Exercício Programa 2

O EP2 consiste em duas partes, para cada parte deve-se criar um Jupyter Notebook (JN).

No início do primeiro JN, mostre a tabela sumária das suas imagens, conforme mostrado ao final do enunciado do EP1. Para cada classe, mostre a quantidade de variações de fundo, iluminação, repetições e total de amostras.

Parte 0: *Requisitos do projeto*

Foi necessária a criação dos requisitos mínimos do projeto devido a alguns equívocos já vistos nas entregas do primeiro EP:

- Para correção, não apenas será visto o código, como o notebook também será executado. Certifique que não existirá erros ao executarmos todas as células do JN;
- Todos os arquivos necessários para executar seu JN **devem** estar no Github, como informado no texto do EP1, se as imagens do seu projeto não foram colocadas no EP1, certifique-se que estejam no EP2;
- **Não** devem existir figuras **duplicadas**. Por exemplo, para você saber quais classes pertencem a determinada imagem, deve-se olhar os metadados;
- Existem 3 opções de metadados, em nenhuma delas o metadado deve estar nos dados do arquivo de imagem, favor atentar as opções dadas no EP1, se estiver errado, refaça no EP2;
- No momento, apenas podem ser utilizadas as bibliotecas **Matplotlib, NumPy, os, pandas**. O cv2 pode ser utilizado para a leitura, escrita, redução e conversão para escala de cinza da imagem;
- A restrição de bibliotecas é para garantir que o processamento da imagem seja feito por vocês. Apesar de não ser permitido o uso de funções de bibliotecas que já tenham todo o processamento criado, é possível usar bibliotecas que facilitam cálculos, como a multiplicação de matrizes e cálculo da convolução que existe no numpy;
- Lembre-se que os EPs são cumulativos, erros não corrigidos antes, podem afetar resultados dos outros EPs.

Não é obrigatório, mas recomendo fortemente a diminuição do tamanho das imagens, para quem está com arquivos grandes, antes da conversão para a escala de cinza. Não apenas ajudará no armazenamento, mas também para o tempo de execução do código. Fiz um teste no colab, e com a base de dados de algumas entregas tive falta de memória RAM na primeira parte do EP2.

Acho que podemos colocar como condição, que a maior imagem tenha 1 Mb. O código abaixo, pode ser usado para reduzir o tamanho, mantendo a proporção:

```
from cv2 import resize
img_redux = resize(img, (0,0), fx=0.5, fy=0.5)
```

Neste caso, deve-se reduzir de aproximadamente metade do tamanho, pode-se usar valores menores, apenas fique atento que os valores de fx e fy devem ser iguais. Use os mesmos fx e fy em todas as imagens.

Parte 1: *Data augmentation*

O primeiro JN, deve conter funções para fazer *data augmentation* do *dataset* do EP1, e sua execução por cada classe. Mais informações sobre *Data Augmentation* são encontradas aqui:

- https://github.com/aleju/imgaug (a biblioteca é para entendimento do problema, não pode ser usada).
- https://www.kaggle.com/parulpandey/overview-of-popular-image-augmentation-packages

As funções de *data augmentation* serão aplicadas a imagens em níveis de cinza. Portanto, o dataset original precisará ser convertido para níveis de cinza. Há várias formas de fazer essa conversão. Recomendamos a multiplicação do vetor rgb_gray abaixo com a matriz rgb da imagem, como no código abaixo:

```
rgb_gray = [0.2989, 0.5870, 0.1140]
img_gray = np.dot(img, rgb_gray)

#Pode-se utilizar a função do OpenCV, também.
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Deve-se criar 5 funções de *data augmentation*. Cada função será aplicada a cada imagem do *dataset* do EP1. Assim, o `originalGrayDataset` terá as imagens originais em níveis de cinza e o `augmentedDataset` terá 5X o tamanho do original. Funções que deverão ser implementadas:

- RGB2gray (converter as imagens RGB originais em níveis de cinza);
- Contrast Stretching;
- Logaritmo da imagem (c = explicação abaixo);
- Exponencial da imagem (c, gamma = explicação abaixo);
- Laplaciano da imagem;
- Filtro da média implementado usando convolução.

Lembre-se que todas as funções de *data augmentation* devem ser feitas usando as fotos em níveis de cinza. Os metadados das imagens originais devem ser associados a novas imagens depois do processamento. Para os processamentos que contém hiperparâmetros, como log e exponencial, crie funções que recebam estes valores, e determine eles para cada uma das 4 iluminações (c e gamma) das fotos tiradas no EP1. A lista de fotos de determinada iluminação,

deve ser passada recuperando os nomes dos arquivos nos dados de metadados, em **nenhum momento deve-se passar a uma lista digitada com os nomes das imagens**.

Caso ainda não tenha criado, crie uma função com o código usado para a visualização no EP1 e mostre os novos datasets: `originalGrayDataset` e `augmentedDataSet`.

Obs: **notamos problemas ao salvar as imagens com a função do matplotlib (imsave). Por isso, sugerimos o uso da função imwrite da biblioteca cv2**:

```
from cv2 import imwrite
imwrite("nome do arquivo", img)
```

Parte 2: Normalização e análise da variação das classes

Deve-se criar um JN que realize a normalização das imagens de cada classe do `augmentedDataSet`. Deve-se usar a equalização de histograma como função inicial de normalização, gerando um `normalizedDataset`. As funções de análise de cada classe abaixo devem ser aplicadas ao `originalGrayDataset`, `augmentedDataset` e `normalizedDataset`. Assim, esse JN deve calcular e mostrar para cada dataset:

- Histograma médio de cada uma das iluminações para o `originalGrayDataset` e cada processamento diferente dentro do `augmentedDataSet (exp, log, etc). E o histograma após a normalização.` (O histograma médio deve ser a média das intensidades dos pixels de cada uma das imagens);
- Para cada histograma da média acima, mostre uma imagem de antes e depois da normalização;
- Ao final, discuta a mudança da normalização dos Datasets. Por exemplo, em quais iluminações tivemos maior diferença, ou se a normalização deixou mais ou menos nítido o objeto, etc;

Entrega

O cabeçalho que está no final deste arquivo (página 4) deve estar na primeira linha de cada um dos dois JN.

Todo o processo deve ser **automatizado**, não apenas a criação dos novos arquivos, mas também a criação das pastas e organização dos *datasets* em diretórios. Uma sugestão, é a criação das pastas com o nome de cada *dataset*, e subpastas para cada dos tipos diferentes de processamento. Por exemplo: augmentedDataset/Logaritmo.

Não é necessário o *upload* das imagens de *data augmentation* no *GitHub*. Apenas as fotos originais.

Ao final, teremos três novos datasets: **originalGrayDataset**, **augmentedDataset** e **normalizedDataset**. O arquivo de metadados pode ou não exigir que adicione cada uma das imagens nele, depende muito de qual estratégia cada um utilizou. Porém, todas as imagens devem ter a possibilidade de serem recuperadas com o uso dos metadados. Por exemplo, deve ser possível recuperar todas as imagens de determinada iluminação no augmentedDataset do exponencial.

Apenas para enfatizar, **nenhum** nome de imagem deve ser digitado no EP, as imagens devem ser recuperadas com as informações dos metadados.
Ao terminar o EP, submeta o link do *GitHub* na atividade do EP2.

Cabeçalho para a primeira célula do JN:

```
# @title
### EP2 MAC0417 / MAC5768
#####################################################################
#  AO PREENCHER ESSE CABEÇALHO COM O MEU NOME E O MEU NÚMERO USP,#
#  DECLARO QUE SOU O ÚNICO AUTOR E RESPONSÁVEL PELA RESOLUÇÃO    #
#  DESTE EP.                                                      #
#  TODAS AS PARTES FORAM DESENVOLVIDAS E IMPLEMENTADAS POR MIM,  #
#  SEGUINDO AS INSTRUÇÕES E QUE PORTANTO, NÃO CONSTITUEM         #
#  DESONESTIDADE ACADÊMICA OU PLÁGIO.                             #
#                                                                 #
#  DECLARO TAMBÉM, QUE SOU RESPONSÁVEL POR TODAS AS CÓPIAS       #
#  DESSE PROGRAMA, E QUE EU NÃO DISTRIBUI OU FACILITEI A         #
#  SUA DISTRIBUIÇÃO. ESTOU CIENTE QUE OS CASOS DE PLÁGIO E       #
#  DESONESTIDADE ACADÊMICA SERÃO TRATADOS SEGUNDO OS CRITÉRIOS   #
#  DEFINIDOS NO CÓDIGO DE ÉTICA DA USP.                           #
#                                                                 #
#  ENTENDO QUE JUPYTER NOTEBOOKS SEM ASSINATURA NÃO SERÃO        #
#  CORRIGIDOS E, AINDA ASSIM, PODERÃO SER PUNIDOS POR            #
#  DESONESTIDADE ACADÊMICA.                                       #
#                                                                 #
#                                                                 #
#  Nome :                                                         #
#  NUSP :                                                         #
#  Turma:                                                         #
#  Prof.:                                                         #
#####################################################################
```