

RECONHECIMENTO DE SINAIS DINÂMICOS DE LÍNGUA DE SINAIS

Halliday Gauss - 18.1.4093

Thiago Borba - 18.1.4015

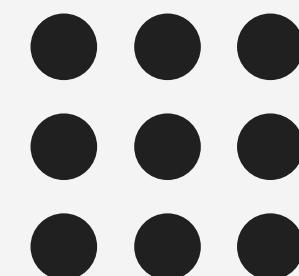
Lucas Souza - 18.1.4083



Introdução

No Brasil, a Linguagem Brasileira de Sinais (Libras) é de suma importância para a comunicação e interações sociais

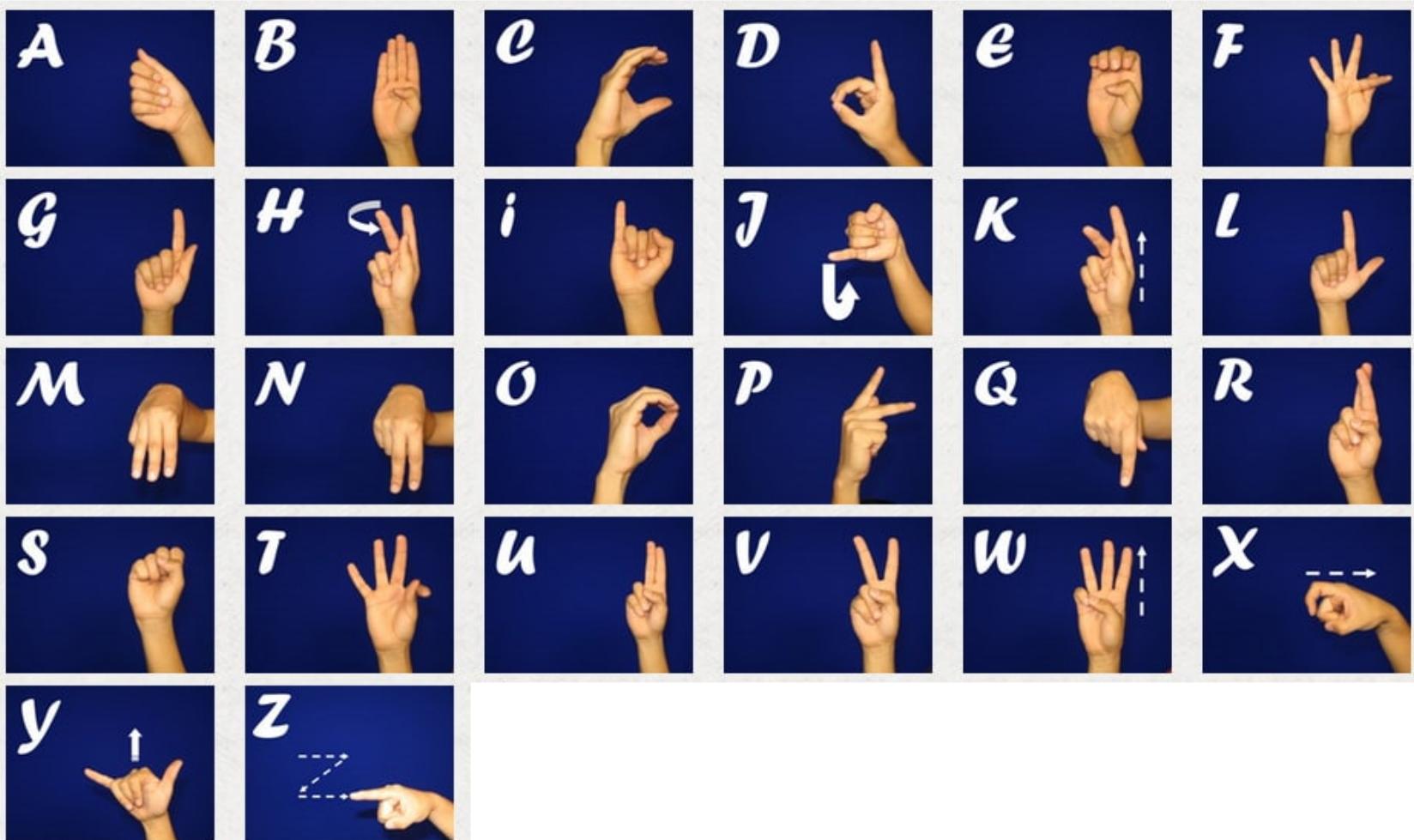
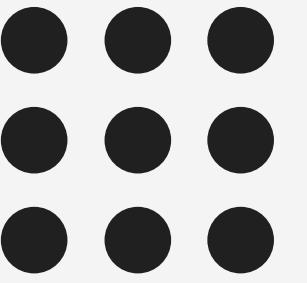
Dado a dificuldade enfrentada pelas pessoas que necessitam dessas línguas para se comunicar, aparelhos tecnológicos são relevantes para ajudá-las a se integrar na sociedade e produzir igualdade e inclusão, como por exemplo na saúde e educação.



O Protótipo

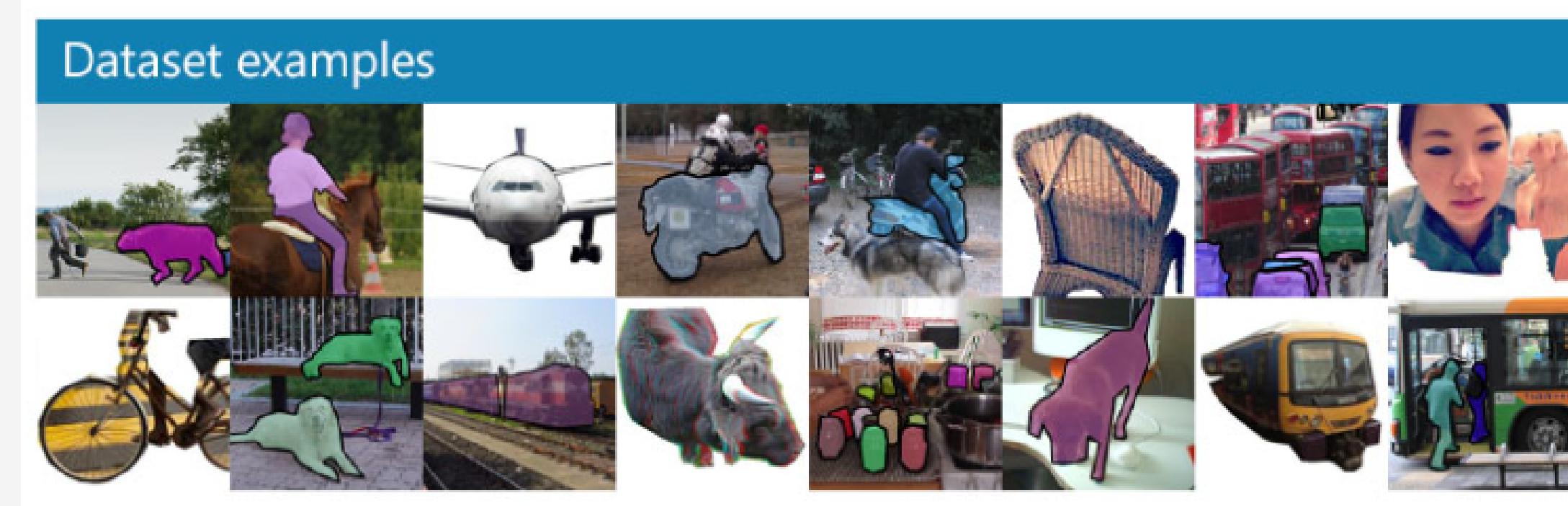
Foi construído um protótipo baseado na ideia do artigo de Simon, T., Joo, H., Matthews, I., & Sheikh, Y. (2017). Hand keypoint detection in single images using multiview bootstrapping. , 1145–1153.

Esse protótipo é capaz de realizar a detecção de sinais de LIBRAS através de pontos chaves da mão.



Modelo COCO CAFFE

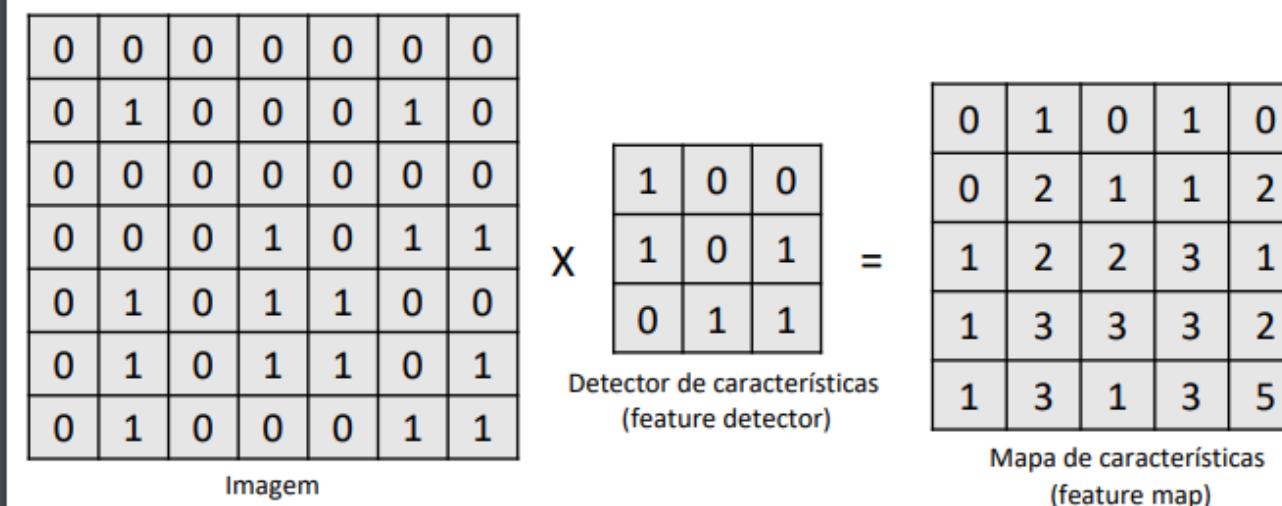
Para atingir tal objetivo foi utilizado um modelo já treinado chamado COCO CAFFE o qual utiliza Redes Neurais Convolucionais e possui um conjunto de treinamento, validação e teste da COCO (Common Objects in Context), contendo mais de 200.000 imagens e 250.000 pessoas nomeadas com pontos chaves, sendo a maioria dessas imagens em escalas médias e grandes. Esse modelo COCO é treinado pelo CAFFE Deep Learning Framework.

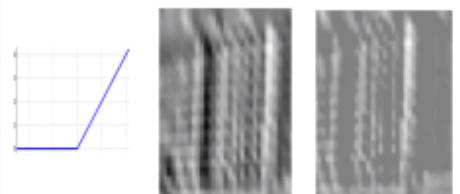


Características do modelo COCO CAFFE

Na primeira etapa é aplicado, pela Rede Neural, a convolução nas imagens através de um detector de características, o que vai resultar numa imagem de menor tamanho que é chamada de Mapa de Características. Em seguida é utilizada a função Relu ou função de ativação que vai atribuir 0 somente para os valores menores que 0.

Etapa 1 – Operador de convolução (Relu)

$$\begin{matrix} & \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{matrix} & \times & \begin{matrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{matrix} & = & \begin{matrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 1 & 2 \\ 1 & 2 & 2 & 3 & 1 \\ 1 & 3 & 3 & 3 & 2 \\ 1 & 3 & 1 & 3 & 5 \end{matrix} \\ \text{Imagem} & & \text{Detector de características (feature detector)} & & \text{Mapa de características (feature map)} \end{matrix}$$




Características do modelo COCO CAFFE

Etapa 2 – Pooling

Após esse procedimento é feito um Max Pooling que retornará uma imagem menor, focada nos maiores valores que são as características mais importante do Mapa de Características.

0	1	0	1	0
0	2	1	1	2
1	2	2	3	1
1	3	3	3	2
1	3	1	3	5



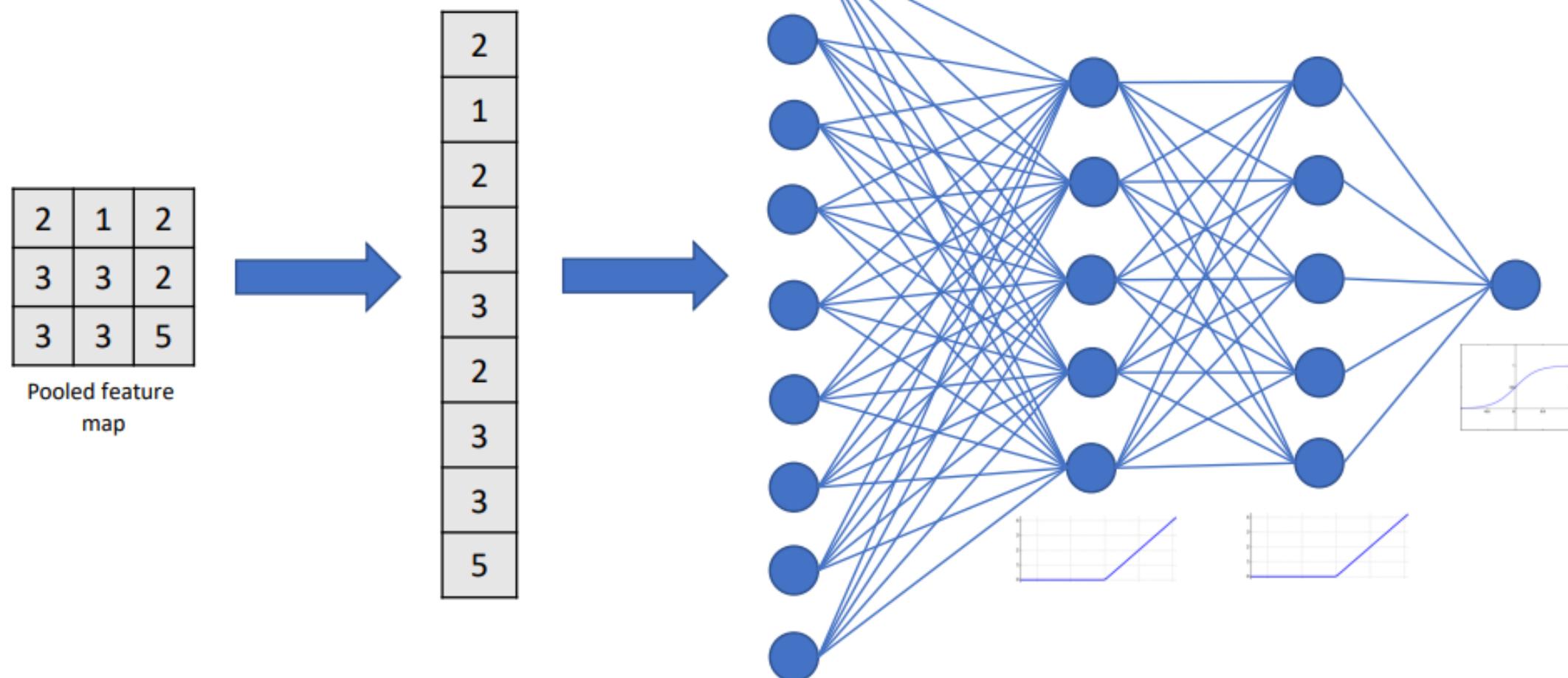
2	1	2
3	3	2
3	3	5

Mapa de características
(feature map)

Características do modelo COCO CAFFE

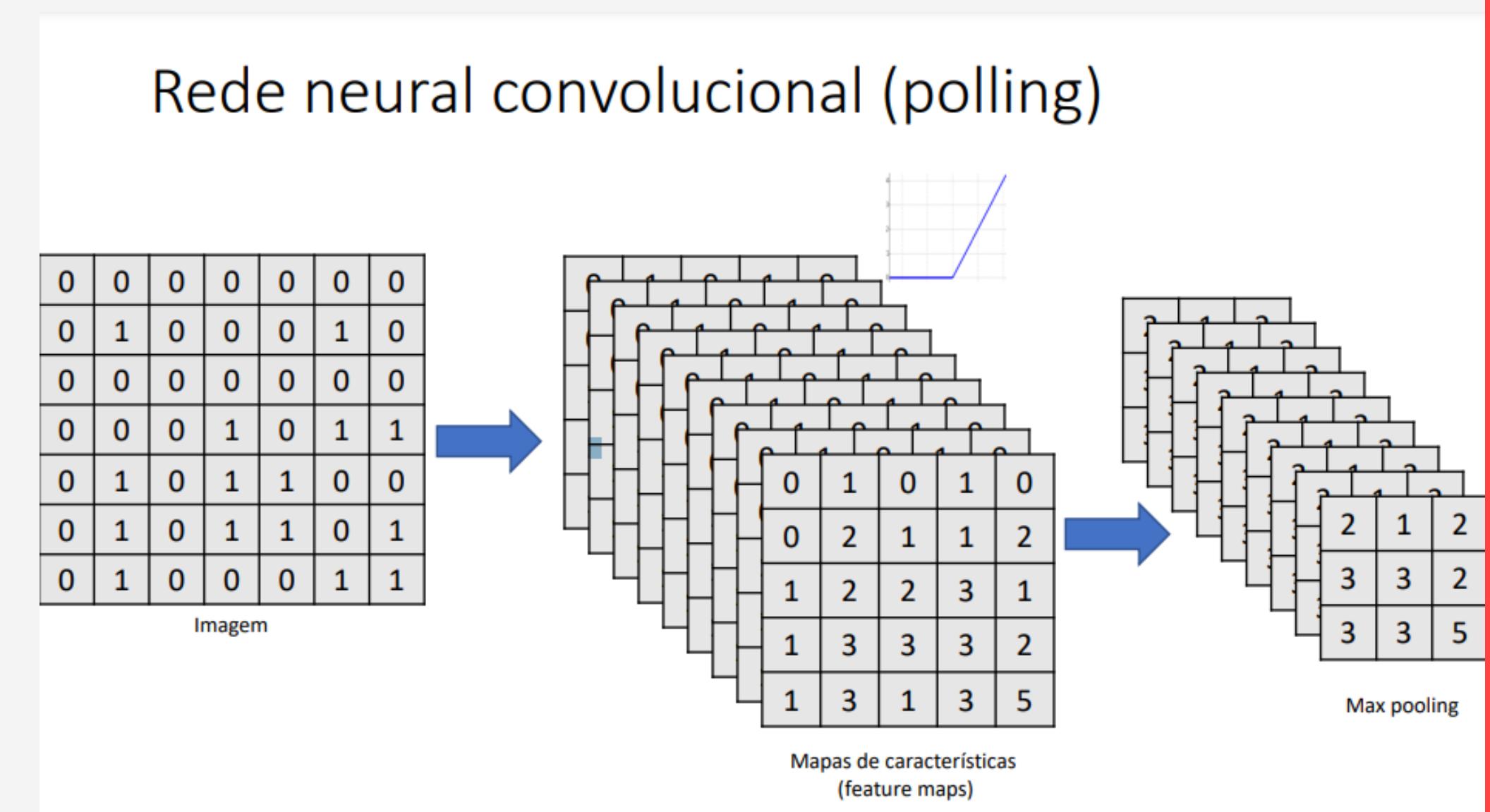
Em seguida é feito um Flattening, ou seja, uma conversão do resultado do Pooling para um vetor que é submetido para uma Rede Neural Densa Tradicional, que depois usará algum algoritmo de classificação, Soft Max por exemplo, para melhor classificação de um ponto chave.

Etapa 3 – Flattening



Características do modelo COCO CAFFE

Cabe ressaltar que é possível, a partir de uma imagem, gerar vários Mapas de Características



Características do modelo COCO CAFFE

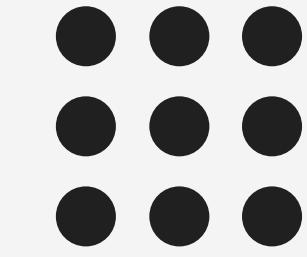
O modelo COCO CAFE tem como base a Arquitetura VGGNET que é uma rede Neural Convolucional, e esse modelo precisa de uma imagem de entrada, a sua largura e comprimento, e então a saída será a marcação em 2D do pontos chaves do objeto. E para cada ponto chave será retornado uma matriz de pontos candidatos.

Características do modelo COCO CAFFE

Nesse procedimento também são gerados:

- Os Mapas de Confiança verificam a probabilidade de cada ponto candidato estar na posição correta. Por exemplo, na ponta do polegar têm-se vários pontos candidatos e deve ser escolhido o ponto que tem a maior confiança (probabilidade), para que então esse ponto candidato se torne um ponto chave.
- Os Mapas de Afinidade indicam se um ponto pode ser associado com outro. Por exemplo: o ponto do topo do polegar e o ponto um pouco abaixo do mesmo devem estar associados em uma linha reta, ou seja, estão bem próximos entre si.

Características do Protótipo

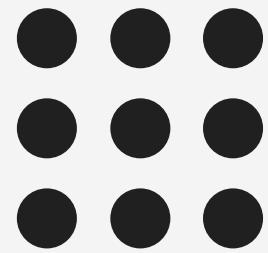


O protótipo baseia-se no uso do modelo COCO CAFFE para o reconhecimento dos pontos chaves da mão, o que irá permitir reconhecer sinais em Libras. Alguns detalhes sobre o modelo utilizado:

- Trata-se a mão inteira como um único objeto.
- Modelo treinado sobre um pequeno conjunto de imagens de mão rotuladas e usam uma Rede Neural para obter estimativas aproximadas dos pontos chave da mão.
- Os autores deste modelo possuíam um enorme sistema multi-view configurado para capturar imagens de diferentes pontos de vista ou ângulos, compostos por 31 câmeras HD
- Os autores deste modelo usaram detectores de ponto chave e imagens de vários ângulos diferentes para criar um detector aprimorado.

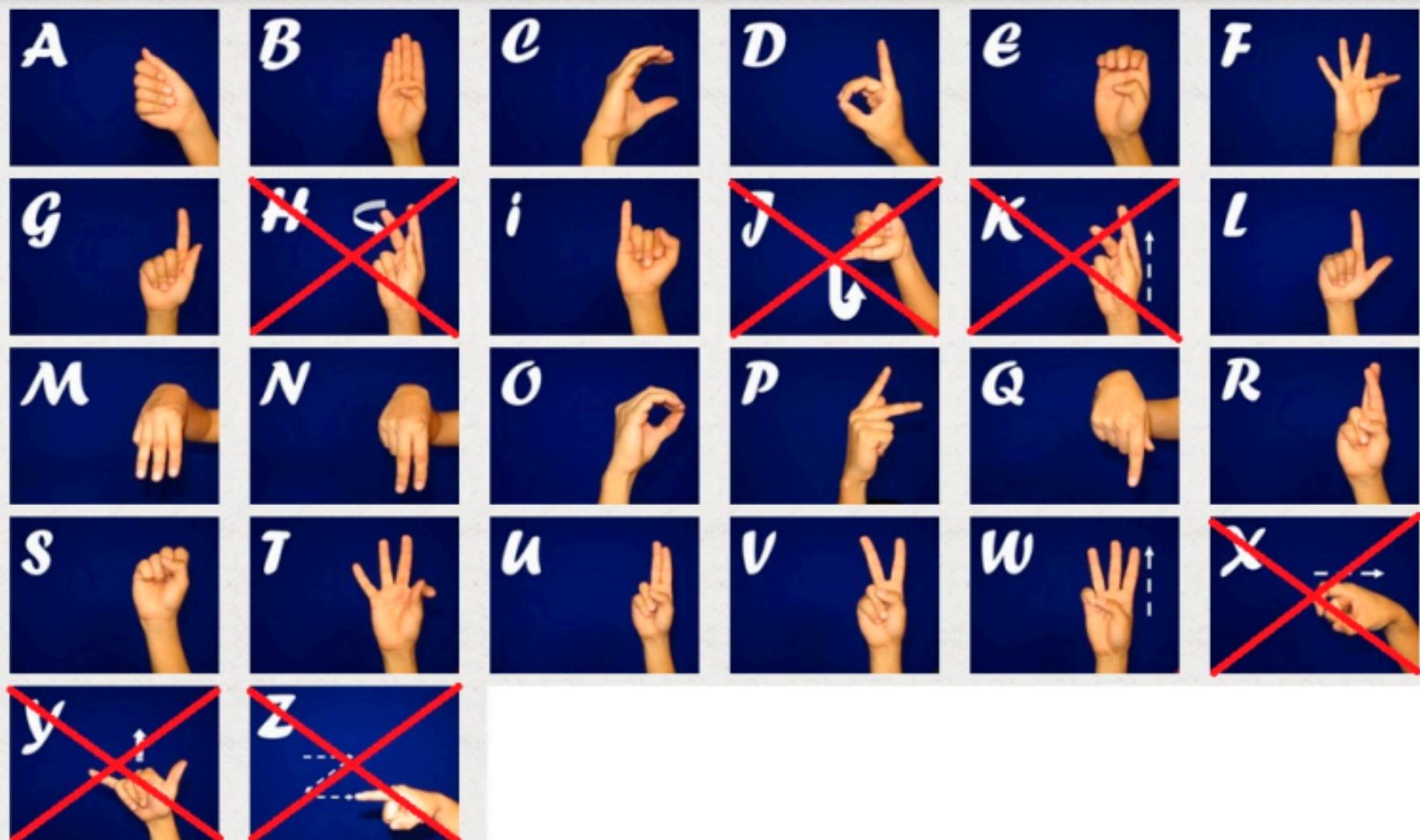
Características do Protótipo

- Este modelo possui 22 pontos chave. A mão tem 21 pontos, enquanto o ponto 22 é representado pelo fundo da imagem.
- A saída possui 22 matrizes, sendo cada matriz o mapa de probabilidade de um ponto chave.

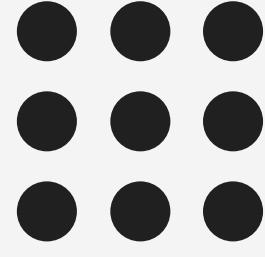


Funcionamento do Protótipo

Utilizando o modelo COCO CAFFE o protótipo implementado realiza o reconhecimento, em vídeos, de todos os símbolos em LIBRAS que representam o alfabeto, com exceção das letras H, J, K, X, Y e Z.

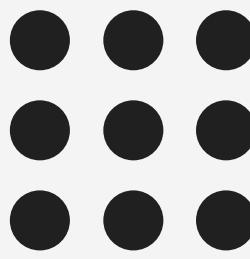
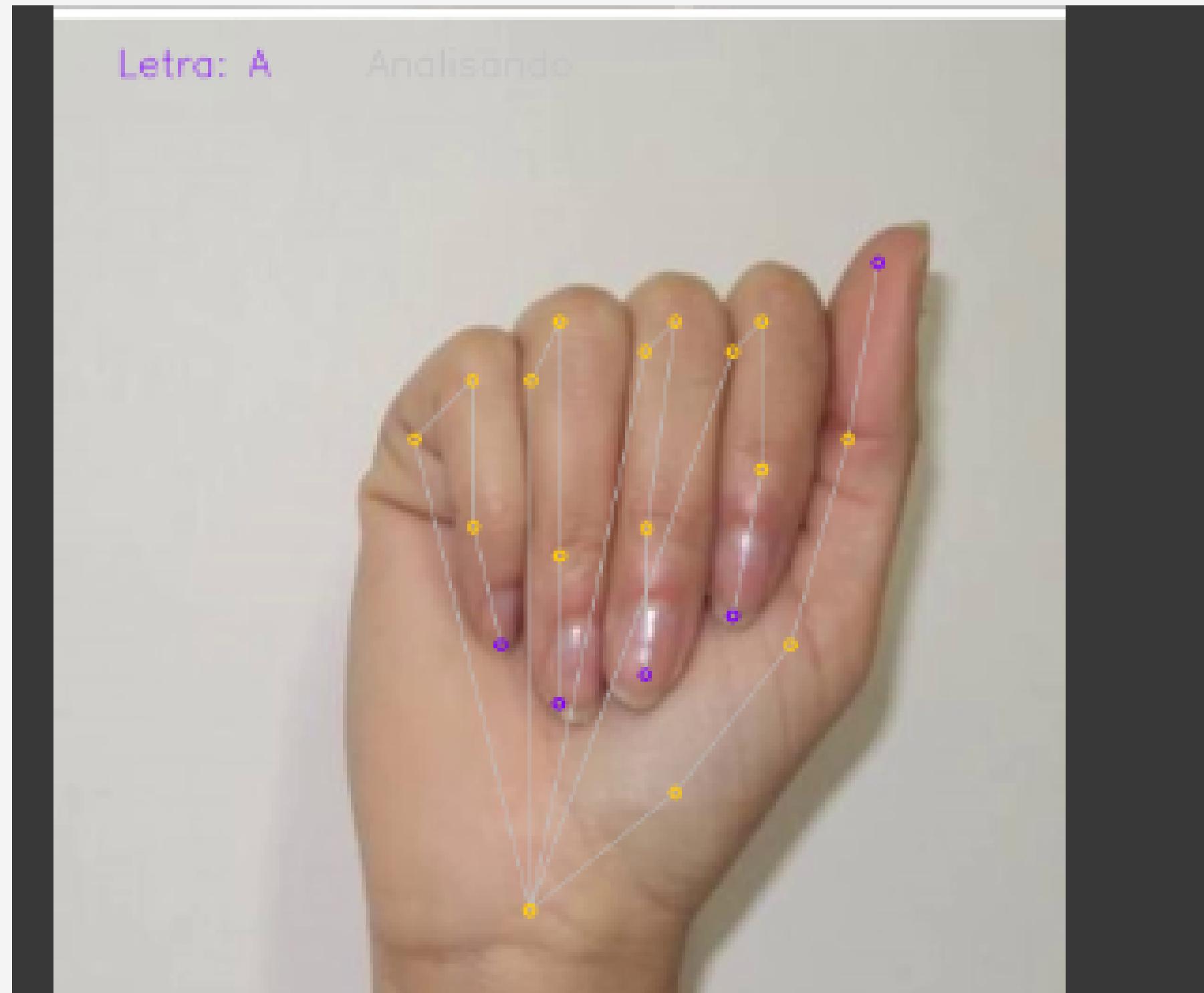


Implementação do Protótipo

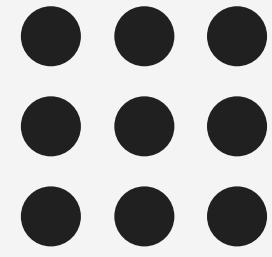


O protótipo utiliza o Google Colab, Python e OpenCV, para realizar esse reconhecimento. Alguns módulos implementados e vídeos contendo os sinais de libras são acessados pelo protótipo através do Google Drive, e então a partir de cada frame do vídeo o algoritmo tenta fazer o reconhecimento com no modelo, e então é gerado como saída um vídeo contendo o rótulo, que é a letra representada pelo sinal, de cada sinal em cada frame analisado. Nesse vídeo também é mostrado os pontos chaves detectados nas mãos em cada frame e a ligação entre eles(esqueleto).

Implementação do Protótipo



Implementação do Protótipo

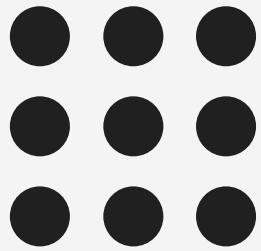


Para a identificação mais dinâmica de várias posições, foram desenvolvidos 4 módulos para extrair algumas características e comparar o deslocamento das articulações (pontos chave). Foi extraído de cada frame a altura, posição e proximidade entre os pontos chaves detectados:

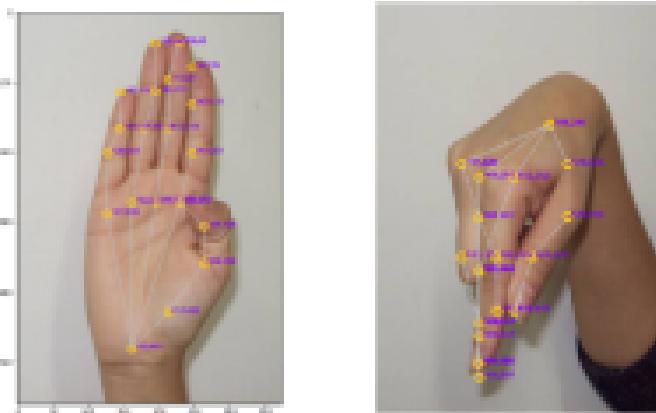
- Altura: verifica se um ponto está acima ou abaixo de outro ponto específico. Por exemplo, se a ponta do dedo está acima do punho.
- Posição: verifica se os dedos estão dobrados ou esticados, na posição horizontal ou na vertical. Também recebe o resultado da altura para saber em que posição a mão está (voltada acima ou abaixo).
- Proximidade: comparar a proximidade entre os pontos chaves detectados. Por exemplo, se o resultado da Posição for igual a 'dobrado' para o dedo indicador e dedo médio e ambos estiverem na mesma altura, então significa que os dedos estão próximos.

Implementação do Protótipo

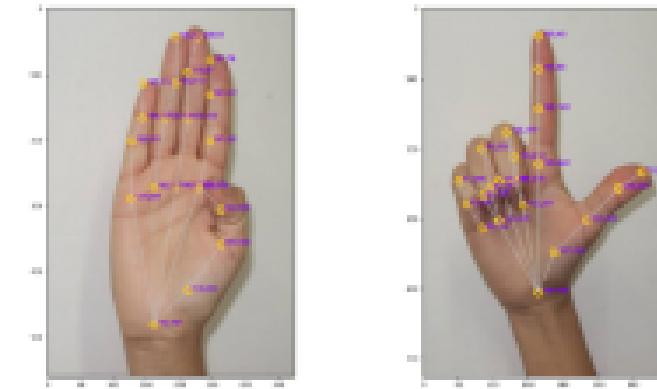
Após extrair todas estas características, é criado o alfabeto de características, onde uma matriz de características recebe o resultado de todos os módulos extratores. Então tudo isso é usado para comparar com uma nova análise (nova imagem) de entrada.



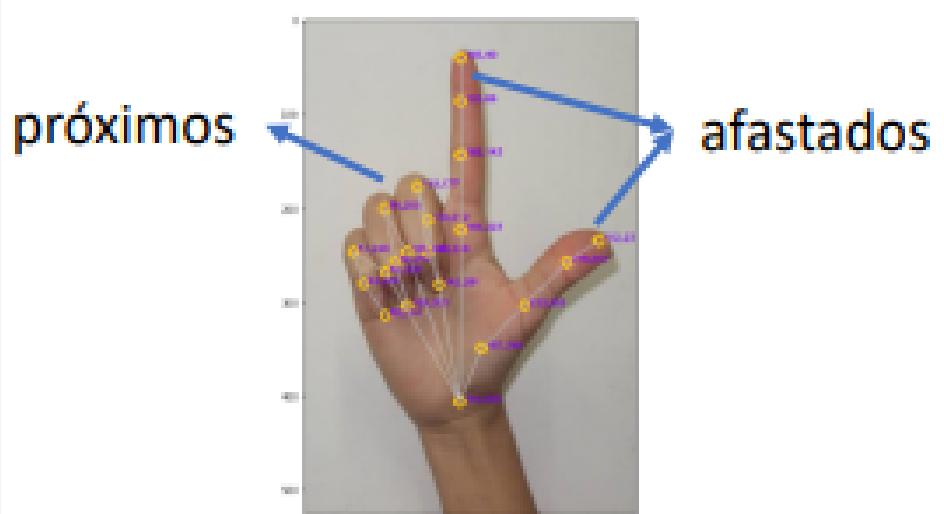
Mão acima ou abaixo



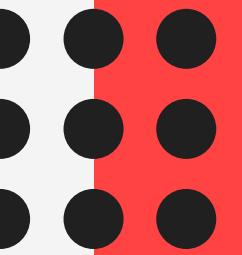
Dedos esticados ou dobrados



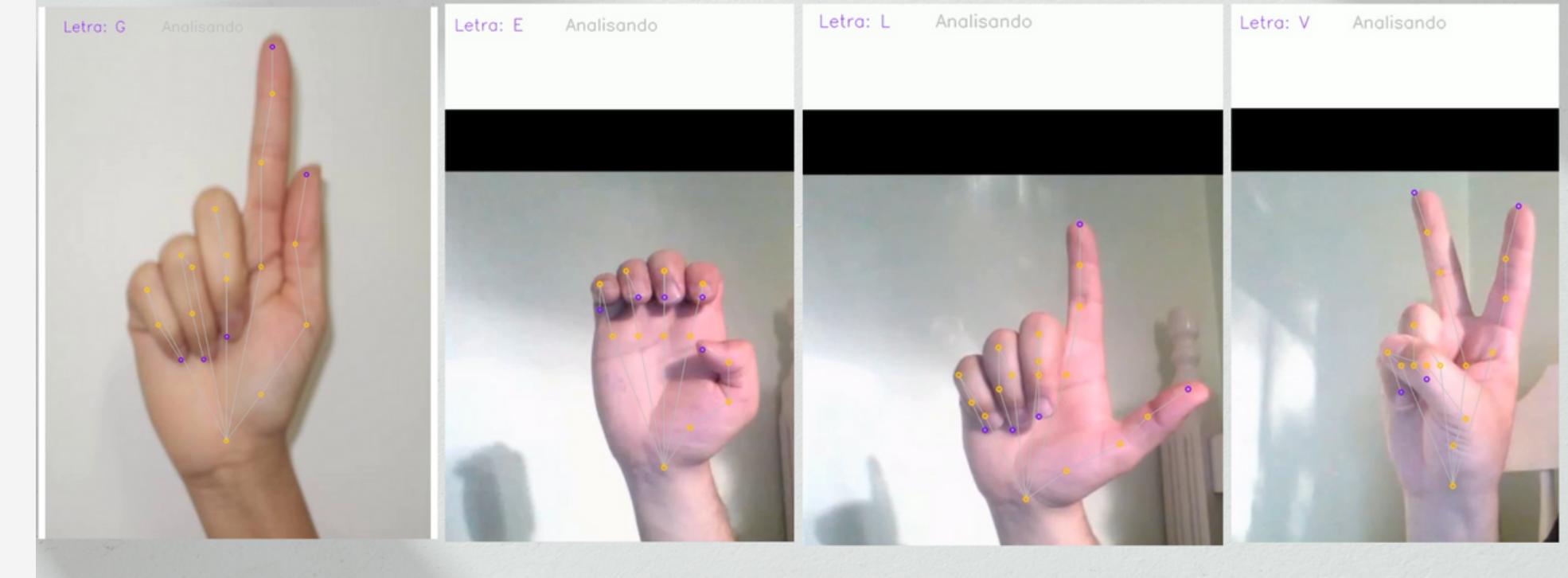
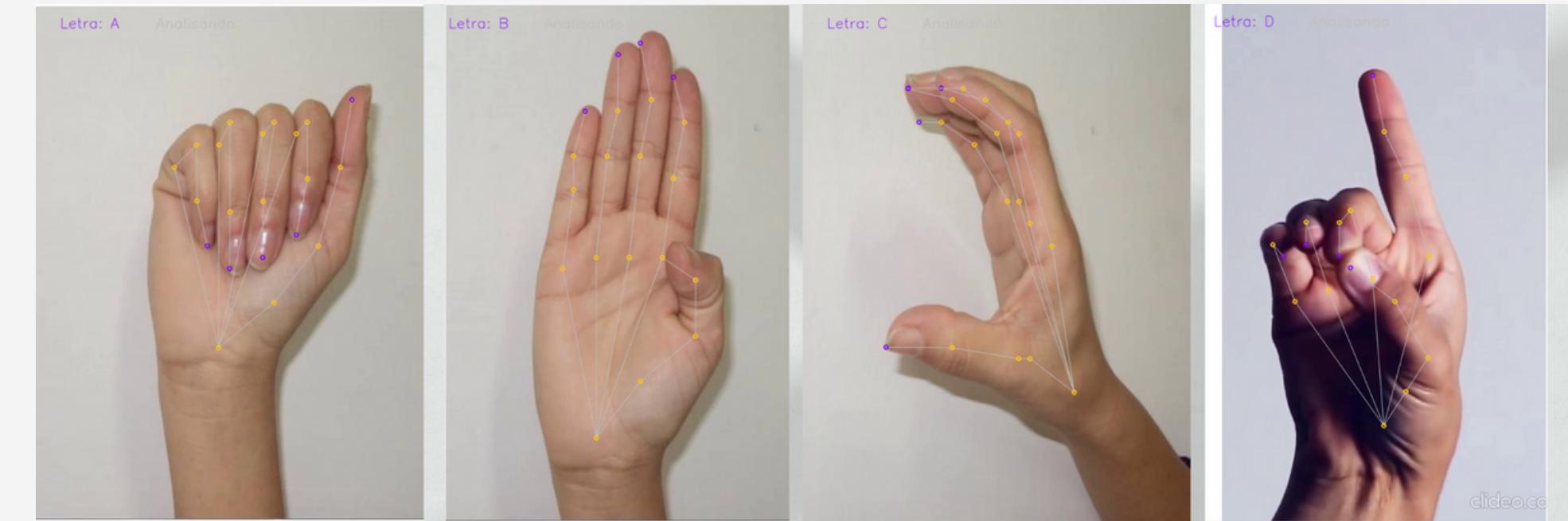
Dedos

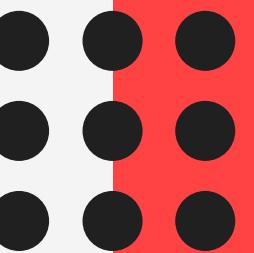


Resultados



O Protótipo apresentou bons resultados em reconhecer os sinais do alfabeto de Libras em vídeos.





Limitações

Apesar do bom funcionamento do protótipo, o mesmo possui algumas limitações.

Devido ao movimento adicional para a execução correta das letras H, J, K, X, Y, Z elas devem ser analisadas em uma função diferente, onde comparamos a transição entre os pontos, e esse protótipo não garante o reconhecimento das mesmas.

Uma outra dificuldade é a análise da letra T, para o dedo polegar, devido a estar sobreposto pelo dedo indicador, o algoritmo não reconhece os pontos da ponta dos dedos.

A letra N e U podem ser confundidas no reconhecimento, pois são parecidas e mudam somente a direção.



Duvidas?

