Representação e classificação utilizando IA

...

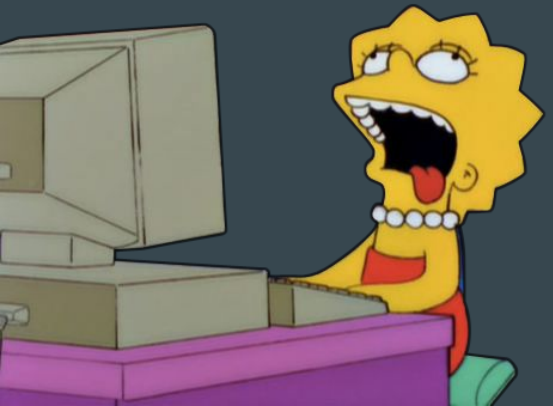
Erick Eckermann Cardoso

Objetivo

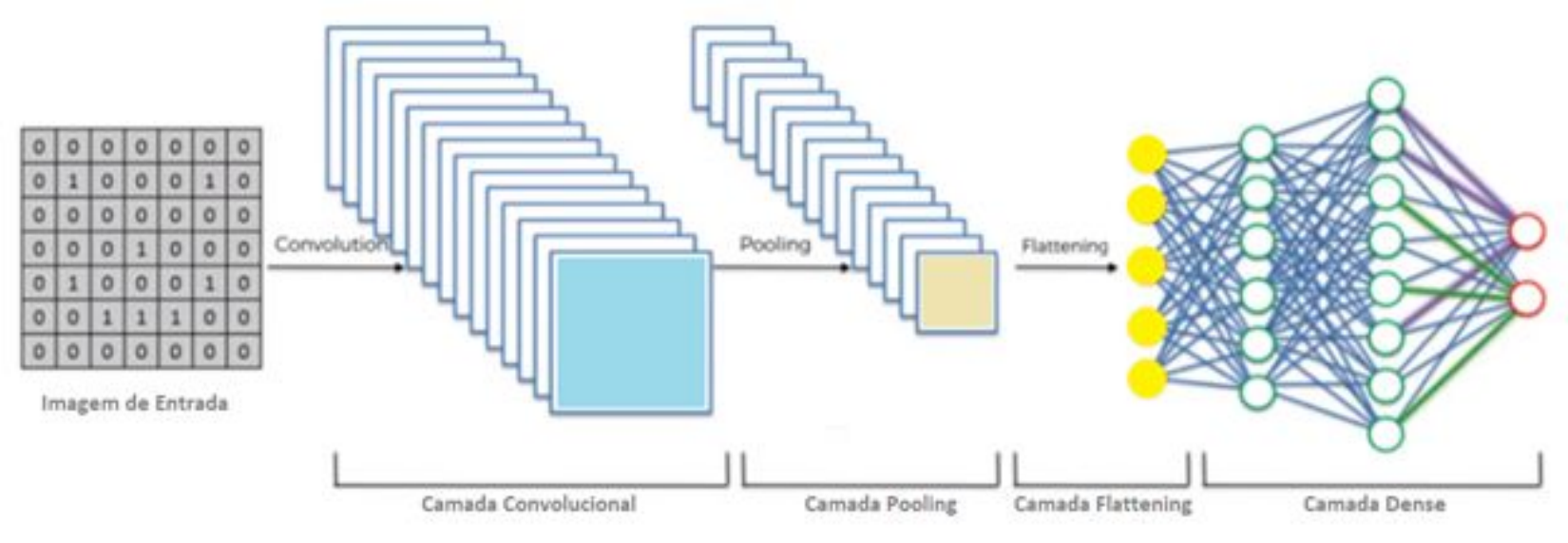
- O objetivo consiste treinar redes neurais convolucionais (CNN) para classificar as imagens de uma base em 6 classes, cada classe consistindo nos personagens dos Simpsons e a última sendo a família completa.
- Para realizar tal tarefa, deveria-se extrair um vetor de características (representação) de cada imagem. As bases de treino e validação são compostas de 253 e 106 imagens, respectivamente.
- Foi utilizado o algoritmo de Knn para a classificação.

Especificação

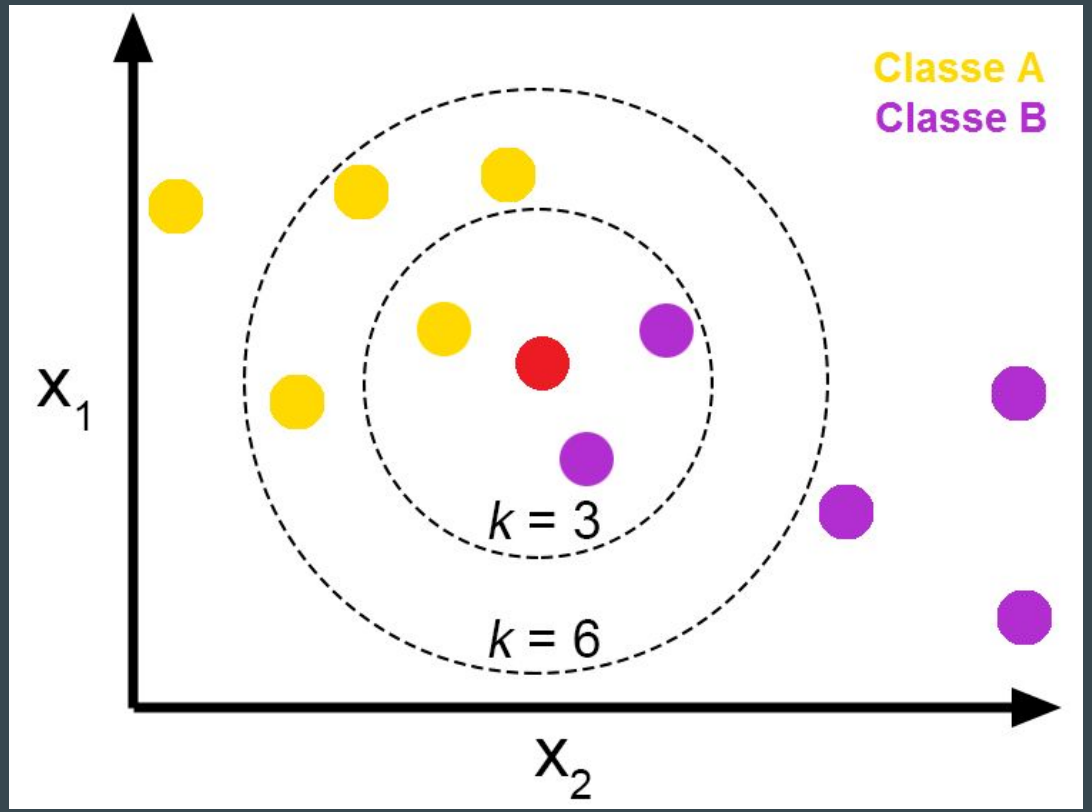
- Trabalho implementado em Python 3
- Utilizando o módulo Tensor Flow e a API Keras para criação, treinamento do modelo pela base, extração do vetor de características e classificação.
- Para a classificação, utilizou-se o algoritmo da classificação com KNN com k = 1.
- Trabalho baseado no artigo de Marcos Tanaka, “Classificação de imagens com deep learning e TensorFlow”.



CNN (Rede neural convolucional)



KNN



Modelos

- Não foi escrita uma CNN do zero, mas sim treinados alguns modelos prontos utilizando um processo chamado Transfer Learning.
- Os testes foram feitos com um total de 3 modelos disponibilizados pela API Keras.
- Estes são: ResNet50, ResNet50V2, EfficientNetB4.
- O **único** onde foi realizado o **treinamento** pela camada densa foi **ResNet50**, onde foram atingidos os melhores resultados.
- Com o objetivo de aumentar a acurácia, foram feitos testes com as imagens das bases normais e cortadas, diminuindo seu tamanho em 20%.

Resultados sem corte e sem treinamento

ResNet50

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.5

ResNet50V2

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.44339622641509435

EfficientNetB4

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.5566037735849056

Resultados com cortes e sem treinamentos

ResNet50

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.6226415094339622

ResNet50V2

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.45283018867924533

EfficientNetB4

- Numéro de Bach: 32
- Paciência: 15
- Accuracy: 0.5566037735849056

Rede treinada (ResNet50)

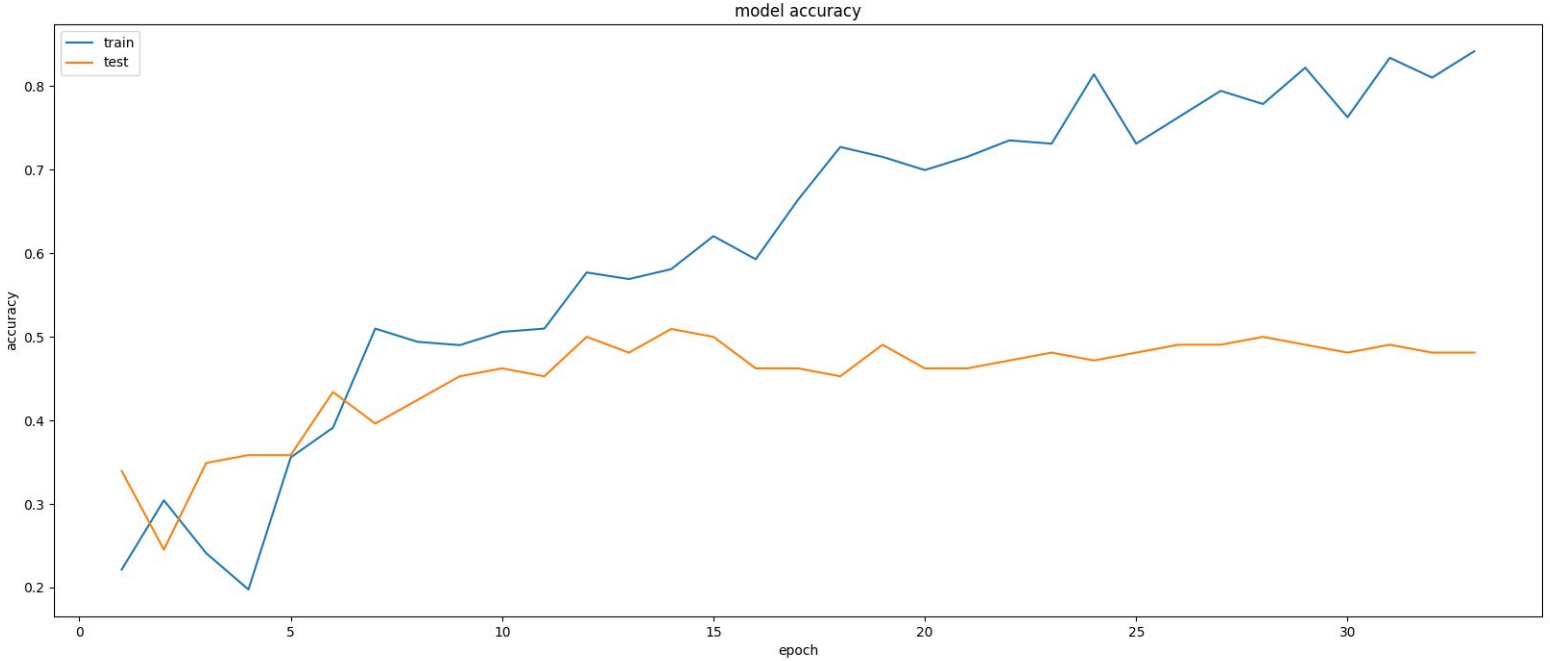
- O treinamento foi realizado em camada densa, alterando os parâmetros para se buscar os melhores resultados.
- Para o primeiro treinamento as camadas convolucionais não são treinadas.
- Método de treinamento foi o transfer learning, o qual é utilizada uma CNN já treinada e adicionadou ma camada a mais, treinando apenas esta camada para nosso objetivo

Resultados

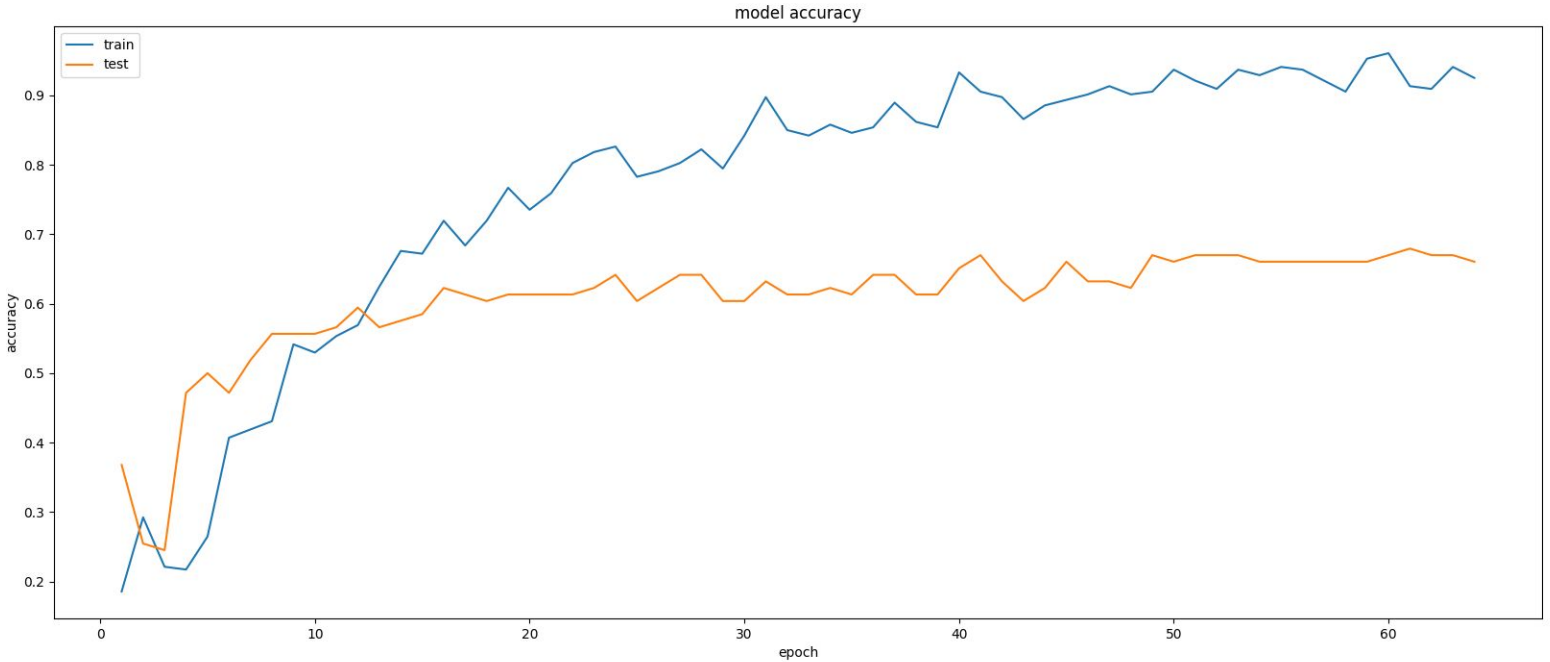
- Base sem imagens cortadas
- Paciência: 15
- Número de Bach: 32
- Accuracy: 0.5377358490566038

- Base com imagens cortadas
- Paciência: 15
- Número de Bach: 32
- Accuracy: 0.7264150943396226

Acurácia para base não modificada



Acurácia base modificada



Conclusão

Em geral, os resultados com as bases onde as imagens tiveram seus tamanhos cortados em 20%, os resultados foram melhores e mantendo na média dos 60% de acurácia.

Com a rede treinada utilizando a camada Densa, o maior resultado atingido foi em média 72%, o que é significativamente melhor do que os outros resultados. Percebe-se também que, no geral, aumentar a paciência interfere no resultado positivamente, porém o custo computacional também aumenta dado ao aumento de iterações. Modificar o bach_size em geral não trouxe mudanças significativas.

Para uma rede apresentar resultados melhores em geral, deve-se haver uma base de treino maior e de maior qualidade, que permita extrair a maior quantidade possível de informações de representação. Assim, pode-se concluir que os resultados com as imagens cortadas tiveram melhor eficiência pois cortar a imagem permitia uma melhor representação, tornando a base mais bem preparada.

Referências

1. [MARCOS TANAKA]:https://imasters.com.br/back-end/classificacao-de-imagens-com-deep-learning-e-tensorflow
2. Documentação oficial tensorflow: https://www.tensorflow.org/?hl=pt-br
3. Documentação Api Keras: https://keras.io/api/

Obrigado