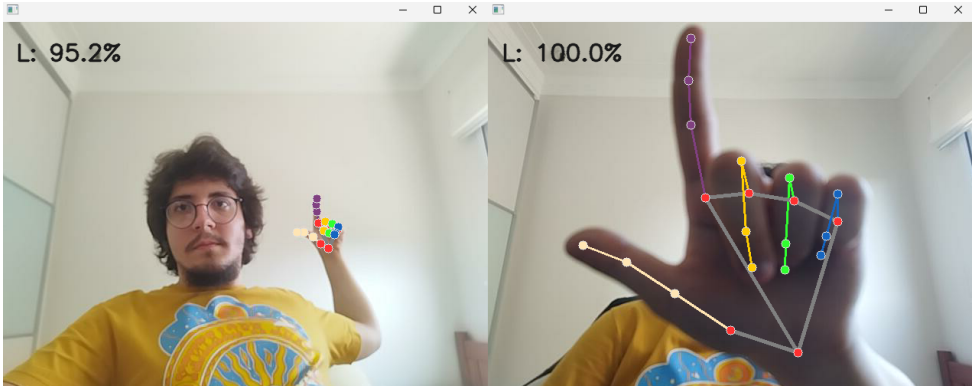
Além dos testes realizados em cima da partição de testes mencionados anteriormente, também foram realizados alguns testes em tempo real utilizando uma aplicação própria. Este protótipo consiste em uma única interface que mostra uma projeção do fluxo de dados da *Webcam*. A posição dos pontos de referência da mão detectada nos últimos n quadros são constantemente armazenados em um *buffer*. O modelo treinado é carregado ao arquivo e aguarda o pressionar da tecla “s” para realizar uma previsão baseada nos dados do *buffer*. No canto superior esquerdo, fica exibido a maior probabilidade prevista (pontuação de confiança) da última chamada feita, e o sinal correspondente. A Figura 19 mostra duas avaliações diferentes do mesmo sinal em cada uma das mãos, em uma tentativa de testar o espelhamento aplicado durante o aumento de dados. Já a Figura 20 mostra casos de teste mais extremos, que envolvem a ampliação e redução do tamanho da mão relativo à tela.

Figura 19 – Teste do reconhecimento para mãos diferentes.



Fonte: Elaborado pelo autor.

Figura 20 – Teste do reconhecimento para distancias diferentes.



Fonte: Elaborado pelo autor.

O reconhecimento dos sinais utilizando esta interface provou-se extremamente capaz, especialmente considerando a falta de familiaridade do modelo com o ambiente. Alguns padrões encontrados nas métricas anteriores se aplicam aos resultados observados no protótipo, como a tendência dos sinais “U” e “W” a obter diversos falsos positivos, indicado pela baixa precisão. Porém, os resultados obtidos utilizando esta aplicação não devem ser considerados tanto quanto os testes anteriores, visto que não houve a presença de sinalizadores fluentes em Libras. Por esse motivo, não foram realizados testes extensivos utilizando as métricas com essa ferramenta.

5 Considerações Finais

O objetivo principal deste trabalho foi a criação de uma ferramenta capaz de detectar e classificar os gestos do alfabeto manual da Língua Brasileira de Sinais, esta que é uma língua que ainda carece de diversos veículos tradicionais de acessibilidade.

É importante ressaltar que o interesse do trabalho em implementar ferramentas de análise automática de Libras é baseado na utilização destas como materiais auxiliares à disseminação da linguagem de sinais, e que a intenção não é a substituição de nenhum profissional qualificado.

Observando os resultados finais obtidos, é clara a capacidade que as redes neurais possuem na categorização de linguagens visuais como Libras, visto que a maioria das letras foram bem reconhecidas. Apesar de constituir apenas uma pequena parcela do dicionário, o sucesso no reconhecimento do alfabeto manual é representativo de uma conquista muito maior em superar as limitações que existiam previamente na análise de dados sequenciais.

Um fator que talvez tenha impedido o modelo de alcançar o seu melhor desempenho se dá pelo baixo volume de dados disponíveis para o aprendizado de Libras, especialmente do alfabeto manual. Essa disparidade é ainda mais evidente quando comparada ao número de materiais em ASL. Dado um número de vídeos maior por sinal do alfabeto, gostaria-se de ter estabelecido uma partição do conjunto de dados dedicada para a validação do modelo. Além disso, o projeto foi desenvolvido com a facilidade de expansão do *dataset* em mente, visto que novas amostras foram sendo coletadas ao longo do projeto. Por isso, nos próximos trabalhos, destaca-se a oportunidade de aumentar a robustez do conjunto de dados, não somente através do número bruto de vídeos no conjunto de treino, mas também pela inclusão de uma maior diversidade no número de sinalizadores, possivelmente através da criação de uma base de dados própria.

Ademais, salienta-se a possibilidade de explorar outros métodos de reconhecimento de imagens envolvendo redes profundas, como o *Visual Transformer* (ViT), *Gated Recurrent Unit* (GRU), ou *Temporal Convolutional Network* (TCN) e comparar a sua capacidade de reconhecimento dos gestos aos resultados obtidos pela rede LSTM neste trabalho.

Por fim, indo além das restrições ao alfabeto manual, existe também a capacidade de expansão do escopo a fim de incorporar múltiplas outras palavras do dicionário de Libras no futuro.

Referências

BRASIL. *Lei nº 10.436, de 24 de abril de 2002: Regulamenta a Língua Brasileira de Sinais - Libras e dá outras providências*. 2002. Disponível em: ⟨https://www.planalto.gov.br/ccivil_03/leis/2002/l10436.htm⟩. Publicada no DOU de 25.4.2002.

CONTRIBUTORS, W. *Feedforward neural network*. 2006. Acesso em: 26/10/2024. Disponível em: ⟨https://en.wikipedia.org/wiki/Feedforward_neural_network#⟩.

CONTRIBUTORS, W. *Recurrent neural network*. 2020. Acesso em: 26/10/2024. Disponível em: ⟨https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=991832590⟩.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. Disponível em: ⟨http://www.deeplearningbook.org⟩.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Disponível em: ⟨https://direct.mit.edu/neco/article-abstract/9/8/1735/6109/Long-Short-Term-Memory⟩.

HUANG, S.; YE, Z. Boundary-adaptive encoder with attention method for chinese sign language recognition. *IEEE Access*, v. 9, p. 70948–70960, 2021. Https://ieeexplore.ieee.org/document/9426906.

IBGE. *PNS 2019: país tem 17,3 milhões de pessoas com algum tipo de deficiência*. 2021. Publicado pela Agência de Notícias do IBGE. Disponível em: ⟨https://agenciadenoticias.ibge.gov.br/agencia-sala-de-imprensa/2013-agencia-de-noticias/releases/31445-pns-2019-pais-tem-17-3-milhoes-de-pessoas-com-algum-tipo-de-deficiencia⟩.

LUCAS, C. *The Sociolinguistics of Sign Languages*. 2001. Acesso em: 26/10/2024. Disponível em: ⟨https://doi.org/10.1017/CBO9780511612824⟩.

MACHADO, M. C. *Classificação automática de sinais visuais da Língua Brasileira de Sinais representados por caracterização espaço-temporal*. 2018. Disponível em: ⟨https://tede.ufam.edu.br/handle/tede/6645⟩.

MACHADO, V. L. V.; WEININGER, M. J. As variantes da língua brasileira de sinais – libras. *Transversal - Revista em Tradução*, v. 4, n. 7, p. 41–65, 2018. Disponível em: ⟨https://repositorio.ufc.br/handle/riufc/38106⟩.

MediaPipe Hands. *Hand landmarks detection guide*. 2024. Acesso em: 26/10/2024. Disponível em: ⟨https://developers.google.com/mediapipe/solutions/vision/hand_landmarker⟩.

MediaPipe Holistic. *Simultaneous Face, Hand and Pose Prediction, on Device*. 2024. Acesso em: 26/10/2024. Disponível em: ⟨https://research.google/blog/

mediapipe-holistic-simultaneous-face-hand-and-pose-prediction-on-device/?hl=es_MX⟩.

MELO, G. F.; OLIVEIRA, P. S. J. Ensino-aprendizagem de libras: mais um desafio para a formação docente. *Boletim Técnico do Senac*, v. 38, n. 3, p. 40–49, 2012. Disponível em: ⟨https://www.bts.senac.br/bts/article/view/155⟩.

MOORES, D. F. The history of language and communication issues in deaf education. In: *The Oxford handbook of deaf studies, language, and education*. [S.l.]: Oxford University Press, 2010. v. 2, p. 17–30. Disponível em: ⟨https://doi.org/10.1093/oxfordhb/9780195390032.013.0002⟩.

REZENDE, T. *Reconhecimento Automático de Sinais da Libras: Desenvolvimento da Base de Dados MINDS-Libras e Modelos de Redes Convolucionais*. 2021. Disponível em: ⟨https://www.researchgate.net/publication/356383630_Reconhecimento_Automatico_de_Sinais_da_Libras_Desenvolvimento_da_Base_de_Dados_MINDS-Libras_e_Modelos_de_Redes_Convolucionais⟩.

RODRIGUES, A. J. *V-LIBRASIL: uma base de dados com sinais na língua brasileira de sinais (Libras)*. 2021. Dissertação (Mestrado em Ciência da Computação), Recife, 2021. Disponível em: ⟨https://libras.cin.ufpe.br⟩.

SARMENTO, A. H. d. A. *Integração de Datasets de Vídeo para Tradução Automática da Libras com Aprendizado Profundo*. 2023. 104 f. Dissertação (Mestrado), Curso de Computação, ICMC, Universidade de São Paulo, São Carlos, 2023. Disponível em: ⟨https://www.teses.usp.br/teses/disponiveis/55/55137/tde-10012024-093541/pt-br.php⟩.

SCHLINDWEIN, A. F.; AQUINO, A. *Aspectos Sintáticos da Libras*. 2021. Acesso em: 26/10/2024. Disponível em: ⟨https://cesad.ufs.br/ORBI/public/uploadCatalago/14461125082022LIBRA_-_Aula_08.pdf⟩.

SILVA, D. R. B. d. *Uma arquitetura multifluxo baseada em aprendizagem profunda para reconhecimento de sinais em libras no contexto de saúde*. 2020. Disponível em: ⟨https://repositorio.ufpb.br/jspui/handle/123456789/21163⟩.

SUNDAR, B.; BAGYAMMAL, T. American sign language recognition for alphabets using mediapipe and lstm. *Procedia Computer Science*, Elsevier BV, v. 215, p. 642–651, 2022. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S1877050922021378⟩.

TAVARES, K. C. d. A.; OLIVEIRA, A. P. P. d. Libras no ensino de inglês mediado pelas novas tecnologias: desafios e possibilidades. *Revista Brasileira de Linguística Aplicada*, Faculdade de Letras - Universidade Federal de Minas Gerais, v. 14, n. 4, p. 1045–1072, Oct 2014. ISSN 1984-6398. Disponível em: ⟨https://doi.org/10.1590/1984-639820145631⟩.

ZUHAIB, M. *Demystifying the Confusion Matrix Using a Business Example*. 2019. Acesso em: 26/10/2024. Disponível em: ⟨https://towardsdatascience.com/demystifying-confusion-matrix-29f3037b0cfa⟩.