Análise de Sentimentos em imagens

Carlos Eduardo e Luiz Sacramento

1

Abstract. *The work aims to study techniques aimed at analyzing feelings in images. Most are based on capturing and classifying facial expressions. Aspects related to the classification of emotions will be explored, as well as resources needed to use this technology. The main point will be to realize how effective the classification method and the unit test results are. During the article, some concepts of multimedia systems, a computational model and their respective classifications belonging to their ecosystem will be discussed. In addition, fundamentals of image processing will be addressed, as well as aspects related to the development of multimedia systems.*

Resumo. *O trabalho visa estudar técnicas voltadas à análise de sentimentos em imagens. Em sua maioria, baseiam-se na captura e classificação de expressões faciais. Será explorado aspectos referentes à classificação das emoções, assim como recursos necessários para utilização dessa tecnologia. O ponto principal será perceber o quão eficaz é o método de classificação e os resultados dos testes unitários. Durante o artigo, serão abordados alguns conceitos de sistemas multimídia, um modelo computacional e suas respectivas classificações pertencentes ao seu ecossistema. Além disso, serão abordados fundamentos de processamento de imagens, assim como aspectos relacionados ao desenvolvimento de sistemas multimídia.*

1. Objetivo

Análisar e descrever os testes de análise de sentimento em imagem utilizando TensorFlow feito pela equipe do Google Brain, essa biblioteca de código aberto foi criada para treinamento de redes neurais e aprendizado de máquina utilizando a licença do apache em 2015. E para o teste, a biblioteca Pytest 3.6.4.

2. Pergunta norteadora

Quais os principais desafios encontrados no algoritmo de análise de sentimento em imagem, utilizado nesse experimento?

3. Metodologia

A metodologia foi dividida em 5 etapas, sendo essas seguidas de maneira sequenciadas, ou seja, uma etapa executada obrigatoriamente após a outra. O planejamento é usar um banco com 35.887 imagens(FER2013, é Opensource criado pelo Pierre-Luccarrier e Aaron Courville), foi utilizado um .xml específico para trabalhar com detecção de faces.

FER2013 é um conjunto de dados bem estudado e tem sido usado em competições ICML e vários trabalhos de pesquisa. É um dos conjuntos de dados mais desafiadores, com precisão de nível humano[...]”(P-2, BAI, Charles; KHANZADA, Amil; CELEP-CIKAY, Ferhat)

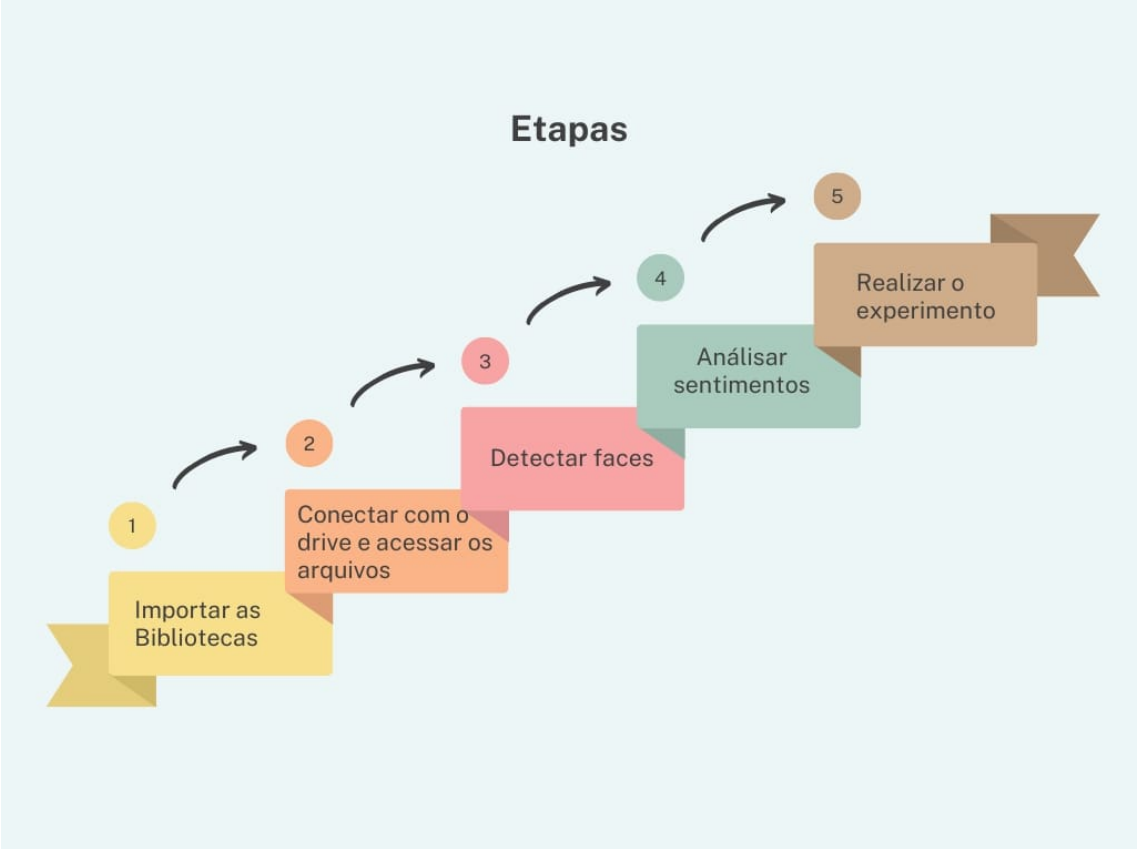

Figure 1. Banco de imagens

3.1. ETAPA 1 - Importando as Bibiotecas

```
import cv2
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow
import zipfile
```

Listing 1. Etapa 1

Todas essas bibliotecas, caso já não estejam instaladas,podem ser adquiridas via PIP A biblioteca pandas é usada para se trabalhar primordialmente com análise de dados numéricos, os quais serão desdobradas no decorrer dessa metodologia. Segue abaixo o comando para a sua instalação do Pandas 1.5.0

```
pip install pandas
```

Listing 2. Pandas

A OpenCV 4.6.0 é a biblioteca que será utilizada para o processamento de imagens, nelaque se encontra o método cv, usado para a leitura das imagens. Segue abaixo a instruçãopara sua instalação:

```
pip3 install opencv-python
```

Listing 3. OpenCv

Em seguida,o Numpy 1.22.0 também será instalado, essa biblioteca para trabalhar com manipulações matemáticas e matrizes, algo bem utilizado na hora de verificar o posicionamento das fotos.

```
pip install numpy
```

Listing 4. Numpy

A Zipfile, é uma biblioteca utilizada para manipular arquivos em zip, nesse contexto, o arquivo utilizado será alguns modelos para detecção de imagens adquiridos através de um conjunto de modelos treinados disponibilizados pelo próprio OpenCV.org.

3.2. ETAPA 2 - Conectando com o drive e acessando os arquivos

```
from google.colab import drive
drive.mount('/content/gdrive')
```

Listing 5. Conexão com o Drive

Nessa etapa, é feita a conexão entre o Colab e a conta do drive utilizada para banco de imagens e o arquivo com os modelos usados para detecção de faces.

4. ETAPA 3 - Detecção de faces

O algoritmo precisa de duas imagens, uma positiva (imagens de rostos) e outra negativa (imagens sem rostos) para treinar o classificador. Em seguida, ele extrai recursos através de um artifício matemático para subtração da soma dos pixels sob o retângulo branco da soma dos pixels sob o retângulo preto. O próximo passo seria encontrar a soma dos pixels sob os retângulos branco e preto. A grosso modo, o classificador irá utilizar modelos matemáticos aritméticos para extração de recursos e identificação dos rostos através de imagens pretas e brancas.

```
face_cascade =  cv2.CascadeClassifier('/content/drive/MyDrive/
    documentos_python/haarcascade_frontalface_default.xml')
faces = face_cascade.detectMultiScale(gray,1.1,3)
faces
```

Listing 6. Detecção de faces

4.1. ETAPA 4 - Análise de sentimentos

Antes de tudo, foi-se utilizada uma técnica chamada Grayscale que serve para converter a imagem em cinza, seu principal objetivo é reduzir ruidos nas imagens. Junto a isso há a segunda vantagem que é redução de tamanho da foto, após conversão em escala cinza, já que não é preciso trabalhar com as escalas de cores BGR ou RGB. Por fim, é definido o sentimento presente em cada uma das imagens, com o código abaixo:

```
import numpy as np

for(x,y,w,h) in faces:
  cv2.rectangle(original,(x,y),(x+y,y+h),(0,255,0),1)
cv2_imshow(original)
```

Listing 7. Análise de setimentos

4.2. ETAPA 5 - Realizar o experimento

O próximo passo é a realização do teste unitário, comentar as saídas encontradas ao análisar tais fotos e se realmente o algoritmo cumpre o prometido.

5. Figuras

A figura abaixo nostra tanto o banco de imagens OpenSource, com algumas de suas imagens salvas, assim como as imagens detectadas e analisadas, cada uma com seus respectivos sentimentos.

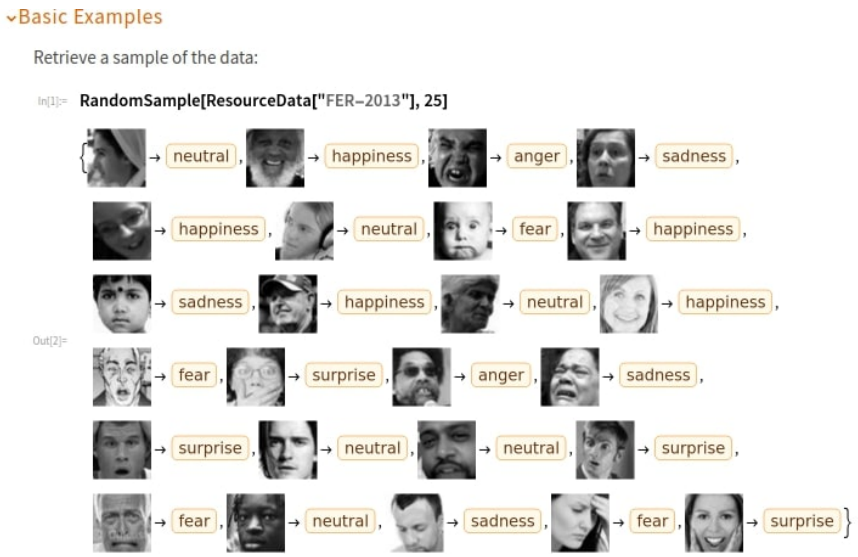

Figure 2. Banco de imagens

Figure 3. Detecção de emoções

6. Resultados e discussões

Dentre os resultados obtidos, o primeiro a ser citado é o teste unitário, o qual foi utilizado primordialmente no momento da digitalização de imagens.

```
1
2 import matplotlib.pyplot as plt
3 %%file test_file_load.py
4
5 import pytest
6 import pandas as pd
7 import logging
8 logging.basicConfig(level=logging.INFO)
9
10 imagem = cv2.imread('Material/testes/teste02.jpg')
11 cv2_imshow(imagem)
12
13 def teste(filepath):
14   try:
15     if len(imagem)>0:
16     # Do something if
17       logging.info(" Succesffully loaded file")
18       return 1
19
20   except FileNotFoundError:
21       # File doesn't exist
22       logging.info(" File not found/ Incorrect file path")
23       return 2
24
25   else:
26     # Return error loading file
27     logging.info(" File load failed")
28     return 3
29
30
31 class TestFileloader:
32   def testthis(self):
33     status  = teste('Material/testes/teste02.jpg')
34     assert status ==2
```

Listing 8. Código teste

Acima há a estrutura utilizada para teste de software com algumas possíveis excessões e erros.

```
1
2
3 ============================= test session starts
    ==============================
4 platform linux -- Python 3.7.15, pytest-3.6.4, py-1.11.0, pluggy-0.7.1
5 rootdir: /content, inifile:
6 plugins: typeguard-2.7.1
7 collected 2 items
8
9 test_file_load.py EF
      [100%]
10
11 ================================== ERRORS
    ==================================
12 _____________________________ ERROR at setup of teste
    _____________________________
13 file /content/test_file_load.py, line 8
```

```
14   def teste(filepath):
15 E       fixture 'filepath' not found
16 >       available fixtures: cache, capfd, capfdbinary, caplog, capsys,
       capsysbinary, doctest_namespace, monkeypatch, pytestconfig,
       record_property, record_xml_attribute, record_xml_property, recwarn
       , tmpdir, tmpdir_factory
17 >       use 'pytest --fixtures [testpath]' for help on them.
18
19 /content/test_file_load.py:8
20 ================================== FAILURES
       ==================================
21 _____________________________ TestFileloader.testthis
       ______________________________
22
23 self = <test_file_load.TestFileloader object at 0x7f87b0798d10>
24
25     def testthis(self):
26 >     status  = teste('Material/testes/teste02.jpg')
27
28 test_file_load.py:28:
29 _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
       _ _ _ _
30
31 filepath = 'Material/testes/teste02.jpg'
32
33     def teste(filepath):
34       try:
35 >       if len(imagem)>0:
36 E       NameError: name 'imagem' is not defined
37
38 test_file_load.py:10: NameError
39 ====================== 1 failed, 1 error in 0.41 seconds
       ======================
```

Listing 9. Saída do teste

Esse primeiro erro acusa uma variável de não respeitar boas práticas de criação ou incialização, ou seja, não foi definida adequadamente.

```
1
2
3 ============================= test session starts
       ==============================
4 platform linux -- Python 3.7.15, pytest-3.6.4, py-1.11.0, pluggy-0.7.1
5 rootdir: /content, inifile:
6 plugins: typeguard-2.7.1
7 collected 0 items / 1 errors
8
9 ================================== ERRORS
       ==================================
10 _____________________ ERROR collecting test_file_load.py
       _____________________
11 test_file_load.py:7: in <module>
12     imagem = cv2.imread('Material/testes/teste02.jpg')
13 E   NameError: name 'cv2' is not defined
14 !!!!!!!!!!!!!!!!!!!! Interrupted: 1 errors during collection
       !!!!!!!!!!!!!!!!!!!!
```

```
15 ============================ 1 error in 0.39 seconds
    ============================
```

Listing 10. Saída do teste 2

Esse erro, diz respeito a utilização de uma biblioteca que não foi importada adequadamente. Vale salientar também que outros cuidados referentes a posicionamento de imagens e tamanho precisaram ser seguidos. Exemplo, imagens de rostos com leve inclinação não foram detectados ou ganharam classificações descontextualizadas. As que têm rostos maiores que 48x48 não foram detectados, algumas com fundo não sólido ou transparente também não foram bem sucedidos nas detecções.

7. Conclusão

A principal crítica a esse modelo é a ineficácia em algum contexto que utilize resoluções maiores ou ângulos diferentes e até mesmo em uma situação condizente a captura de fotos em tempo real, a qual as pessoas estejam se movimentando livremente. Entretanto, com exceção do que foi dito acima, o banco de imagens FER2013 cumpre razoavelmente bem aquilo que promete junto a biblioteca Tensorflow.

8. References

ATITAYA, Y; MATTHEW, N. D; TEERADAJ, R. **Multimodal Sentiment Analysis on Video Streams using Lightweight Deep Neural Networks**. In: Scitepress p 442-451, Disponível em: https://www.scitepress.org/Papers/2021/103044/103044.pdf, acesso em 19/09/2022.

BALTEZ, R; CESAR, C; CRISTINA, R; GONÇALVES, C; CORREA, G. **Análise de sentimento para Reviews Apresentados em Vídeos: Modelo de Redes Neurais Treinado em Base de Reviews Escritos**, In: EAESP, p 2-19, Disponível em: https://pesquisaeaesp.fgv.br/sites/gvpesquisa.fgv.br/files/arquivos/gus-4-23684-61155-1-pb.pdf, acesso em 20/09/2022.

BARDIN, Laurence. **ANÁLISE DE CONTEÚDO. Tradução de Luis Antero Reto e Augusto Pinheiro**. São Paulo: Edições 70 LTDA, 1997, 9 45. Disponível em: https://ia802902.us.archive.org/8/items/bardin-laurence-analise-de-conteudo/bardin-laurence-analise-de-conteudo.pdf acesso em 31/08/2022

IA EXPERT ACADEMY. **Reconhecimento de Emoções com TensorFlow**. Disponível em: https://www.youtube.com/watch?v=lRV4MGP$_w Mkt = 309s, acessoem 19/09/2022$.

LI, Z; LI, R; JIN, G. **Sentiment Analysis of Danmaku Videos Based on Naıve Bayes and Sentiment Dictionary**. In: IEEE, p 75073 - 75084, Disponível em: https://ieeexplore.ieee.org/document/9060892, Acesso em 23/09/2022.

PEREZ, V; MIHALCEA, R; PHILIPPE, L. **Utterance-Level Multimodal Sentiment Analysis. In: Aclanthology**, p 973-982, Disponível em: https://aclanthology.org/P13-1096.pdf, acesso em 09/08/2022.

WANG, W; WU, J; KAZUAKI, F; WADA, S; KURIHARA, S. VAE-Based **Adversarial Multimodal Domain Transfer for Video-Level Sentiment Analysis**. In:

IEEE, p 51315 - 51324, Disponível em: https://ieeexplore.ieee.org/document/9772490, acesso em 11/10/2022.

AARON, Courville; CARRIER, **Pierre. The Facial Expression Recognition 2013 (FER-2013)**. Disponível em: https://datarepository.wolframcloud.com/resources/FER-2013, acesso em 15/10/2022.

BAI, Charles; KHANZADA, Amil; CELEPCIKAY, Ferhat. **Reconhecimento de Expressão Facial com Deep Learning**, p 2, Disponível em: http://cs230.stanford.edu/projects$_w inter_2 020/reports/32610274.pdf, acessoem 21/10/2022.$