Luiz Felipe Nagatani Rabello

Relatório LAMIA - Módulo 20

Seção 1

A seção 1 consiste em uma breve introdução ao curso, apresentando os materiais que serão utilizados no decorrer e os conteúdos que serão abordados.

Seção 2

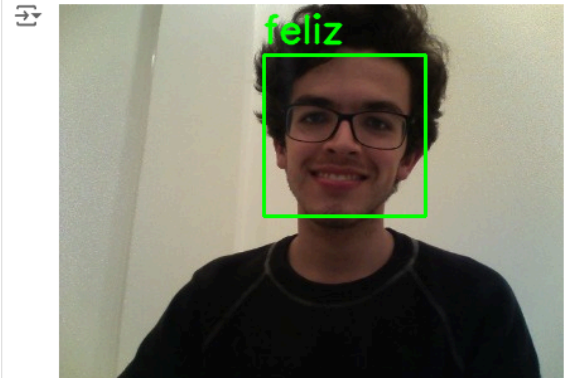
É na seção 2 que se inicia a prática, começando com uma introdução ao conceito de reconhecimento de imagens na tecnologia, mostrando como funciona na teoria e suas aplicações

Como primeiro bloco de código é mostrado como importar as bibliotecas opencv e o tensorflow, juntamente com o processo de instalar e atualizar as bibliotecas.

Esta seção provou-se útil ao mostrar ferramentas do colab, como integração com o google drive e como importar imagens do mesmo, conteúdo não abordado em outros cards anteriores de forma aprofundada.

É apresentado um exemplo de como reconhecer emoções numa foto utilizando o cascadeclassifier com o “haar cascade frontalface” exigindo tratamento da imagem recebida a fim de obter a região de interesse para uma predição mais precisa do modelo.

```
cv2.putText(original, label, (162, 40-10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2, cv2.LINE_AA)
cv2.rectangle(original, (162, 40), (162+128, 40 + 128), (0, 255, 0), 2)
cv2_imshow(original)
```



Assim como também foi requisitado um exercício para realizar o reconhecimento em várias faces na mesma foto, sendo necessário criar um bloco de iteração onde fizesse o processo para cada face encontrada no cascade classifier.

```
1/1 [==============================] - 0s 35ms/step
[7.8443103e-03 1.5464441e-04 7.0808366e-02 7.4692351e-01 1.6651674e-03
 1.7088430e-01 1.7196917e-03]
0.7469235
feliz
```

```
1/1 [==============================] - 0s 36ms/step
[1.0294173e-02 1.7985894e-04 3.2855803e-01 2.4662589e-01 1.2978880e-03
 4.1240034e-01 6.4379716e-04]
0.41240034
surpreso
```

```
1/1 [==============================] - 0s 37ms/step
[2.0797981e-03 9.4365978e-06 1.3799014e-03 9.0104491e-01 2.1679797e-03
 1.4511482e-03 9.1866873e-02]
0.9010449
feliz
```

Seção 3

A seção 3 aborda redes convolucionais no reconhecimento de emoções, provando se util ao utilizar a convolução para extrair características úteis da imagem.

Para testes foi utilizado o dataset fer2013, utilizado em uma competição kaggle em 2013, o dataset contém 35.887 imagens, categorizadas em 7 emoções sistematizados no arquivo fer2013.csv

Durante o tratamento do dataset foram explicados alguns conceitos interessantes abordados de forma mais rasa em cards anteriores, como a relação entre as partições de treinamento, teste e validação

Um conceito novo foi apresentado neste card: os regularizers, que acrescentam um termo à otimização a fim de evitar o overfitting. Foi apresentado o regularizer L2, que adiciona um termo proporcional a soma do quadrado dos pesos

Alguns callback novos foram introduzidos, como o earlystopping, que avalia as melhorias do modelo após cada época e caso o mesmo não apresente nenhuma melhoria significativa após um certo número de épocas, o processo de treinamento é encerrado.

Foram efetuados testes com imagens e os resultados foram expostos através do openCV, onde a cor da moldura e do texto mudaram conforme a expressão identificada.

Os resultados divergiram no treinamento do modelo, pois no caso do instrutor houve ocorrências de raiva ou tristeza.

exemplo de predição realizada em uma imagem

Seção 4

Na seção 4 são abordados aspectos básicos de redes neurais, conteúdo mostrado nos cards 11-12, uma escolha inusitada, visto que ele aplica todos os conhecimentos para somente depois explicar a teoria, fazendo com que um aluno inexperiente não conseguisse acompanhar a prática da seção 3.

Para fins de teste do face cascade e como o modelo retira as regiões de interesse, além dos testes com espaços de cores realizados acima, foi criado um bloco de código onde se escolhe uma emoção do array de expressões, após a escolha todos a faces que apresentem tal emoção são printadas, porém somente na região de interesse.

```
emocao = int(input())
for (x, y, w, h) in faces:


  roi_gray = gray[y:y + h, x:x + w]
  roi_gray = roi_gray.astype('float') / 255.0
  cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray, (48, 48)), -1), 0)
  prediction = loaded_model.predict(cropped_img)[0]
  resul = np.argmax(prediction)


  if (resul == emocao):
    roi_gray= (roi_gray * 255).astype('uint8')
    cv2_imshow(roi_gray)

cv2_imshow(original)
```

```
3
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
```