Insper

Relatório do Projeto de Machine Learning

Libras - Convolutional neural network

Eiki Luis Yamashiro
Fernando Fincatti
Lais Nascimento da Silva

Professor Fábio Ayres

São Paulo
Novembro/2021

1. Introdução

A Língua Brasileira de Sinais (Libras) é uma língua de modalidade gestual-visual onde é possível se comunicar através de gestos. É utilizada principalmente por deficientes auditivos, sendo assim uma importante ferramenta de inclusão e legalmente reconhecida como meio de comunicação e expressão desde 24 de Abril de 2002, através da Lei nº 10.436.

Dessa forma, tem-se que Libras é um assunto de extrema relevância, pois a linguagem é parte integrante no nosso desenvolvimento como seres humanos, é através da comunicação que a gente pode compartilhar ideias, sentimentos, emoções e mensagens. Além disso, tem-se que cerca de 360 milhões de pessoas sofrem com algum tipo de deficiência no mundo, uma pesquisa realizada pelo IBGE apontou que mais de 9 milhões de pessoas possuem deficiência auditiva, isso corresponde a mais de 5% da população do Brasil. Cerca de 20% dos deficientes auditivos idosos não conseguem sair sozinhos, apenas 37% está no mercado e 87% não usa aparelhos auditivos devido ao preço. Portanto, promover comunicação e inclusão para essa grande quantidade de pessoas é preciso.

Este projeto de Machine Learning se baseia nessa necessidade e tem como meta promover acessibilidade, pois a falta de acessibilidade e acolhimento limita o acesso dos surdos às oportunidades básicas. Para isso, usou-se a técnica de Convolutional neural network (CNN), para criar uma rede neural que identificasse alguns gestos em libras, sendo estes letras do alfabeto. Com este aprendizado de classificação a partir de imagens é possível criar mecanismos para facilitar a comunicação entre surdos e pessoas que não sabem libras, de forma autônoma, quebrando as barreiras de comunicação.

Neste relatório será apresentada a base de dados usada para este projeto. Além disso, serão descritos os mecanismos usados para a criação da CNN, quais estratégias foram adotadas para gerar o aprendizado correto da rede neural. E os resultados serão expostos, mostrando o quão assertiva a rede neural ficou.

2. Base de Dados

A base de dados utilizada é composta por um conjunto de imagens divididas em letras do alfabeto, foi retirada do github(1). Contém todas as letras do alfabeto, exceto as letras *h, j, k x* e *z*, ou seja, são 21 símbolos e portanto 21 classes diferentes para a rede neural aprender. No total são 46262 imagens, originalmente elas já estavam divididas em treinamento e teste, porém os dois grupos foram agregados e separados posteriormente, em conjunto de treino e conjunto de teste, pela função train_test_split da biblioteca scikit-learn do python, dessa maneira o valores de X, tanto para treinamento quanto para teste já saem em formato de tensor e isso será de extrema importância para próximos passos do projeto.

Ademais, tem-se que a separação gerou um grupo de treinamento com 90% das imagens, ou seja, 4165 imagens são usadas para treinamento e os 10% restantes

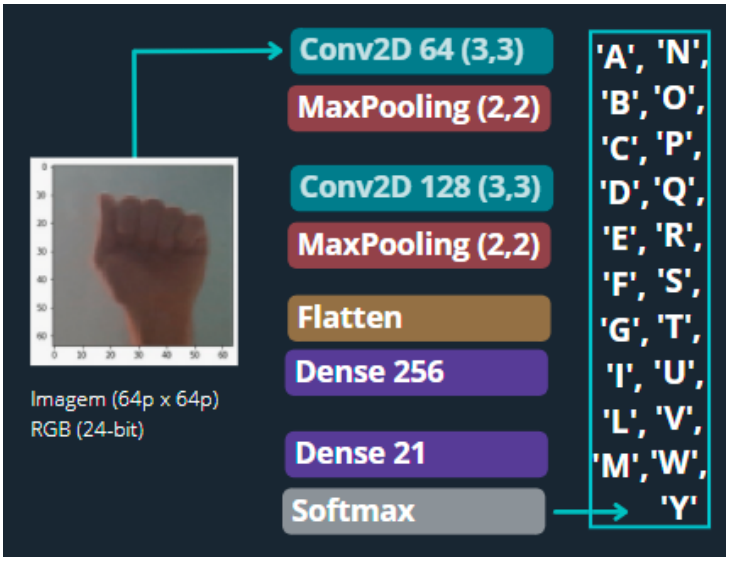
foram usados para o teste, sendo 4627 imagens. As imagens têm 64x64 píxes e estão em RGB. Com os grupos devidamente separados, eles foram salvos no formato pickle, para que não fosse necessário ler o dataset, que é extenso, inteiro todas as vezes.
O dataset é extenso, porém tem suas limitações. Como já citado anteriormente, estão faltando as letras h, j, k, x e z, pois os sinais dessas letras necessitam de movimento, portanto apenas com a imagem o aprendizado não ficaria correto. Outra limitação que pode afetar os resultados é o fato de que o fundo das imagens é sempre branco e o dataset possui apenas mãos brancas.

3. Modelo

Para criar o modelo pensou-se em uma rede neural convolucional, pois ela é uma classe de rede neural artificial do tipo feed-forward, que vem sendo aplicada com sucesso no processamento e análise de imagens digitais. Ou seja, justamente o que o projeto precisa para conseguir classificar os gestos em letras.

3.1) Primeiro modelo

Foi usado o modelo *sequential* do TensorFlow. Com ele foi possível montar uma sequência de camadas, sendo que cada camada tem um tensor de entrada e um tensor de saída. A camada convolucional, foi desenvolvida com base na operação de convolução e possui um alto desempenho para o processamento de imagens. A primeira camada com 64 neurônios e uma janela (3,3) é o suficiente para processar uma imagem 64p x 64p, no caso utilizou-se a Conv2D para aproveitar as informações contidas nos canas RGB de 24-bits, após isso adiciona-se uma camada MaxPooling, seguida por uma convolucional bidimensional de 128 neurônios e uma MaxPooling. A camada Flatten recebe uma informação bidimensional e achata ela para um vetor unidimensional, possibilitando com que os dados possam passar pelas camadas dense ou fully connected. Finalizou-se o modelo com uma dense de 21 neurônios, um para cada letra com uma função de ativação de softmax.



Topologia da LIBRAS - CNN

3.2) Validação

Para analisar se houve overfitting nos resultados do treinamento do primeiro modelo, o conjunto treinamento foi dividido em treinamento e validação, sendo que 20% do conjunto original de treinamento passou a ser usado para validação e os 80% restante para treino. Nesta situação o Batch size é de 64 a usa-se 15 Epochs. A comparação entre o resultado obtido no treinamento e na validação pode ser visto pelo

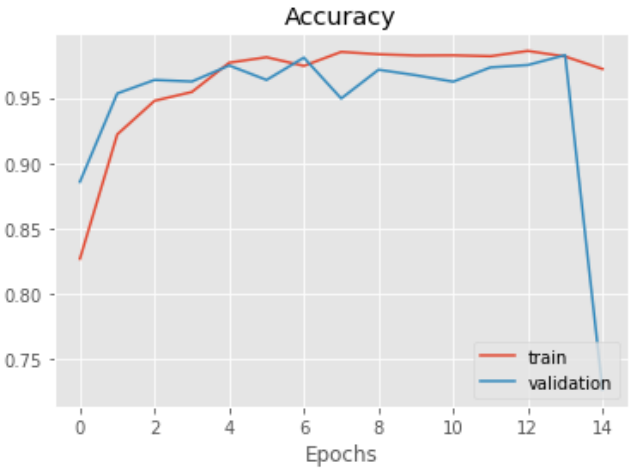


Gráfico 1: Acurácia treino e validação

3.3) Data Augmentation

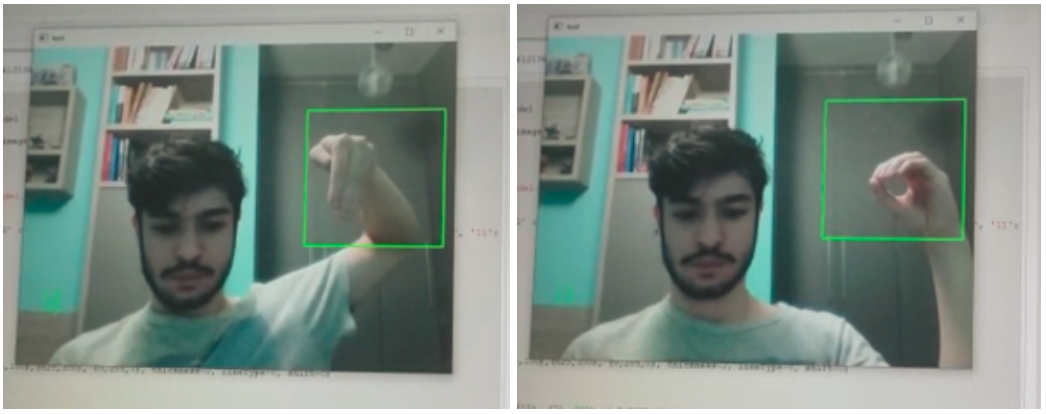
A fim de melhorar os resultados obtidos no treinamento foi pensando a estratégia de data augmentation aumentar a quantidade e a diversidade das imagens do dataset. Para isto foi, analisou-se que tentar aplicar a das imagens para gerar diferentes

imagens iria prejudicar o modelo, visto que ao rotacionar alguns letras podem se transformar em outras, como por exemplo a letra ‘u’ e a letra ‘p’, por serem parecidas podem ser confundidas e assim prejudicar o aprendizado. Outro métodos de data augmentation que foi descartado foi o de flip pelo mesmo motivo, neste caso, o sinal ‘d’ poderia ficar muito parecido com o gesto que representa a letra ‘i’.
Diante disso, usou-se no projeto uma outra solução para o data augmentation, a aplicação de filtros, que possibilita a mudança da cor do background e da mão, alteração no brilho da imagem, e técnicas para borrar a imagem.
Entretanto, ao gerar novas imagens o modelo não foi capaz de convergir, mesmo utilizando alterações mínimas nas imagens, portanto a feature foi retirada do modelo final.

4. Tempo Real

Para tornar o projeto mais aplicável às situações do dia a dia, foi utilizado a estratégia de integrar a webcam, para identificação dos símbolos em tempo real. Dessa forma, seria possível utilizar a ferramenta em um contexto realístico, onde o objetivo é auxiliar na comunicação com pessoas portadores de deficiência.

.



5. Resultado

A acurácia de treinamento ficou acima de 80% e convergiu para 94%, e como pode se observar pelo gráfico a acurácia de validação ficou bem próxima. Dessa forma, é possível concluir que não houve overfitting. Então aplicou-se este modelo ao conjunto de teste e a acurácia foi de 94,03%.

6. Conclusão

Assim, tem-se que ao final do projeto construiu-se um ferramenta com Machine Learning capaz de gerar maior acessibilidade para os deficiente auditivos. Há grandes chances de acertar o símbolo que se está querendo representar, além disso com a integração via webcam pode resolver muitos problemas cotidianos.

Uma possível iteração seria adicionar o software desenvolvido em um componente embarcado, ou mesmo integrar a um aplicativo de celular, de forma a produtificar a solução encontrada.

7. Referências

1. https://github.com/lucaaslb/cnn-libras/tree/master/dataset