UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
IMD1101 - Aprendizado de Máquina – 2025.1
Anne Magály de Paula Canuto

Extração de características baseadas em HOG e CNN
Aplicação do Método PCA
Aplicação de Técnica Supervisionada - kNN

A identificação de objetos em imagens tem se tornado um desafio interessante. A partir disso, sua aplicação tem sido verificada em várias de interesse. Dentre os descritores de características mais utilizados estão o *Histogram of Oriented Gradients* (HOG) e a Convolutional Neural Network (CNN).

O descritor HOG pode ser visto como um descritor de gradientes ou das direções das bordas, e tem aplicação em uma série de tarefas e pode ser usado em diversas linguagens, como C++, Python, Java através de bibliotecas como OpenCV ou Scikit-image.

A Rede Neural Convolucional é um algoritmo de Deep Learning desenvolvido para trabalhar com imagens e vídeos. Ela recebe imagens como entradas, extrai e aprende os recursos da imagem, e classifica com base nos recursos aprendidos.

Visando a elaboração do projeto da disciplina de Aprendizado de Máquina, escolheu-se uma base de imagens de pets (cachorros e gatos) que pode ser acessa através do seguinte link:

- *https://www.robots.ox.ac.uk/~vgg/data/pets/*

Para a extração de características das referidas imagens, cada grupo deverá seguir a seguinte metodologia:

1. Ler as imagens baixadas do site de acordo com o sorteio feito em sala de aula;
2. Redefinir o tamanho de cada uma delas para 256 x 256 pixels e 128 x 128 pixels;
3. Aplicar o algoritmo de HOG para gerar quatro versões diferentes das imagens, utilizando as seguintes configurações:

a.
```
hog_features = []
for filename in uploaded_images.keys():
  image = imread(filename)
  image_resized = resize(image, (128,128))
  fd, hog_image = hog(image_resized, orientations=9, pixels_per_cell=(16, 16),
                      cells_per_block=(2, 2), visualize=True, channel_axis=-1)
  hog_features.append(fd)
```

b.
```
hog_features = []
for filename in uploaded_images.keys():
  image = imread(filename)
  image_resized = resize(image, (128,128))
  fd, hog_image = hog(image_resized, orientations=9, pixels_per_cell=(20, 20),
                      cells_per_block=(2, 2), visualize=True, channel_axis=-1)
  hog_features.append(fd)
```

c.
```
hog_features = []
for filename in uploaded_images.keys():
  image = imread(filename)
  image_resized = resize(image, (256,256))
  fd, hog_image = hog(image_resized, orientations=9, pixels_per_cell=(16, 16),
                      cells_per_block=(2, 2), visualize=True, channel_axis=-1)
  hog_features.append(fd)
```

d.
```
hog_features = []
for filename in uploaded_images.keys():
  image = imread(filename)
  image_resized = resize(image, (256,256))
  fd, hog_image = hog(image_resized, orientations=9, pixels_per_cell=(20, 20),
                      cells_per_block=(2, 2), visualize=True, channel_axis=-1)
  hog_features.append(fd)
```

4. Salvar os quatro datasets originários da transformação (em CSV), contendo +/- 800 imagens, sendo 400 imagens de cachorros e 400 imagens de gatos de acordo com o sorteio de cada grupo;

5. Aplicar os modelos treinados de CNN (VGG16 e VGG19) nas 800 imagens de cada grupo, explorando os valores para *pooling* (‘avg’ e ‘max’), além de redefinir o tamanho de cada uma das imagens para 256 x 256 pixels e 128 x 128 pixels utilizando as configurações abaixo:

a.
```
## Carregar o modelo de CNN pre-treinado (VGG16)
model_16 = VGG16(weights='imagenet', include_top=False, pooling='avg')
```

b.
```
####### transformações ######
cnn_features = []
for filename in uploaded_images.keys():
  imagem = io.imread(filename)
  ###print(imagem.shape)
  print(filename)
  imagem_reduzida = resize(imagem, (128,128))
  x = image.img_to_array(imagem_reduzida)
  x = np.expand_dims(x, axis=0)
  x = preprocess_input(x)
  features = model_16.predict(x)
  features_flatten_vgg16 = features.flatten()
  cnn_features.append(features_flatten_vgg16)
```

c.
```
## Carregar o modelo de CNN pre-treinado (VGG19)
model_19 = VGG19(weights='imagenet', include_top=False, pooling='max')
```

d.
```
####### transformações ######
cnn_features = []
for filename in uploaded_images.keys():
  imagem = io.imread(filename)
  ###print(imagem.shape)
  print(filename)
  imagem_reduzida = resize(imagem, (256,256))
  x = image.img_to_array(imagem_reduzida)
  x = np.expand_dims(x, axis=0)
  x = preprocess_input(x)
  features = model_19.predict(x)
  features_flatten_vgg19 = features.flatten()
  cnn_features.append(features_flatten_vgg19)
```

6. De posse das doze (12) bases de dados (quatro obtidas através da utilização do HOG, quatro obtidas através do VGG16 e quatro obtidas através do VGG19), utilize as seguintes opções de treinamento e teste:
    a. 10-fold cross-validation;
    b. split-percentage: 70/30.

Na sequência, execute experimentos com a técnica k-NN (nearest neighbors), de tal forma que você possa escolher diferentes valores para alguns parâmetros pertencentes às técnicas estudadas.

Para o k-NN, você poderá variar o número de vizinhos de 1 a 10 (ou qualquer intervalo), de acordo com a Figura 1. Para cada execução, de acordo com os valores de *k* e referentes a cada uma das opções de treinamento e teste, guarde o resultado da **acurácia** na célula correspondente, como mostra a figura abaixo. Por último, calcule a média e o desvio padrão para cada valor de *k*.

		k-NN (Acurácia)									
Bases	**Treino/Teste**	**k=1**	**k=2**	**k=3**	**k=4**	**k=5**	**k=6**	**k=7**	**k=8**	**k=9**	**k=10**
HOG_128_16x16 (1765)	70/30										
	10-fold CV										
HOG_128_20x20 (901)	70/30										
	10-fold CV										
HOG_256_16x16(8101)	70/30										
	10-fold CV										
HOG_256_20x20 (4357)	70/30										
	10-fold CV										
CNN_VGG16_128_avg (513)	70/30										
	10-fold CV										
CNN_VGG16_128_max (513)	70/30										
	10-fold CV										
CNN_VGG16_256_avg (513)	70/30										
	10-fold CV										
CNN_VGG16_256_max (513)	70/30										
	10-fold CV										
CNN_VGG19_128_avg (513)	70/30										
	10-fold CV										
CNN_VGG19_128_max (513)	70/30										
	10-fold CV										
CNN_VGG19_256_avg (513)	70/30										
	10-fold CV										
CNN_VGG19_256_max (513)	70/30										
	10-fold CV										
Média =>		#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!		#DIV/0!
Desv. Pad. =>		#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!		#DIV/0!

Figura 1. Resultados experimentais do k-NN (variação de k de 1 a 10).

De posse dos valores dos experimentos, escolha os seis melhores e suas respectivas bases. Para essas, aplique o método PCA com 10 componentes de tal forma que obterá como resultante seis novas bases. Ao final, você terá doze (12) bases, sendo seis originadas através do HOG e CNN e seis através do PCA. Altere a planilha (Figura 1), de forma que as novas seis bases possam tomar os lugares das seis piores, e na sequência, execute os experimentos para as novas bases.

Observe que depois de fazer as execuções e guardar todos os valores dos experimentos, crie um documento com as duas tabelas (a tabela acima e a tabela com as bases do PCA). Além disso, responda as seguintes perguntas:

1. Qual foi o melhor método de extração HOG ou CNN? Tente explicar o motivo do comportamento da sua base de dados
2. O que aconteceu com a acurácia quando diminuímos o número de atributos com o PCA? Explique o comportamento dos modelos.
3. Em relação ao k-NN, o que aconteceu quando aumentamos o número de vizinhos? Explique o motivo deste comportamento

Bom trabalho!