

Classificação de Imagens no Conjunto de Dados MPEG7 Modificado

Arthur Roberto de Paula Neto - 8079

November 20, 2024

1 Introdução

Este trabalho tem como objetivo a classificação de imagens no conjunto de dados MPEG7 modificado utilizando técnicas de aprendizado supervisionado.

2 Importação de Bibliotecas e Configuração Inicial

O primeiro passo foi importar as bibliotecas necessárias para o processamento de imagens e construção do modelo. As bibliotecas utilizadas são:

- `os` - para manipulação de arquivos e diretórios;
- `numpy` - para operações numéricas;
- `skimage` - para processamento de imagens;
- `sklearn` - para construção e avaliação do modelo;
- `matplotlib` e `seaborn` - para visualização de dados.

O código para importação das bibliotecas é o seguinte:

```
1 import os
2 import numpy as np
3 from skimage import util, transform,
  filters, color, measure, morphology
4 from sklearn import model_selection,
  neighbors, metrics, preprocessing
5 import matplotlib.pyplot as plt
6 import seaborn as sns
7 import pandas as pd
8 import cv2
```

3 Carregamento e Pré-processamento dos Dados

As imagens foram carregadas a partir de um diretório no Google Drive, e as imagens foram redimensionadas para 1/4 do tamanho original para reduzir o custo computacional.

```
1 from google.colab import drive
2 drive.mount('/content/drive')
3
```

```
4 ds_path = '/content/drive/My Drive/
  mpeg7_mod'
5 classes_list = os.listdir(ds_path)
6 image_list = []
7 label_list = []
8 filename_list_ = []
9
10 for classe in classes_list:
11     filename_list = os.listdir(os.path.join(
12         ds_path, classe))
13     for filename in filename_list:
14         img_temp = plt.imread(os.path.join(
15             ds_path, classe, filename))
16         img_temp = transform.resize(
17             img_temp, (img_temp.shape
18                 [0]//4, img_temp.shape[1]//4),
19                 anti_aliasing=True)
20         image_list.append(img_temp)
21         label_list.append(classe)
22         filename_list_.append(filename)
```

4 Seleção das Imagens e Rótulos

Foi realizada a seleção das primeiras 6 imagens de cada classe, garantindo um conjunto de dados balanceado para o treinamento do modelo.

```
1 seg_list_temp = []
2 filename_list_temp = []
3
4 for i, class_name in enumerate(classes_list):
5     seg_list_temp += [seg_list[j] for j in
6         np.where(label_list==class_name)
7         [0][:6]]
8     filename_list_temp += [filename_list_[j]
9         for j in np.where(label_list==
10             class_name)[0][:6]]
```

5 Cálculo das Características das Imagens

Foi realizada a segmentação binária das imagens e, em seguida, extraídas as características de cada objeto segmentado, como área, maior eixo, menor eixo, solidez e excentricidade.

```

1 features = ['area', 'major_axis', 'minor_axis', 'solidity', 'eccentricity']
2 seg_list = []
3 list_label = []
4 feature_mat = []
5
6 for i, (image, label) in enumerate(zip(image_list, label_list)):
7     img_float = util.img_as_float(image)
8     if len(img_float.shape) == 2:
9         img_gray = img_float
10    else:
11        img_gray = color.rgb2gray(img_float)
12    img_seg = morphology.remove_small_objects(img_gray > 0, 1000)
13    seg_list.append(img_seg)
14    im_lbl = measure.label(img_seg)
15    props = measure.regionprops(im_lbl)
16
17    for prop in props:
18        area = prop.area
19        major_axis = prop.major_axis_length
20        minor_axis = prop.minor_axis_length
21        solidity = prop.solidity
22        eccentricity = prop.eccentricity
23        feature_list = [area, major_axis, minor_axis, solidity, eccentricity]
24    feature_mat.append(feature_list)

```

6 Transformação e Normalização das Características

Foi realizada a normalização das características para garantir que todas possuam a mesma escala, utilizando a média e o desvio padrão do conjunto de treinamento.

```

1 X_train_mean = X_train.mean(0)
2 X_train_std = X_train.std(0)
3
4 X_train_norm = (X_train - X_train_mean) / X_train_std
5 X_test_norm = (X_test - X_train_mean) / X_train_std

```

7 Construção do Modelo de Classificação

Utilizamos dois modelos de aprendizado supervisionado: o K-Nearest Neighbors (K-NN) e o Perceptron. Para ambos, as imagens foram classificadas com base nas características extraídas.

7.1 Classificador K-NN

Foi treinado um classificador K-NN com $k = 3$.

```

1 clf = neighbors.KNeighborsClassifier(n_neighbors=3)
2 clf.fit(X_train_norm, y_train)
3 pred = clf.predict(X_test_norm)

```

7.2 Classificador Perceptron

Em seguida, um Perceptron foi treinado utilizando os dados normalizados.

```

1 clf = Perceptron(max_iter=1000, eta0=0.01, random_state=42)
2 clf.fit(X_train_norm, y_train)
3 pred = clf.predict(X_test_norm)

```

8 Avaliação do Modelo

A avaliação do modelo foi feita utilizando as métricas de precisão, recall e f1-score, além de uma matriz de confusão para verificar o desempenho do classificador.

```
print(classification_report(y_test, pred))
```

A matriz de confusão foi gerada para visualização dos acertos e erros do classificador:

```

1 conf_matrix = confusion_matrix(y_test, pred)
2 sns.heatmap(conf_matrix, annot=True, cmap="Blues", fmt='g')

```

References

- [1] Repositórios do Prof. João Fernando Mari