Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Reconhecimento de Sinais do Alfabeto de Libras por Meio de Aprendizado Profundo

Rosa Maria Ottoni Fernandes Maciel

JUIZ DE FORA
SETEMBRO, 2024

Reconhecimento de Sinais do Alfabeto de Libras por Meio de Aprendizado Profundo

Rosa Maria Ottoni Fernandes Maciel

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Luiz Maurílio da Silva Maciel

JUIZ DE FORA

SETEMBRO, 2024

Reconhecimento de Sinais do Alfabeto de Libras por Meio de Aprendizado Profundo

Rosa Maria Ottoni Fernandes Maciel

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Luiz Maurílio da Silva Maciel
Doutor em Engenharia de Sistemas e Computação

Marcelo Caniato Renhe
Doutor em Engenharia de Sistemas e Computação

Rodrigo Luís de Souza da Silva
Doutor em Engenharia Civil

JUIZ DE FORA
27 DE SETEMBRO, 2024

Aos meus pais, irmãos, filho e amigos.

Ao meu marido, pelo apoio e sustento.

Resumo

Cerca de 1% da população brasileira apresenta alguma deficiência auditiva, sendo necessário ter meios que auxiliem a comunicação dessas pessoas. No Brasil, tem-se a Língua Brasileira de Sinais (Libras), que é a forma de comunicação padrão dos deficientes auditivos. Como a Libras consiste em uma forma de comunicação visual, surge a possibilidade de aplicar visão computacional para reconhecer gestos dessa língua. Neste trabalho realiza-se o reconhecimento de alguns sinais do alfabeto de Libras utilizando a rede neural YOLOv8. Para o treinamento e avaliação da rede foi organizado um conjunto de dados composto por 1.664 imagens extraídas de um repositório público e 252 imagens capturadas e anotadas especificamente para esse trabalho. O treinamento da rede foi realizado variando alguns parâmetros para encontrar a melhor configuração. Os resultados indicaram que a maioria das classes alcançaram altas taxas de acerto. Porém, algumas classes apresentaram dificuldades para detecção causadas por limitação de dados e similaridade de gestos.

Palavras-chave: Aprendizado profundo; Detecção de objetos; Identificação de sinais; Língua de sinais; Redes neurais; Visão computacional.

Abstract

About 1% of the Brazilian population has some form of hearing impairment, making it necessary to have tools that assist these individuals in communication. In Brazil, there is the Brazilian Sign Language (Libras), which is the standard form of communication for hearing-impaired people. Since Libras consists of a visual form of communication, there is the possibility of applying computer vision to recognize gestures from this language. In this work, some signs from the Libras alphabet are recognized using the YOLOv8 neural network. For the training and evaluation of the network, a dataset was organized comprising 1,664 images extracted from a public repository and 252 images specifically captured and annotated for this work. The network was trained by varying some parameters to find the best configuration. The results indicated that most classes achieved high accuracy rates. However, some classes faced detection difficulties due to data limitations and gesture similarity.

Keywords: Deep learning; Object detection; Signals identification; Sign language; Neural networks; Computer vision.

Agradecimentos

Primeiramente, agradeço ao meu marido e orientador Luiz Maciel, pelo amor, paciência e apoio incondicional ao longo dessa jornada. Às vezes, foram suas palavras de encorajamento que me mantiveram firme nos momentos mais desafiadores.

Aos meus pais Maria e Aloysio, por sempre acreditarem no meu potencial e me incentivarem desde o início. Vocês são a base de tudo que conquistei.

À minha irmã Rafaela, pelo carinho e companheirismo, sempre presente para compartilhar risos e oferecer conselhos quando eu mais precisava.

Ao meu filho Luiz Miguel, que me dá forças todos os dias e me inspira a ser uma pessoa melhor. Que você cresça sabendo que este trabalho também é uma vitória sua.

E, claro, ao meu querido gato Luigi, cuja presença tranquila e reconfortante tornou as longas horas de estudo mais leves e acolhedoras.

Aos meus professores, pela paciência, dedicação e orientação valiosa ao longo de todo essa jornada. Suas contribuições foram fundamentais para o desenvolvimento deste projeto e para o meu crescimento acadêmico.

Aos meus colegas e amigos, que compartilharam momentos de desafios e conquistas. Agradeço pelas conversas, pela motivação e por todo o apoio durante essa caminhada. Vocês tornaram este período mais leve e inspirador.

A todos que, direta ou indiretamente, contribuíram para a conclusão deste trabalho, meu mais sincero agradecimento.

"Tu te tornas eternamente responsável por aquilo que cativas".

Saint-Exupéry (O Pequeno Príncipe)

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

AP	Average Precision
ASL	American Sign Language
BSL	British Sign Language
COCO	Common Objects in Context
CSL	Chinese Sign Language
CNN	Convolutional Neural Networks
FN	False Negatives
FP	False Positives
GSL	German Sign Language
HEIF	High Efficiency Image File
HOG	Histogram of Oriented Gradients
HSV	Hue Saturation Value
IBGE	Instituto Brasileiro de Geografia e Estatística
IoU	Intersection over Union
JPEG	Joint Photographic Experts Group
LGP	Língua Gestual Portuguesa
Libras	Língua Brasileira de Sinais
LSA	Lengua de Señas Argentina
LSF	Langue des Signes Française
mAP	mean Average Precision
ReLU	Rectified Linear Unity
RNNs	Recurrent Neural Network
SLN	Sign Language of Netherlands
TP	True Positives
VOC	Visual Object Classes
YOLO	You Only Look Once
ZIM	Zernike Invariant Moments

1 Introdução

A Libras é a Língua Brasileira de Sinais, sendo a principal forma de comunicação de pessoas com algum de tipo de deficiência auditiva. A língua consiste em comunicar-se por meio de gestos e expressões faciais que podem ser observados por meio da visão. De acordo com a Lei n° 10.436 de 24 de abril de 2002 (BRASIL, 2002), a Libras foi considerada como meio legal de comunicação e expressão. A mesma lei exige que o governo, as instituições públicas de saúde e educação devem garantir o atendimento e a inclusão dos portadores de deficiência auditiva. Tendo em vista que a Libras é uma língua muito importante na sociedade e que muitas pessoas não conseguem compreender os sinais, surge a necessidade de criar meios para facilitar a comunicação.

1.1 Justificativa/Motivação

De acordo com o Instituto Brasileiro de Geografia e Estatística (IBGE)(IBGE, 2021), cerca de 2,3 milhões de brasileiros possuem alguma deficiência auditiva, representando assim 1,1% da população. Uma grande dificuldade enfrentada é a falta de pessoas que compreendam e se comuniquem em Libras, tornando-se uma língua restrita. Cerca de 22,4% das pessoas com deficiência auditiva entre 5 a 40 anos conhecem Libras, mostrando-se a importância da criação de um sistema que auxilie na difusão da língua.

Como a língua utiliza-se de sinais realizados com o movimento das mãos, posicionadas em alguma parte do corpo e também pode ser representada por expressões faciais, espera-se que possa ser reconhecida por meio de imagens e vídeo através de visão computacional. A visão computacional tem se destacado na solução de problemas de reconhecimento facial (KAUR et al., 2020), classificação de expressões faciais (LI; DENG, 2020) e reconhecimento de gestos (TURK; ATHITSOS, 2020). Nesse contexto, o aprendizado profundo se destaca, visto que muitos trabalhos na atualidade estão utilizando esta abordagem. Com isso, observa-se a possibilidade da criação de uma aplicação voltada para reconhecimento de Libras por meio de visão computacional e aprendizado profundo.

Uma ferramenta voltada para o reconhecimento de Libras por meio de visão computacional pode contribuir de várias maneiras para as pessoas deficientes auditivas. Uma delas seria para uma comunicação mais acessível, possibilitando a comunicação entre pessoas surdas e ouvintes. Outra maneira seria no acesso a serviços públicos, auxiliando a servidores na compreensão da necessidade da pessoa com deficiência auditiva.

1.2 Descrição do Problema

O problema abordado nesse trabalho consiste em, dada uma imagem de uma pessoa realizando gestos de Libras, reconhecer os gestos utilizando o aprendizado profundo. Mais especificamente, este trabalho pretende, dada uma imagem de uma pessoa realizando os sinais de algumas letras do alfabeto em Libras, localizar e classificar a letra representada.

1.3 Objetivos

O objetivo geral desse trabalho consiste em desenvolver um método que reconheça sinais do alfabeto em Libras em imagens através de aprendizado profundo. Os objetivos específicos são:

- Coletar e realizar anotação de imagens de sinais do alfabeto em Libras;
- Estudar e implementar métodos de aprendizado profundo para detecção de objetos em imagens;
- Treinar modelos de aprendizado profundo para detectar sinais do alfabeto de Libras em imagens;
- Analisar os resultados do treinamento nas imagens coletadas.

2 Fundamentação Teórica

Este capítulo descreve os fundamentos necessários para o embasamento teórico desse trabalho. Ele é dividido em quatro seções. A Seção 2.1 aborda sobre Libras, a Seção 2.2 sobre redes neurais convolucionais, a Seção 2.3 aborda sobre a detecção de objetos e, por fim, a Seção 2.4 trata sobre a rede neural utilizada neste trabalho.

2.1 Libras

A Libras é a língua de comunicação dos surdos brasileiros. Da mesma forma que existem diferentes línguas faladas no mundo, a línguas de sinais também é específica para cada país (CRISTIANO, 2017). Seguem alguns exemplos de línguas de sinais de alguns países:

- Língua Gestual Portuguesa (LGP);
- Língua de Sinais Neerlandesa (*Sign Language of Netherlands* – SLN);
- Língua de Sinais Americana (*American Sign Language* – ASL);
- Língua de Sinais Argentina (*Lengua de Señas Argentina* – LSA);
- Língua de Sinais Britânica (*British Sign Language* – BSL);
- Língua de Sinais Francesa (*Langue des Signes Française* – LSF);
- Língua de Sinais Alemã (*German Sign Language* – GSL);
- Língua de Sinais Chinesa (*Chinese Sign Language* – CSL).

Em Libras, existem 5 parâmetros importantes que estruturam os sinais (CRISTIANO, 2018):

1. Configuração da(s) mão(s): consiste no posicionamento que a mão dominante assume durante a realização do sinal. Segundo Martins (2020) existem mais de 110 configurações. A Figura 2.1 mostra exemplos de configuração da(s) mão(s).

Figura 2.1: Exemplos de configuração da(s) mão(s) (SOUZA; MONTEIRO, 2006).

2. Ponto ou local de articulação: representa o local do corpo onde será realizado o sinal. A mão pode estar tocando alguma parte do corpo ou posicionada à frente do corpo. A Figura 2.2 apresenta exemplos de ponto ou local de articulação.

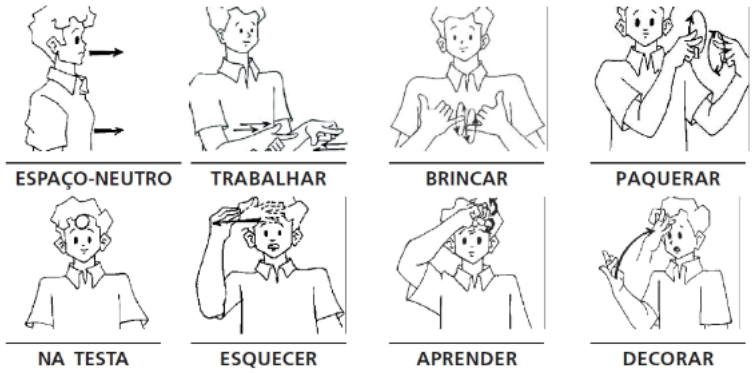


Figura 2.2: Exemplos de ponto ou local de articulação (SOUZA; MONTEIRO, 2006).

3. Movimento: o sinal pode ser estático ou apresentar algum movimento. A movimentação pode ser realizada de várias formas como: linear, semicircular, circular, helicoidal, sinuoso e angular. Tem-se também diferentes sentidos como: para frente, à esquerda e à direita. A Figura 2.3 mostra exemplos de sinais com movimento.

Figura 2.3: Exemplos de sinais com movimento (SOUZA; MONTEIRO, 2006).

4. Orientação: é a orientação da palma da mão como, por exemplo, para cima, para baixo, para trás e para frente. A Figura 2.4 apresenta exemplos de sinais com diferentes orientações.

Figura 2.4: Exemplos de sinais com diferentes orientações (SOUZA; MONTEIRO, 2006).

5. Expressão facial e/ou corporal: são componentes que utilizam-se de expressões faciais, movimento da cabeça e ombros, olhares e linguagem corporal. A Figura 2.5 mostra exemplos de sinais com expressão facial e/ou corporal.

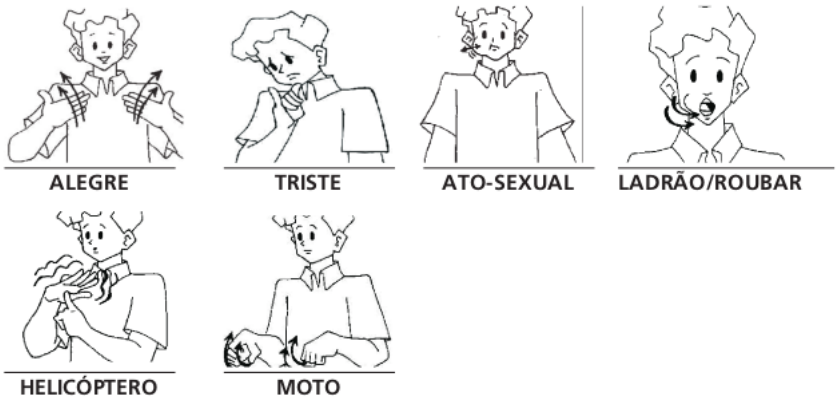


Figura 2.5: Exemplos de sinais com expressão facial e/ou corporal (SOUZA; MONTEIRO, 2006).

Percebe-se que a alteração de um único parâmetro pode diferenciar o significado do sinal.

Para este trabalho optou-se por utilizar os gestos que representam as letras do alfabeto. A Figura 2.6 apresenta os gestos do alfabeto em Libras. É importante notar que as letras H, J, K, X, Y e Z apresentam movimentos ao serem representadas.

2.2 Redes Neurais Convolucionais

De acordo com Goodfellow, Bengio e Courville (2016), redes neurais convolucionais (*Convolutional Neural Networks* – CNN) são um tipo de rede neural que processa dados em

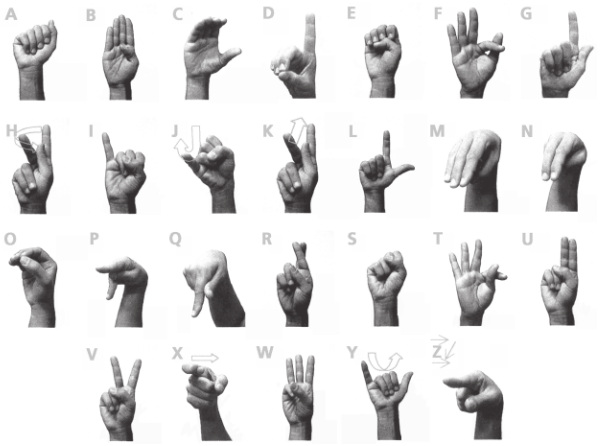


Figura 2.6: Alfabeto em Libras (SOUZA; MONTEIRO, 2006).

forma de grade, como imagens. As redes convolucionais utilizam a operação matemática convolução, que é um tipo de operação linear.

A operação de convolução pode ser definida matematicamente pela equação:

$$s(t) = (x * w)(t), \tag{2.1}$$

onde x é a imagem de entrada (*input*), w é o filtro ou núcleo (*kernel*) e a saída (*output*) s é o mapa de características (*feature map*).

No caso de imagens, utiliza-se a operação de convolução discreta que é descrita pela equação:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{2.2}$$

Em aprendizado de máquina, utiliza-se como dados de entrada uma matriz multidimensional, e o filtro são os parâmetros que serão aprendidos pela rede neural. Normalmente assume-se que as funções sejam 0 em todas as posições e possua valores em um conjunto finito de posições. Dessa forma, permite-se que a soma infinita seja realizada com uma quantidade finita de valores na prática (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 2.7 apresenta um exemplo de convolução em 2D.

Segundo Goodfellow, Bengio e Courville (2016), uma camada de uma rede convolucional possui 3 estágios. O primeiro estágio consiste em aplicar diversas convoluções gerando um conjunto de ativações lineares. Essas convoluções são adicionadas a um valor de viés (*bias*). O segundo estágio envolve a aplicação de uma função de ativação não

linear sobre as ativações lineares. Uma função de ativação muito utilizada nas CNNs é a unidade linear retificada (*Rectified Linear Unity* – ReLU). No último estágio, para alterar a saída da camada, utiliza-se a função de agrupamento (*pooling*).

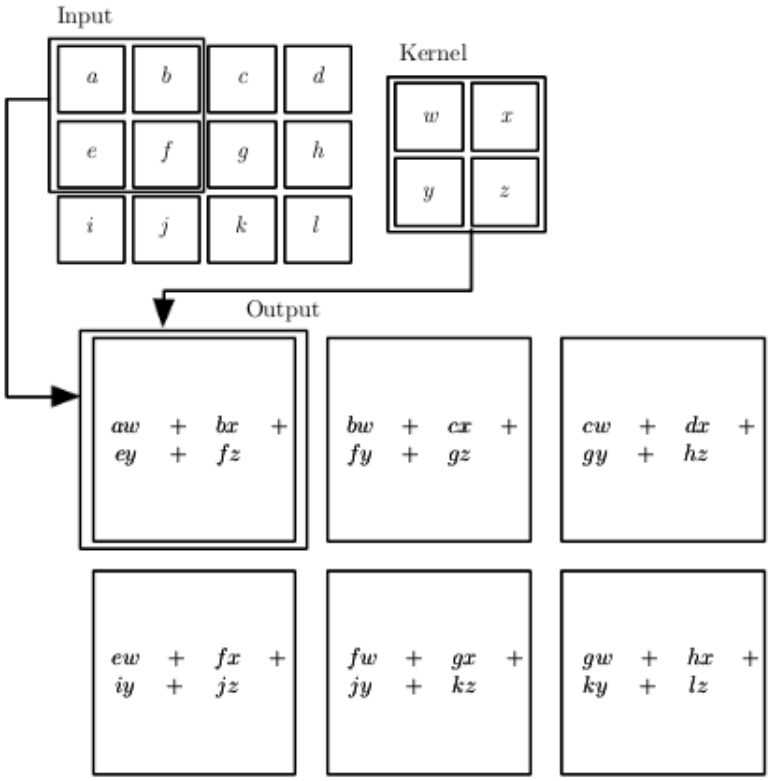


Figura 2.7: Exemplo de convolução em 2D (GOODFELLOW; BENGIO; COURVILLE, 2016).

A função de *pooling* retorna uma estatística de determinada região. Uma das mais utilizadas é o *max pooling* que resulta no valor máximo da vizinhança retangular. Outras funções de *pooling* utilizadas são: *min pooling*, *average pooling* e L2 *pooling*. Os elementos da última camada agrupada são vetorizados e usados como entrada para camadas totalmente conectadas (*fully connected*) que são responsáveis pela classificação da entrada (GONZALEZ; WOODS, 2018). A Figura 2.8 mostra um exemplo de rede neural convolucional.

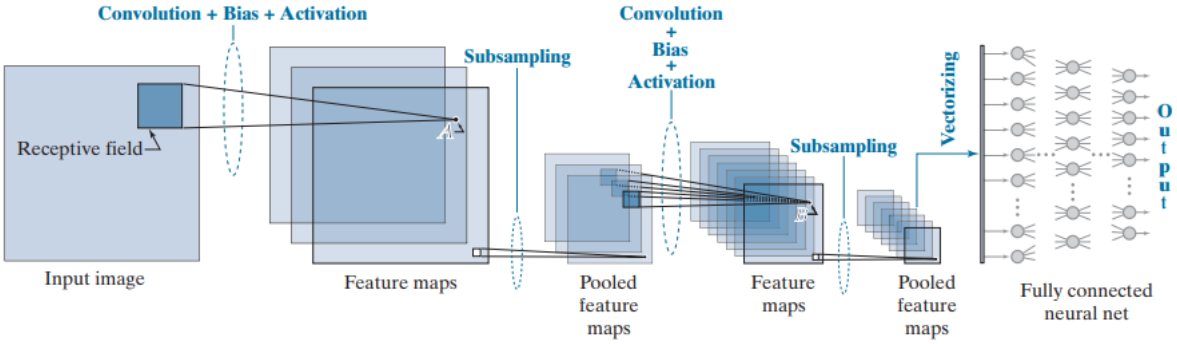


Figura 2.8: Exemplo de rede neural convolucional (GONZALEZ; WOODS, 2018).

No treinamento de redes neurais, os pesos são ajustados para corrigir os erros na saída de forma que correspondam ao valor esperado. Nesse processo, o erro é definido como uma função de perda e a correção é realizada a partir do gradiente dessa função. Normalmente, os pesos são ajustados com base no erro acumulado para determinado número de amostras. Esse número de amostras é chamado de tamanho do lote (*batch size*). Quando todas as amostras do conjunto de treino são inferidas na rede, completa-se uma época de treinamento. O tamanho do lote e o número de épocas são hiperparâmetros a serem ajustados para cada problema (SZELISKI, 2022).

Outro hiperparâmetro importante é a taxa de aprendizado (*learning rate*). Essa taxa define a proporção da correção dos pesos em relação ao gradiente. Valores altos da taxa de aprendizado indicam uma correção maior, mas podem atrapalhar na convergência. Por outro lado, valores baixos representam passos menores, podendo levar a uma convergência lenta. Normalmente, o valor da taxa de aprendizado precisa ser definido experimentalmente (GONZALEZ; WOODS, 2018).

Em geral, o treinamento de redes neurais envolve três conjuntos de dados: treino, validação e teste. O conjunto de treino é utilizado para ajustar os pesos da rede, de modo a mapear as amostras do conjunto nas respectivas saídas esperadas. O conjunto de teste é formado por dados que não foram utilizados no treinamento e servem para avaliar o desempenho do modelo treinado. O conjunto de validação também é diferente do conjunto de treino e pode ser utilizado durante o treinamento para tomar decisões, como selecionar o melhor modelo (SZELISKI, 2022).

2.3 Detecção de Objetos

A detecção de objetos tem como objetivo localizar e classificar instâncias de objetos em uma imagem. Esses objetos devem pertencer a classes predefinidas, como animais, pessoas, veículos, objetos, etc. Os objetos a serem detectados podem ter diferentes escalas, orientações e perspectivas na imagem. A detecção busca localizar caixas delimitadoras (*bounding boxes*) que delimitam o objeto e determinar a classe desse objeto (AMIT; FELZENSZWALB; GIRSHICK, 2020).

As caixas delimitadoras são retângulos que definem os limites dos objetos presentes na imagem. Uma forma de definir uma caixa delimitadora é especificar as coordenadas do canto superior esquerdo e as dimensões (largura e altura) que determinam a localização e o tamanho do objeto na imagem (AMIT; FELZENSZWALB; GIRSHICK, 2020). A Figura 2.9 ilustra um exemplo de caixa delimitadora.

Figura 2.9: Exemplo de caixa delimitadora, na localização da letra A em Libras.

Conjuntos de dados são necessários para o treinamento e avaliação de modelos de detecção de objetos. Alguns dos conjuntos mais utilizados para detecção de propósito geral são:

- PASCAL VOC (*Visual Object Classes*): possui em seu conjunto de dados mais de 10 mil imagens, divididas em 20 classes e com anotações de caixas delimitadoras (EVERINGHAM et al., 2010);

- COCO (*Common Objects in Context*): contém em seu conjunto de dados 328 mil imagens, distribuídas em 91 classes e com anotações de caixas delimitadoras (LIN et al., 2014);

- ImageNet: apresenta mais de 14 milhões de imagens, contendo mais de 21 mil classes, mas nem todas as imagens possuem anotações de caixas delimitadoras (DENG et al., 2009).

Os principais modelos de detecção de objetos são treinados nos conjuntos PASCAL VOC ou COCO. Para problemas específicos, como é o caso do presente trabalho, é necessário criar um conjunto de dados próprio, com anotações dos objetos de interesse. Os modelos pré-treinados nos conjuntos PASCAL VOC ou COCO podem ser usados como ponto de partida para o treinamento de modelos para tarefas específicas.

2.4 *You Only Look Once* (YOLO)

A YOLO foi apresentada pela primeira vez por Joseph Redmon em 2016, sendo um dos algoritmos mais populares e eficientes para detecção de objetos. A YOLO se destaca entre outras formas por abordar a detecção como um problema de regressão direta. Ela possui a capacidade de prever simultaneamente as caixas delimitadoras e as probabilidades das classes de cada objeto em apenas uma passagem pela rede neural (REDMON et al., 2016).

O funcionamento da YOLO, consiste em dividir a imagem de entrada em uma grade de $S \times S$ células. Ela prediz um conjunto de caixas delimitadoras e classes de objetos para cada célula. Associa-se uma pontuação de confiança para cada caixa delimitadora, que indica a precisão de que a caixa envolva um objeto e a probabilidade do objeto pertencer à classe determinada. A execução dessa tarefa em uma única passagem pela imagem torna a detecção eficiente (REDMON et al., 2016). A Figura 2.10 apresenta como é realizado o processo de detecção da YOLO.

Uma das vantagens da YOLO é a realização de previsões em tempo real. Dessa forma, a YOLO torna-se uma alternativa adequada para um sistema de reconhecimento de sinais de Libras. Ao longo dos últimos anos, a YOLO apresentou diferentes versões, sendo que as versões iniciais apresentavam algumas limitações de precisão em relação a

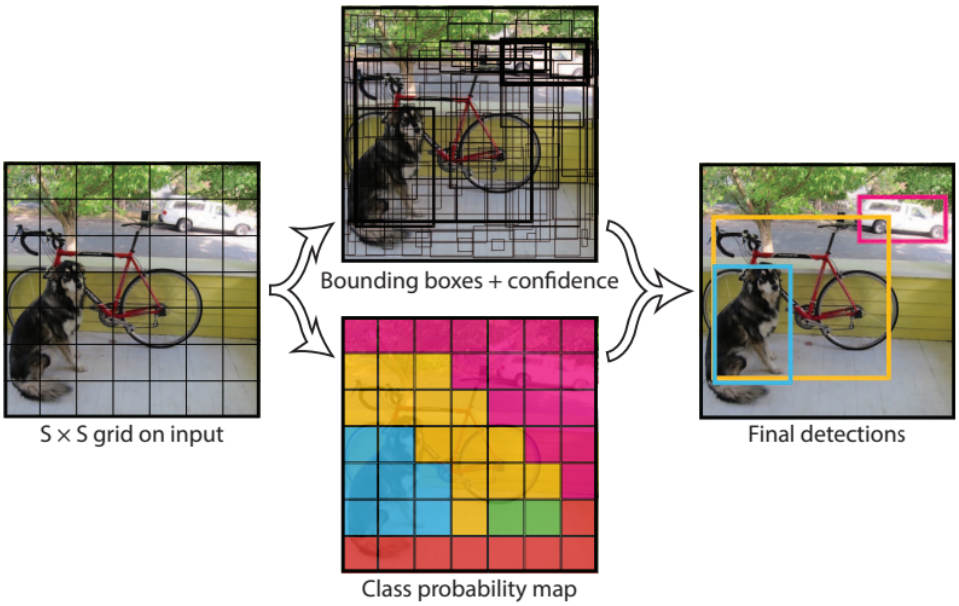


Figura 2.10: Processo de detecção da YOLO (REDMON et al., 2016).

métodos mais lentos. Atualmente, a YOLO encontra-se na versão 10, sendo que a versão 8 é a mais estável. Dessa forma, a versão 8 foi a escolhida para o presente trabalho.

3 Revisão Bibliográfica

Este capítulo apresenta uma revisão bibliográfica sobre trabalhos que utilizaram visão computacional para o reconhecimento de línguas de sinais. O capítulo se divide em duas seções: a Seção 3.1 trata o reconhecimento de outras línguas de sinais e a Seção 3.2 aborda o reconhecimento de Libras.

3.1 Reconhecimento de Outras Línguas de Sinais

Daroya, Peralta e Naval (2018) propuseram identificar os sinais do alfabeto da ASL por meio de aprendizado profundo. O método envolveu a utilização da rede neural DenseNet para a classificação dos sinais. Eles utilizaram 24 sinais do alfabeto, excluindo as letras J e Z, pois elas tem movimento. Dessa forma, o trabalho se limitou a classificar os sinais das letras sem movimento.

Taskiran, Killioglu e Kahraman (2018) desenvolveram um sistema em tempo real para o reconhecimento da ASL por meio de aprendizado profundo. O método do trabalho consiste primeiramente, em extrair da imagem a região onde se tem pele, em seguida usa-se a CNN para classificar os gestos a partir das imagens. Com isso, o modelo é treinado com o conjunto de dados e realiza as previsões em tempo real para cada *frame* do vídeo. Eles utilizaram os 26 sinais do alfabeto, mais os 10 dígitos, totalizando 36 sinais.

Ananthanarayana et al. (2021) abordaram em seu trabalho métodos de aprendizado profundo para a tradução das línguas de sinais GSL, ASL e CSL. O método proposto envolveu CNN e o modelo de transformador (*transformer*) para codificar e traduzir a língua de sinais. Os autores utilizaram várias características de entrada como: articulações do corpo e dedos, pontos faciais e vetores de redes neurais convolucionais. Além disso, vários modelos de tradução automática são empregados, abrangendo sequências para sequência, redes mais complexas com atenção e aprendizado por reforço. Eles utilizaram um conjunto de dados composto por 1.078 palavras de GSL, 1.457 expressões e falas de ASL e 1.000 expressões e falas de CSL, totalizando 3.535 sinais.

Li, Yan e Du (2022) utilizam tecnologias de visão computacional e aprendizado profundo para reconhecer língua de sinais, focando em melhorar a comunicação de pessoas surdas. Os autores realizaram a detecção de 30 sinais da CSL. Duas abordagens foram estudadas: CNN+LSTM e detecção com a YOLOv5. A YOLOv5 se destacou por sua detecção rápida de gestos das mãos, sendo mais promissora para aplicações de tradução em tempo real.

3.2 Reconhecimento de Libras Usando Visão Computacional

Koroishi e Silva (2015) propuseram investigar uma estratégia baseada em probabilidades para reconhecer sinais de Libras usando um sensor RGB-D. A identificação dos sinais é fundamentada na classificação semântica da língua. Os autores utilizaram Kinect de XBOX para capturar as imagens e realizaram a segmentação da mão. O movimento é calculado pela diferença na posição da mão direita entre dois *frames*. Eles avaliaram o sistema em 39 sinais. Esse sistema tem como limitação a necessidade de um sensor de profundidade.

Bastos (2016) apresentou uma forma de identificar gestos e utilizou Libras como aplicação prática. Utilizou-se dos descritores Histograma de Gradientes Orientados (*Histogram of Oriented Gradients* – HOG) e Momentos Invariantes de Zernike (*Zernike Invariant Moments* – ZIM) para extrair características das imagens. O classificador Perceptron Multicamada foi utilizado para classificar os gestos. A proposta foi avaliada em 40 sinais estáticos. Esse método não foi avaliado em sinais com movimento.

Rezende (2016) abordou em seu trabalho o reconhecimento automático de expressões faciais na Libras. Utilizou-se de um sensor RGB-D para captura das imagens. O método envolveu detecção e recorte da região facial, a condensação dos vídeos, a criação de vetores de características com base nos pontos faciais e textura e os classificadores de sinais k-NN e SVM. A autora avaliou 10 sinais de Libras. Essa proposta se limitou a reconhecer apenas expressões faciais e depende de sensores RGB-D.

Silva et al. (2020) desenvolveu um modelo para reconhecimento de sinais de Libras

relacionados à área da saúde, baseado somente em imagens ou vídeos. Em seu modelo eles empregaram CNNs e Redes Neurais Recorrentes (*Recurrent Neural Network* – RNNs) para o processamento de dados visuais com dependência temporal. Além disso, é adotada uma arquitetura de múltiplos fluxos que integra informações de imagens RGB, fluxos óticos e *keypoints* para o reconhecimento dos sinais de Libras na área da saúde. Em seu trabalho 50 sinais foram avaliados.

4 Método Proposto

Este capítulo descreve o método proposto neste trabalho para a detecção de sinais do alfabeto de Libras. A Seção 4.1 apresenta o processo para a construção do conjunto de dados. A Seção 4.2 detalha o procedimento para treinamento e avaliação do modelo de detecção. A Figura 4.1 apresenta uma visão geral do método proposto.

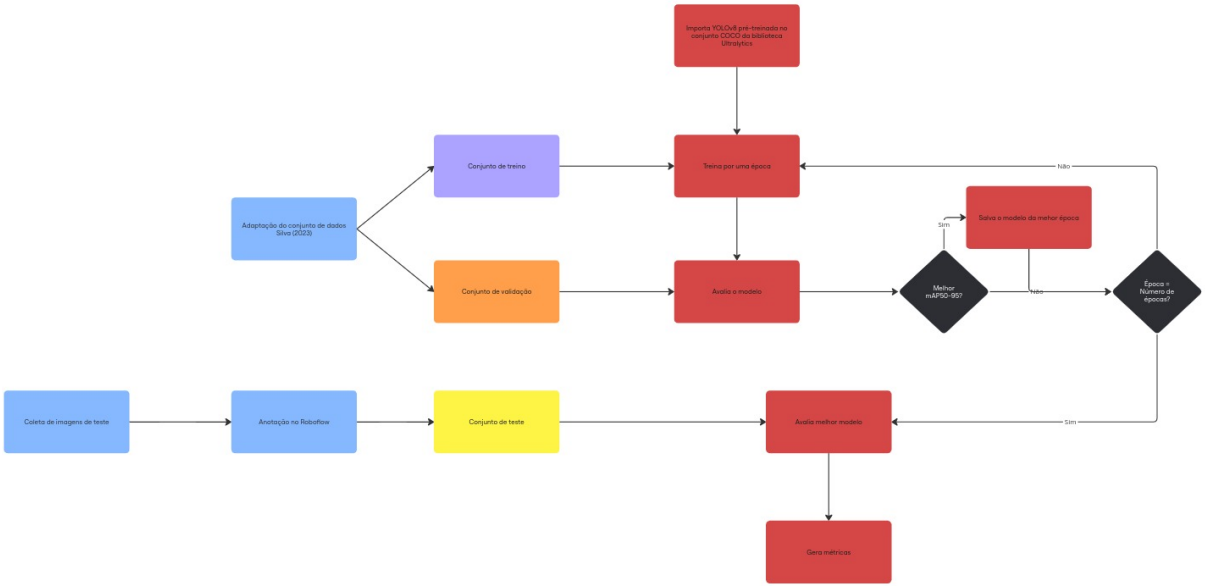


Figura 4.1: Visão geral do método proposto.

4.1 Construção do Conjunto de Dados

Esta seção descreve o processo de construção do conjunto de dados utilizado para treinar o modelo de reconhecimento de sinais do alfabeto de Libras. A Subseção 4.1.1 apresenta os conjuntos de treino e validação, que foram obtidos a partir de um conjunto de dados já existente. A Subseção 4.1.2 descreve o conjunto de teste, que foi construído especificamente para o presente trabalho.

4.1.1 Conjuntos de Treino e Validação

Em pesquisa realizada na plataforma Roboflow[1], encontrou-se um conjunto de dados proposto por Silva (2023), composto por 1.733 imagens, com resolução de 3.264 *pixels* de largura e 2.448 *pixels* de altura. As imagens estavam divididas em 22 classes, conforme a Tabela 4.1. Observa-se que nem todas as letras foram representadas, não sendo incluídas as letras que possuem movimentos, exceto a letra K, que foi configurada de forma estática. Outra observação é que a letra D foi representada de duas formas distintas, D1 e D2. A configuração D1 é representada com o gesto de forma que o dedo indicador esteja para trás e o D2 o dedo indicador virado para frente. Identificou-se quatro pessoas distintas realizando os gestos, porém, em algumas imagens apareceu apenas a mão representando o gesto, não sendo possível identificar se eram pessoas diferentes. Originalmente, o conjunto de dados estava dividido em três subconjuntos: treino (70%), validação (20%) e teste (10%).

Tabela 4.1: Distribuição do conjunto de dados proposto por Silva (2023).

Classe	**Número de Imagens**	**Percentual do Total**	**Classe**	**Número de Imagens**	**Percentual do Total**
A	83	4,79%	M	77	4,44%
B	92	5,31%	N	79	4,56%
C	68	3,92%	O	81	4,67%
D1	20	1,15%	P	90	5,19%
D2	69	3,98%	Q	95	5,48%
E	95	5,48%	R	75	4,33%
F	91	5,25%	S	76	4,39%
G	97	5,60%	T	85	4,90%
I	72	4,15%	U	82	4,73%
K	73	4,21%	V	82	4,73%
L	79	4,56%	W	72	4,15%

Para o presente trabalho, o conjunto de dados proposto por Silva (2023) sofreu algumas alterações para ser utilizado como conjunto de treino e validação. Removeu-se a classe D2, pois optou-se em captar a forma D1 para o conjunto de teste. Essa escolha foi motivada pela configuração D1 ser a mais utilizada. Após a remoção da classe D2, o conjunto de dados passou a ter 1.664 imagens, que foram divididas aleatoriamente em dois subconjuntos: treino (85%) e validação (15%). Esses dois subconjuntos foram utilizados

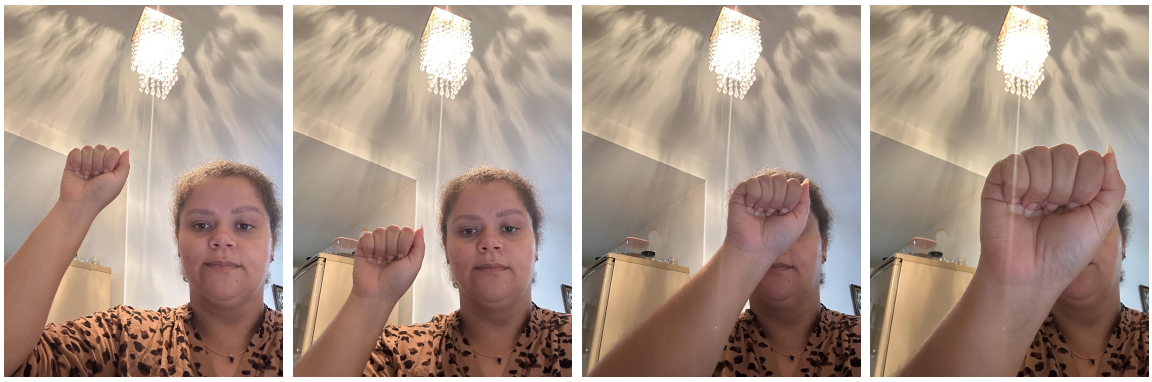
[1]https://roboflow.com/

para treinar a rede YOLOv8.

4.1.2 Conjunto de Teste

Para a construção do conjunto de teste deste trabalho, foram envolvidas três pessoas distintas para realizar a representação do alfabeto em Libras. Os gestos foram capturados por meio de um *smartphone* iPhone 14 Plus, através da câmera frontal. O *smartphone* foi posicionado sobre um suporte em cima de uma mesa, e a pessoa representando o gesto ficou sentada em uma cadeira de frente para a câmera. Definiu-se quatro posições para a representação de cada gesto, sendo elas:

1. Gesto acima do ombro, mais afastado da cabeça (Figura 4.2a);
2. Gesto acima do ombro, mais próximo da cabeça (Figura 4.2b);
3. Gesto em frente ao rosto, longe da câmera (Figura 4.2c);
4. Gesto em frente ao rosto, próximo da câmera (Figura 4.2d).



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

Figura 4.2: Representações da letra A, em cada uma das posições. (a) Gesto acima do ombro, mais afastado da cabeça. (b) Gesto acima do ombro, mais próximo da cabeça. (c) Gesto em frente do rosto, longe da câmera. (d) Gesto em frente do rosto, próximo da câmera.

Foram representadas as mesmas letras que o conjunto de dados proposto por Silva (2023), excluindo a classe D2. Ao todo foram coletadas 252 imagens, cada classe sendo representada por 12 imagens, 4 imagens por pessoa, com resolução de 2.316 *pixels* de largura e 3.088 *pixels* de altura. As imagens foram convertidas do formato HEIF (*High Efficiency Image File*) para JPEG (*Joint Photographic Experts Group*) e renomeadas

indicando a letra representada e o número na sequência de imagens da classe (por exemplo, A_1.jpg, A_2.jpg e assim por diante).

Após serem renomeadas, realizou-se a importação das imagens para a plataforma Roboflow. Cada uma das imagens foi anotada, demarcando uma caixa delimitadora em torno do gesto realizado pela mão e indicando a classe. A Figura 4.3 representa um exemplo de como é realizada a anotação no Roboflow. Ao finalizar as anotações das imagens, o conjunto de teste foi exportado no formato da YOLOv8. A criação de um conjunto de teste específico teve como objetivo evitar viés na avaliação, analisando o desempenho da rede em imagens de pessoas diferentes daquelas que aparecem nos conjuntos de treinamento e teste.

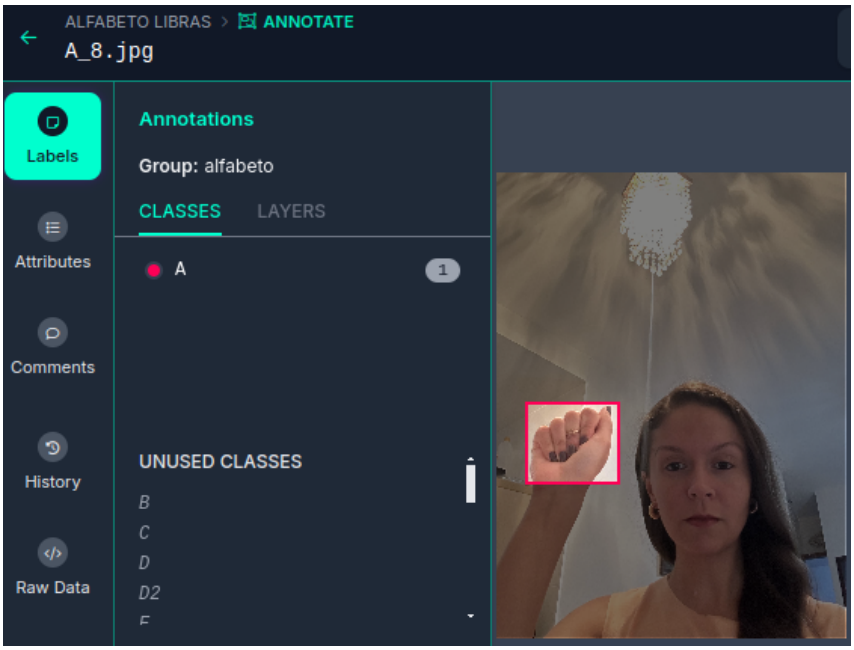


Figura 4.3: Exemplo de anotação no Roboflow.

4.2 Treinamento e Avaliação da YOLOv8

Para a realização dos experimentos foi utilizada a implementação oficial da YOLOv8 disponibilizada pela Ultralytcs[2]. A implementação da rede é baseada no *framework* PyTorch (PASZKE et al., 2019), sendo utilizado para a construção, treinamento e inferência dos modelos. O treinamento foi realizado a partir de modelos pré-treinados no conjunto de dados COCO.

[2]https://www.ultralytics.com/pt

Construiu-se um *script* na linguagem *Python* que realiza o treinamento do modelo no conjunto de treino durante um certo número de épocas. A cada época, o modelo foi avaliado no conjunto de validação. Caso ocorra uma melhora na métrica de avaliação, o modelo é salvo. Ao alcançar o número de épocas, o melhor modelo é avaliado no conjunto de dados de teste para gerar as métricas de avaliação. O Capítulo 5 detalha as configurações, métricas de avaliação e resultados obtidos nos experimentos.

5 Experimentos e Resultados

Este capítulo aborda os experimentos e resultados obtidos na pesquisa. São apresentados em detalhes os procedimentos realizados e as análises obtidas. A Seção 5.1 descreve como foi realizada a configuração dos experimentos. As principais métricas utilizadas para avaliação são apresentadas na Seção 5.2. Na Seção 5.3 apresenta-se os resultados quantitativos obtidos. A Seção 5.4 realiza uma análise qualitativa dos resultados.

5.1 Configuração dos Experimentos

No treinamento da YOLOv8, a taxa de aprendizado começa com um valor LR_0 e decai linearmente até uma taxa final $LR_f = 0,01 \cdot LR_0$. O valor padrão de LR_0 é 0,01. Nos experimentos executados neste trabalho realizou-se a variação da taxa LR_0, como forma de analisar o impacto nos resultados. Adicionalmente, o tamanho do lote, o número de épocas e o tamanho da entrada foram hiperparâmetros ajustados durante os experimentos deste trabalho. Essas varições de parâmetros serão apresentadas de maneira mais clara na Seção 5.3. A máquina utilizada para rodar os experimentos apresenta as seguintes configurações: processador Intel Core i7-8700K 3.70GHz, 16 GB de RAM e placa de vídeo NVIDIA GeForce GTX 1650 4GB.

Utilizou-se os mesmos padrões de aumento de dados (*data augmentation*) do modelo original, que são:

- *Flip*: Inverte as imagens na orientação horizontal ou vertical, que auxilia no espelhamento dos objetos.
- Escalamento: Ocorre o redimensionamento das imagens para tamanhos diferentes, simulando os objetos em diferentes escalas.
- Translação: Simula mudanças de posições dos objetos, de forma que as imagens se desloquem em diferentes direções.

- Rotação: Gira as imagens em diferentes ângulos, permitindo variar a orientação dos objetos.

- Variação do HSV (*Hue Saturation Value*): Altera os valores de matiz, saturação e valor das imagens para que possa simular variações de iluminação e cores.

- Mosaico: Combina quatro imagens diferentes em apenas uma imagem, aumentando a diversificação dos dados e contribuindo para a melhoria na detecção de objetos em diferentes cenários.

- Mistura: Combina duas imagens de forma ponderada, mesclando os *pixels* e as anotações.

- Apagamento: Corta de maneira aleatória uma região da imagem e troca por uma cor preta ou branca, simulando oclusões.

- Perspectiva: Realiza transformações perspectivas nas imagens, gerando distorções para simular diferentes posições de visualização.

- Cisalhamento: Distorce as imagens ao inclinar ou esticar em um determinado eixo, que auxilia a lidar com deformações.

- Recorte: Seleciona uma região no interior da imagem e escala para o tamanho original.

A Tabela 5.1 apresenta os parâmetros de aumento de dados padrão da YOLOv8. Na realização de testes deste trabalho variou-se alguns desses valores, o que será discutido melhor na Seção 5.3.

5.2 Métricas de Avaliação

As métricas utilizadas para avaliar os experimentos realizados foram: precisão, revocação (*recall*) e mAP (*mean Average Precision*).

A métrica de precisão busca medir a proporção de predições corretas em relação ao total de predições positivas realizadas pelo modelo. A precisão está associada à pergunta:

Tabela 5.1: Parâmetros de aumento de dados padrão da YOLOv8 (ULTRALYTICS, 2024).

Parâmetro	**Padrão**	**Variação**	**Descrição**
`hsv_h`	0,015	0,0 - 1,0	Percentual máximo de variação da matiz.
`hsv_s`	0,7	0,0 - 1,0	Percentual máximo de variação da saturação.
`hsv_v`	0,4	0,0 - 1,0	Percentual máximo de variação do valor.
`degrees`	0,0	-180 - +180	Ângulo máximo de rotação.
`translate`	0,1	0,0 - 1,0	Percentual máximo de translação.
`scale`	0,5	$\geq$ 0,0	Valor máximo de escalamento.
`shear`	0,0	-180 - +180	Ângulo máximo de cisalhamento.
`perspective`	0,0	0,0 - 0,001	Valor máximo da intensidade da distorção da imagem.
`flipud`	0,0	0,0 - 1,0	Probabilidade de *flip* vertical.
`fliplr`	0,5	0,0 - 1,0	Probabilidade de *flip* horizontal.
`mosaic`	1,0	0,0 - 1,0	Probabilidade de aplicar o mosaico.
`mixup`	0,0	0,0 - 1,0	Probabilidade de aplicar a mistura.
`erasing`	0,4	0,0 - 0,9	Probabilidade de aplicar o apagamento.
`crop_fraction`	1,0	0,1 - 1,0	Fração da imagem recortada.

"Quantas amostras são realmente positivas, dentre aquelas que foram classificadas como positivas pelo modelo?". A precisão é definida pela equação:

$$P = \frac{TP}{TP + FP}, \tag{5.1}$$

onde,

- Verdadeiros Positivos (*True Positives* – TP): Número de amostras que o modelo previu corretamente como positivas.
- Falsos Positivos (*False Positives* – FP): Número de amostras que o modelo previu incorretamente como positivas.

A métrica de revocação visa medir a proporção de predições corretas em relação ao total de amostras positivas do conjunto. A revocação responde à seguinte pergunta: " Quantas das amostras verdadeiramente positivas, o modelo previu corretamente?". A revocação é dada pela fórmula:

$$R = \frac{TP}{TP + FN}, \tag{5.2}$$

onde,

- Falsos Negativos (*False Negatives* – FN): Número de amostras que o modelo deveria ter previsto como positivas, mas não foram identificadas.

Em detecção de objetos, a determinação se uma detecção está correta é realizada utilizando a métrica de IoU (*Intersection over Union*). A IoU avalia a similaridade entre duas caixas delimitadoras, sendo a primeira a caixa detectada pelo modelo e a segunda a caixa que representa a localização real do objeto (*ground truth*). O cálculo da IoU é dado pela razão da área de interseção sobre área de união. O valor de IoU varia de 0 a 1, sendo 0 quando não há sobreposição entre as caixas e 1 quando as caixas se sobrepõem completamente. A Figura 5.1 ilustra o cálculo da IoU. Para determinar se uma detecção está correta verifica-se se o IoU é maior que um limiar (por exemplo 0,5).

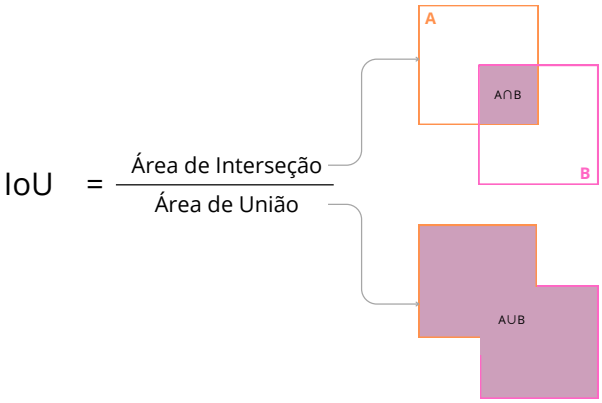


Figura 5.1: Cálculo da IoU.

A mAP é uma das principais métricas utilizadas na avaliação de modelos para problemas de detecção de objetos. O cálculo dessa métrica é realizado através da média da precisão média (*Average Precision* – AP) para todas as classes do conjunto de dados. Com a combinação de precisão e revocação, a mAP busca medir quanto um modelo de detecção identifica e localiza os objetos nas imagens. Para o cálculo da mAP, primeiramente obtém-

se a AP que é dada pela área sob a curva de precisão-revocação, refletindo a precisão média durante as variações de níveis de revocação. Por fim, a mAP é a média das APs de todas as classes.

Existem dois tipos principais de cálculos da mAP. A mAP50, introduzida pelo conjunto de dados PASCAL VOC, utiliza o limiar de IoU de 0,5 para indicar uma detecção correta. A segunda forma de cálculo mAP50-95 foi apresentada pelo conjunto de dados COCO. O cálculo utiliza a média de múltiplos limiares de IoU, com variação de 0,5 a 0,95 com incrementos de 0,05.

5.3 Resultados Quantitativos

Está seção apresenta os principais resultados obtidos com a YOLOv8 para o problema de detecção de alguns sinais do alfabeto de Libras. Para cada experimento realizado, avaliou-se o desempenho do modelo no conjunto de validação ao longo do treinamento e escolheu-se o modelo da melhor época com base na métrica da mAP50-95. O melhor modelo selecionado é avaliado no conjunto de teste para gerar as métricas reportadas. Dessa forma, busca-se evitar viés nos resultados.

Inicialmente, utilizou-se os valores padrões dos seguintes hiperparâmetros:

- Tamanho do lote: 16;
- Taxa de aprendizado inicial (LR_0): 0,01;
- Tamanho da imagem de entrada: 640 $\times$ 640 *pixels*.

Optou-se por utilizar a versão YOLOv8n que possui menos parâmetros, por questões de limitações *hardware*.

A primeira análise realizada foi para identificar o melhor número de épocas para treinar o modelo. Avaliou-se 5 números de épocas distintos: 20, 40, 60, 80 e 100. Foram calculados a precisão, revocação, mAP50 e mAP50-95. Adicionalmente, com base na saída de mAP50-95 no conjunto de validação, identificou-se qual a melhor época selecionada. A Tabela 5.2 apresenta os resultados no conjunto de teste. Pode-se observar na coluna melhor época, que a variação foi de 15 a 38, ou seja, a melhor época sempre ocorreu antes

da época de número 40. Dessa forma, optou-se por utilizar 40 épocas para os próximos experimentos. A utilização de 40 épocas também permite a execução mais rápida do treinamento.

Tabela 5.2: Resultados no conjunto de teste para diferentes números de épocas.

Número de épocas	**Melhor época**	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
20	17	0,716	0,655	0,757	0,520
40	15	0,721	0,541	0,679	0,488
60	36	0,638	0,657	0,718	0,494
80	28	0,713	0,630	0,756	0,534
100	38	0,819	0,641	0,777	0,560

A segunda avaliação após a escolha do melhor número de épocas, foi variar a taxa de aprendizado inicial LR_0. Os valores avaliados foram: 0,1; 0,01; 0,005; 0,001; 0,0001. A Tabela 5.3 mostra os resultados obtidos no conjunto de teste. O melhor resultado analisando todas as métricas é de uma taxa de aprendizado de 0,001. Observa-se que as outras taxas de aprendizado obtiveram os mesmos valores para todas as métricas. Esse comportamento pode ser atribuído a uma convergência do modelo utilizando as demais taxas. Realizou-se mais de uma repetição de cada teste para confirmar essa convergência. Dessa forma, adotou-se a taxa de 0,001 para os próximos experimentos.

Tabela 5.3: Resultados no conjunto de teste para diferentes taxas de aprendizado.

Taxa de aprendizado	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
0,1	0,721	0,541	0,679	0,488
0,01	0,721	0,541	0,679	0,488
0,005	0,721	0,541	0,679	0,488
0,001	**0,772**	**0,584**	**0,707**	**0,514**
0,0001	0,721	0,541	0,679	0,488

Após a avaliação da taxa de aprendizado, analisou-se a variação dos tamanhos de lote. Os valores utilizados foram: 2, 4, 8, 16. A Tabela 5.4 descreve os resultados obtidos. Observa-se que o lote de tamanho 8 apresentou melhor resultado em três métricas (revocação, mAP50 e mAP50-95). O lote de tamanho 2 obteve a maior precisão, mas as outras métricas não foram tão boas quanto o lote de tamanho 8. O lote de tamanho 4 alcançou a revocação e mAP50 com valores iguais ao do lote de tamanho 8, porém a

precisão e o mAP50-95 foram inferiores. O lote de tamanho 16, que é o valor padrão da YOLOv8, não obteve o melhor resultado em nenhuma das métricas analisadas.

Tabela 5.4: Resultados no conjunto de teste para diferentes tamanhos de lote.

Tamanho do lote	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
2	**0,797**	0,669	0,757	0,527
4	0,696	**0,695**	**0,766**	0,528
8	0,709	**0,695**	**0,766**	**0,560**
16	0,772	0,584	0,707	0,514

Outro experimento realizado foi a variação no tamanho da imagem de entrada. Apesar do tamanho de lote 8 ser o melhor resultado analisado, por questões de restrições de *hardware*, foi necessário reduzir o tamanho do lote para 2. O tamanho da imagem deve ser múltiplo de 32, por isso utilizou-se as seguintes variações: 640, 768, 960 e 1280. A Tabela 5.5 apresenta os resultados obtidos. Nota-se que a métrica precisão para o tamanho de imagem 960 foi o melhor resultado, porém as outras três métricas não foram tão satisfatórias. Em relação ao tamanho da imagem de 640, que é o valor padrão da YOLOv8, observa-se que as métricas de revocação, mAP50 e mAP50-95 obtiveram os melhores resultados em comparação com os outros tamanhos de imagem. Portanto, para os próximos experimentos foi mantido o tamanho de imagem 640 e o tamanho de lote 8, que foi o melhor na análise anterior.

Tabela 5.5: Resultados no conjunto de teste para diferentes tamanhos de imagem de entrada.

Tamanho da imagem	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
640	0,797	**0,669**	**0,757**	**0,527**
768	0,735	0,662	0,749	0,520
960	**0,811**	0,667	0,745	0,521
1280	0,791	0,618	0,751	0,512

Novos experimentos foram realizados para avaliar separadamente algumas variações do aumento de dados. Utilizou-se o tamanho da imagem 640, tamanho do lote 8, $LR_0 = 0,001$ e 40 épocas. Foram feitos 5 experimentos, variando os seguintes parâmetros:

1. `fliplr`: modificado de 0,5 para 0,0 que significa desligado;

2. `mosaic`: tem o valor padrão de 1,0 que significa a probabilidade de 100% de aplicar o mosaico e foi alterado para 0,0;

3. `degrees`: o valor de ângulo de rotação padrão é 0° e foi modificado para 15°;

4. `perspective`: ajustado de 0,0 para o valor máximo da intensidade de distorção permitida que é 0,001;

5. `erasing`: tem como valor padrão 0,4 de probabilidade de apagamento da imagem, sendo alterado para 0,0.

A Tabela 5.6 apresenta os resultados no conjunto de dados de teste para diferentes aumentos de dados. Analisando os resultados, observa-se que o parâmetro `erasing` foi o que apresentou o melhor resultado em duas métricas: revocação e mAP50-95. Entretanto, nota-se que são os mesmos resultados obtidos com o aumento padrão (Tabela 5.4). Outros dois parâmetros em destaque são o `fliplr` e `perspective`, que apresentaram o melhor resultado para precisão e mAP50, respectivamente. As variações dos parâmetros `mosaic` e `degrees` não alcançaram bons resultados para nenhuma das métricas. Dessa forma, optou-se por manter os padrões de aumento de dados.

Tabela 5.6: Resultados no conjunto de teste para diferentes aumentos de dados.

Parâmetro	**Valor**	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
`fliplr`	0,0	**0,797**	0,542	0,711	0,492
`mosaic`	0,0	0,756	0,563	0,704	0,487
`degrees`	15	0,756	0,606	0,755	0,469
`perspective`	0,001	0,728	0,690	**0,777**	0,494
`erasing`	0,0	0,709	**0,695**	0,766	**0,560**

O último experimento realizado visou avaliar versões maiores da YOLOv8. Além da versão YOLOv8n, foram avaliadas as versões YOLOv8s e YOLOv8m, que possuem um número maior de parâmetros. Na versão YOLOv8m não foi possível realizar o teste com tamanho de lote igual a 8 por questões de *hardware*. Com isso, utilizou-se o tamanho do lote igual a 4. A Tabela 5.7 apresenta os resultados no conjunto de teste para cada versão da YOLOv8. Observa-se que o número de parâmetros da YOLOv8s comparado com a YOLOv8n é mais que o triplo. Quando compara-se a YOLOv8n com a YOLOv8m a quantidade de parâmetros é 8 vezes maior. Pela análise das métricas, nota-se que

a YOLOv8s é a que obteve melhor resultado para precisão, revocação e mAP50. A YOLOv8n apresentou o melhor resultado apenas para o mAP50-95 e a YOLOv8m não conseguiu superar nenhuma das métricas quando comparada às outras duas versões. O desempenho ruim da YOLOv8m pode ser explicado pela redução do tamanho do lote. Futuramente, pode-se realizar experimentos com essa versão e outras maiores utilizando um *hardware* com mais recursos.

Tabela 5.7: Resultados no conjunto de teste para diferentes versões da YOLOv8.

Versão	**Tamanho do lote**	**Número de parâmetros**	**Precisão**	**Revocação**	**mAP50**	**mAP50-95**
YOLOv8n	8	3,2M	0,709	0,695	0,766	**0,560**
YOLOv8s	8	11,2M	**0,758**	**0,726**	**0,775**	0,546
YOLOv8m	4	25,9M	0,736	0,667	0,748	0,526

Para avaliar o desempenho do modelo por classe, foi gerada uma matriz de confusão normalizada para a versão YOLOv8s. Seguiu-se a implementação padrão para o cálculo da matriz de confusão, que avalia as detecções com confiança acima de 0,25 e um IoU de 0,45 para classificar detecções como verdadeiras. A Figura 5.2 apresenta a matriz de confusão normalizada para a YOLOv8s. Observa-se que para as classes B e Q o teste obteve 100% de acerto. Outro ponto a ser destacado são as 8 classes que obtiveram mais de 80% de acerto: A (92%), E (83%), F (83%), G (83%), L (92%), M (92%), R (92%) e U (83%). Por outro lado, pode-se observar dois destaques negativos, que foram as classes D e N, as quais não tiveram nenhum acerto. A classe D possui apenas 20 imagens no conjunto de dados de treino, o que pode ter dificultado o aprendizado dessa classe. Na classe N, observou-se uma confusão com as classes M e Q, o que pode ser justificado pela similaridade entre os gestos. Como pode ser observado na Figura 5.3, esses gestos são realizados com a mão sendo apontada para o mesmo sentido.

5.4 Resultados Qualitativos

Esta seção aborda uma visão qualitativa dos resultados da detecção. São apresentados exemplos que ilustram casos onde a rede detectou corretamente as letras e casos onde as letras não foram detectadas ou detectadas incorretamente.

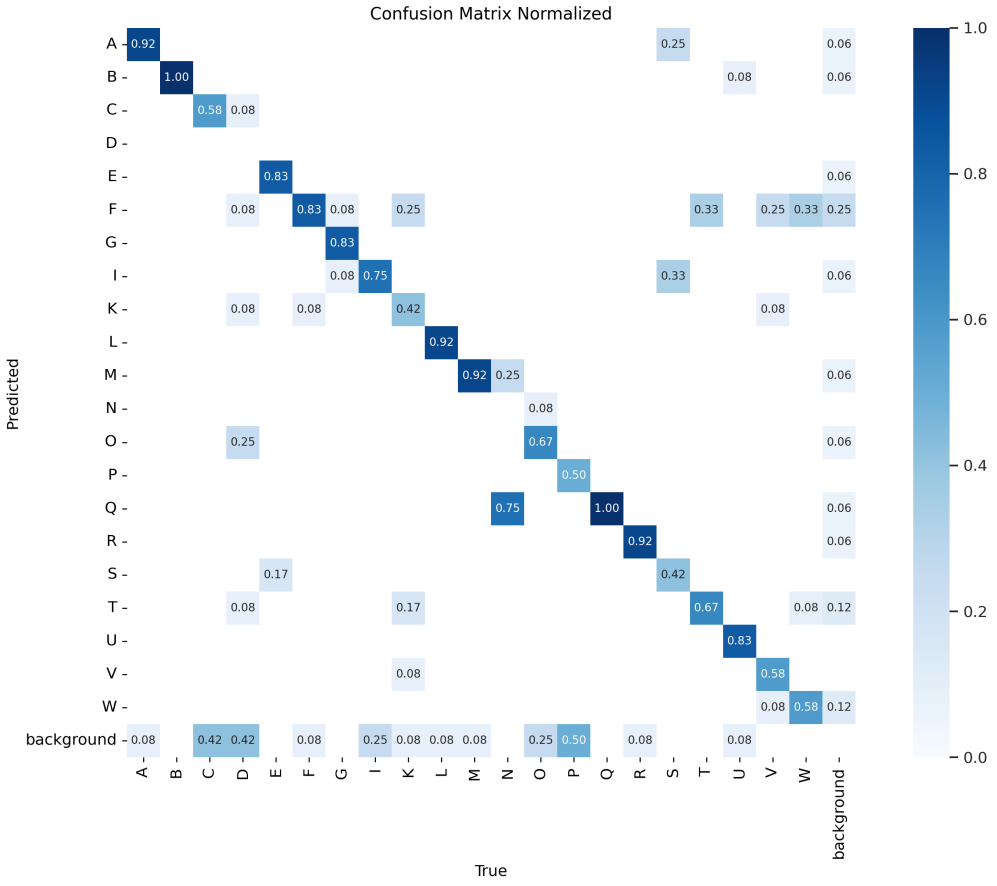
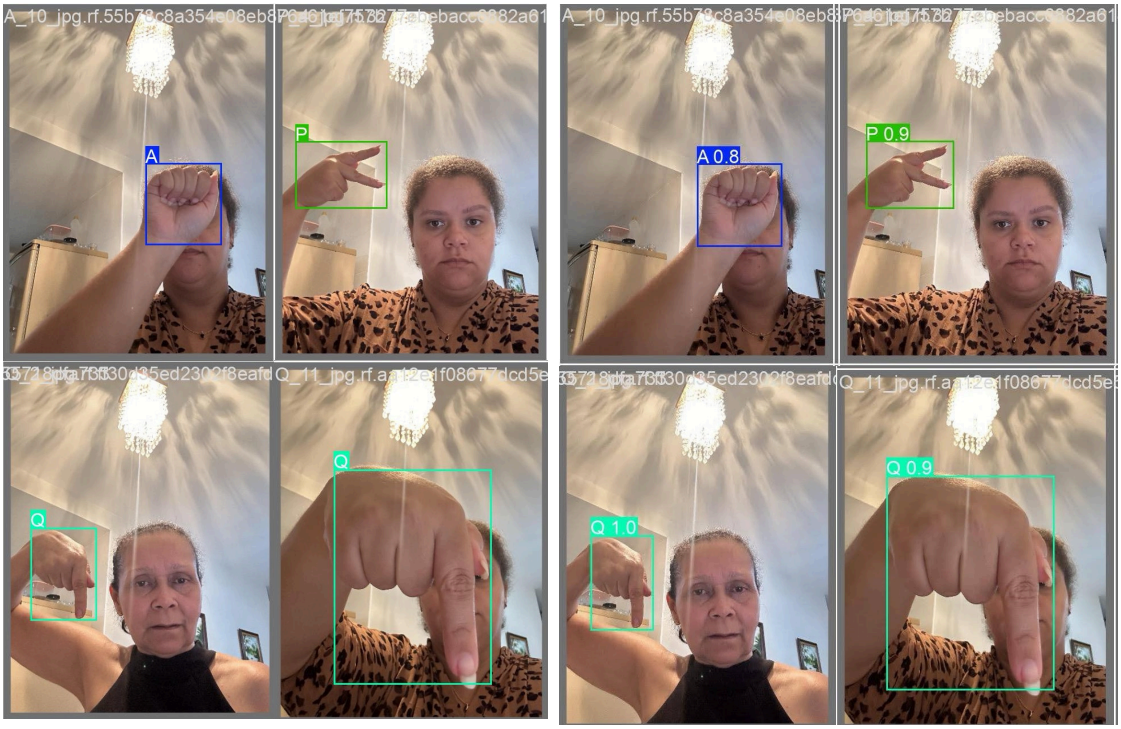


Figura 5.2: Matriz de confusão normalizada para YOLOv8s.

Figura 5.3: Similaridade dos gestos das classes M, N e Q (SOUZA; MONTEIRO, 2006).

A Figura 5.4 apresenta exemplos de predições corretas, ou seja, letras que foram adequadamente localizadas e classificadas. Observa-se que a letra A, representada na configuração em frente ao rosto, longe da câmera, obteve uma confiança de 80%, mostrando a capacidade do modelo em detectar os gestos mesmo em frente ao rosto. As letras P e Q (inferior direita na Figura 5.4b), com configuração acima do ombro mais afastado da cabeça e gesto em frente ao rosto e próximo à câmera, respectivamente, apresentaram uma confiança de 90%. A letra Q (inferior esquerda na Figura 5.4b), com a configuração do gesto acima do ombro e mais próximo à cabeça, apresentou um excelente resultado, acertando o gesto com 100% de confiança. Adicionalmente, observa-se que as caixas de-

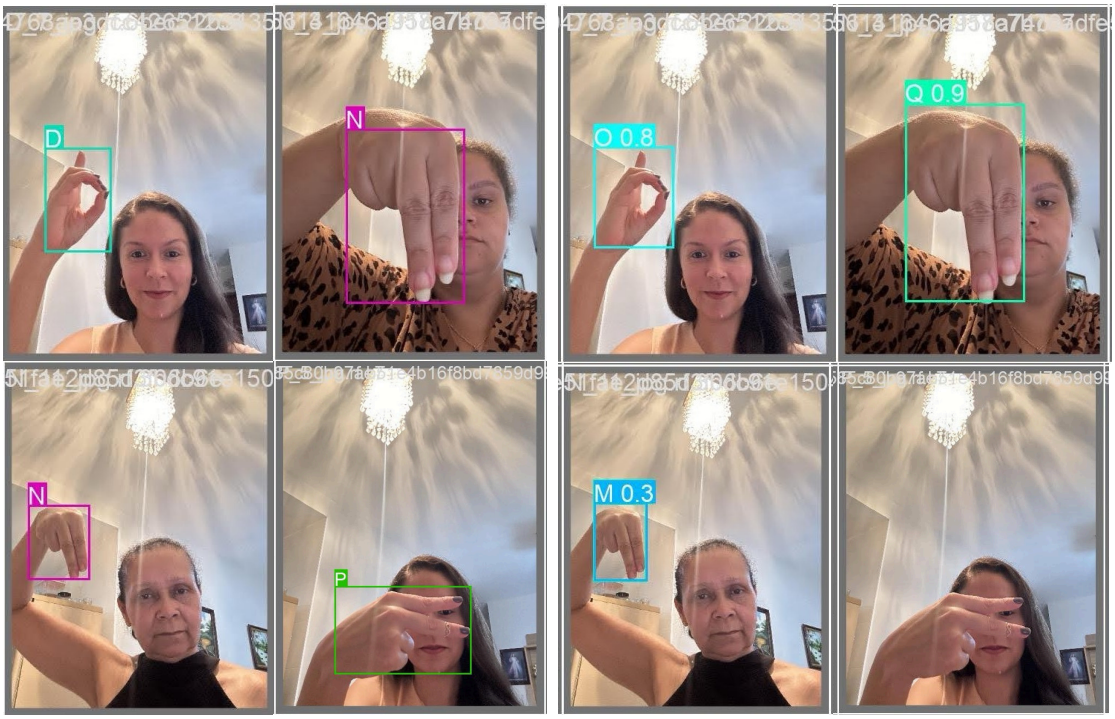
limitadoras detectadas apresentam formas similares ao *ground truth*, indicando uma boa capacidade de localização da YOLOv8.



(a) *Ground truths.* (b) Predições da YOLOv8.

Figura 5.4: Exemplos de predições corretas.

A Figura 5.5 mostra exemplos de predições incorretas, ou seja, gestos que foram confundidos por outros ou não foram reconhecidos. Observa-se que a letra D foi confundida com a letra O, com 80% de confiança. Essa confusão pode ter ocorrido por serem gestos bem parecidos. Outro ponto a ser levado em consideração é que a letra D, em seu conjunto de treino, continha apenas 20 imagens, o que dificultou o aprendizado desse gesto. Na análise da letra N, pode-se observar que ela foi confundida com duas outras letras, M e Q. A similaridade entre essas letras foi discutida anteriormente na Seção 5.3 e ilustrada na Figura 5.3. Finalmente, nota-se que a letra P não foi reconhecida. Isso pode ser explicado pelo gesto ter sido realizado em frente ao rosto. Apesar de ter sido possível identificar os gestos em frente ao rosto em alguns casos, esse tipo de configuração representa um desafio para a rede, devido ao baixo contraste entre a mão e o rosto.



(a) *Ground truths*. (b) Predições da YOLOv8.

Figura 5.5: Exemplos de predições incorretas.

6 Conclusão

Dado que cerca de 2,3 milhões de brasileiros possuem alguma deficiência auditiva, tornam-se importantes meios que possam auxiliar essas pessoas a se comunicarem. Nesse contexto, este trabalho buscou avaliar uma rede neural para a identificação de alguns sinais do alfabeto de Libras.

Realizou-se primeiro a construção do conjunto de dados. Os conjuntos de treino e validação foram extraídos da plataforma Roboflow, a partir da adaptação de um conjunto existente. Para o conjunto de teste, foram capturadas 252 imagens com gestos realizados em 4 configurações diferentes por 3 pessoas distintas. As imagens foram cuidadosamente anotadas e exportadas para o formato da YOLO. Esse conjunto de dados é uma contribuição desse trabalho, pois pode ser utilizado por outras pessoas futuramente.

O conjunto de dados foi utilizado para o treinamento da YOLOv8. Realizaram-se diferentes tipos de testes, para alcançar a melhor configuração para a rede. Chegou-se à conclusão de que rodar por 40 épocas a versão YOLOv8s, utilizando um tamanho de lote 8 e taxa de aprendizado inicial de 0,001 alcançou o melhor resultado para o modelo. Notou-se altas taxas de acerto para a maioria das classes. Porém, identificou-se algumas limitações, devido à limitação de dados e similaridade de alguns gestos.

Como trabalhos futuros, pretende-se realizar experimentos com uma melhor configuração de *hardware*, ampliar o conjunto de dados, incluir letras com movimentos e desenvolver uma aplicação que possa utilizar o modelo em um contexto prático. A longo prazo, pode-se expandir a aplicação para outros tipos de gestos em Libras.

Bibliografia

AMIT, Y.; FELZENSZWALB, P.; GIRSHICK, R. Object detection. *Computer Vision: A Reference Guide*, Springer, p. 875–883, 2020.

ANANTHANARAYANA, T. et al. Deep learning methods for sign language translation. *ACM Transactions on Accessible Computing (TACCESS)*, ACM New York, NY, v. 14, n. 4, p. 1–30, 2021.

BASTOS, I. L. O. Reconhecimento de sinais da libras utilizando descritores de forma e redes neurais artificiais. Instituto de Matemática. Departamento de Ciência da Computação, 2016.

BRASIL. Lei nº 10.436, de 24 de abril de 2002.: Dispõe sobre a língua brasileira de sinais - libras e dá outras providências. *Diário Oficial [da] República Federativa do Brasil*, Brasília, DF, 2002. Disponível em: ⟨https://www.planalto.gov.br/ccivil_03/leis/2002/l10436.htm⟩.

CRISTIANO, A. *O que É libras?* 2017. Disponível em: ⟨https://www.libras.com.br/o-que-e-libras⟩.

CRISTIANO, A. *Os Cinco Parâmetros da Libras*. 2018. Disponível em: ⟨https://www.libras.com.br/os-cinco-parametros-da-libras⟩.

DAROYA, R.; PERALTA, D.; NAVAL, P. Alphabet sign language image classification using deep learning. In: IEEE. *TENCON 2018-2018 IEEE Region 10 Conference*. [S.l.], 2018. p. 0646–0650.

DENG, J. et al. Imagenet: A large-scale hierarchical image database. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.]: IEEE, 2009. p. 248–255.

EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. *International journal of computer vision*, Springer, v. 88, n. 2, p. 303–338, 2010.

GONZALEZ, R. C.; WOODS, R. E. *Digital image processing*. [S.l.]: Pearson, 2018. ⟨https://dl.icdst.org/pdfs/files4/01c56e081202b62bd7d3b4f8545775fb.pdf⟩.

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

IBGE. *PNS 2019: país tem 17,3 milhões de pessoas com algum tipo de deficiência*. Instituto Brasileiro de Geografia e Estatística - IBGE, 2021. 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#:~:text=2%2C3%20milh%C3%B5es%20de%20pessoas%20tinham%20defici%C3%AAncia%20auditiva&text=Por%20volta%20de%202%2C9,superior%20conclu%C3%ADdo%20tinham%20essa%20condi%C3%A7%C3%A3o.⟩

KAUR, P. et al. Facial-recognition algorithms: A literature review. *Medicine, Science and the Law*, SAGE Publications Sage UK: London, England, v. 60, n. 2, p. 131–139, 2020.

KOROISHI, G. O.; SILVA, B. V. L. Reconhecimento de sinais da libras por visão computacional. *Mecatrone*, v. 1, n. 1, 2015.

LI, S.; DENG, W. Deep facial expression recognition: A survey. *IEEE transactions on affective computing*, IEEE, v. 13, n. 3, p. 1195–1215, 2020.

LI, T.; YAN, Y.; DU, W. Sign language recognition based on computer vision. In: IEEE. *2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*. [S.l.], 2022. p. 927–931.

LIN, T.-Y. et al. Microsoft coco: Common objects in context. *European conference on computer vision*, Springer, p. 740–755, 2014.

MARTINS, L. M. *Língua Brasileira de Sinais (Libras) – Módulo básico II*. Governador do Distrito Federal, 2020. Disponível em: ⟨https://egov.df.gov.br/wp-content/uploads/2023/03/Apostila-8.pdf⟩.

PASZKE, A. et al. Pytorch: An imperative style, high-performance deep learning library. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2019. p. 8024–8035.

REDMON, J. et al. You only look once: Unified, real-time object detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 779–788.

REZENDE, T. M. Aplicação de técnicas de inteligência computacional para análise da expressão facial em reconhecimento de sinais de libras. Universidade Federal de Minas Gerais, 2016.

SILVA, D. R. B. d. et al. Uma arquitetura multifluxo baseada em aprendizagem profunda para reconhecimento de sinais em libras no contexto de saúde. Universidade Federal da Paraíba, 2020.

SILVA, E. *Alfabeto em libras Computer Vision Project*. 2023. Acesso em: 16 de ago. 2024. Disponível em: ⟨https://universe.roboflow.com/elainesilva/alfabeto-em-libras-qrvnw⟩.

SOUZA, T. A. F. de; MONTEIRO, M. S. *Libras em Contexto: Curso Básico : Livro do Professor*. 6. ed. Brasília: Ministério da Educação, Secretaria de Educação Especial, 2006. 448 p. Ilustrado. Disponível em: ⟨https://repositorio.faculdadefama.edu.br/xmlui/handle/123456789/13⟩.

SZELISKI, R. *Computer Vision: Algorithms and Applications*. 2nd. ed. [S.l.]: Springer, 2022. ISBN 9783030343729.

TASKIRAN, M.; KILLIOGLU, M.; KAHRAMAN, N. A real-time system for recognition of american sign language by using deep learning. In: IEEE. *2018 41st international conference on telecommunications and signal processing (TSP)*. [S.l.], 2018. p. 1–5.

TURK, M.; ATHITSOS, V. Gesture recognition. *Computer vision: a reference guide*, Springer, p. 1–6, 2020.

ULTRALYTICS. *Augmentation Settings and Hyperparameters*. 2024. Accessed: 2024-08-16. Disponível em: ⟨https://docs.ultralytics.com/modes/train/#augmentation-settings-and-hyperparameters⟩.