

## PSI-5796: Aprendizagem Profunda para Processamento de Imagens

Primeiro período de 2022 Exercício-programa Prof. Hae

Data de entrega: 26/06/2022 até 24:00 horas

Redação alterada no dia 31/05/2022 – as partes alteradas estão em azul.

Obs. 1: Cada dia de atraso acarreta perda de 1 ponto no exercício.

Obs. 2: Este EP deve ser resolvido individualmente. EPs iguais receberão nota zero.

O objetivo deste exercício é classificar os 5 tipos de grãos de arroz (figura 1), do conjunto de dados: <https://www.muratkoklu.com/datasets/> e verificar se alinhar os grãos horizontalmente ajuda (ou não) nessa tarefa.

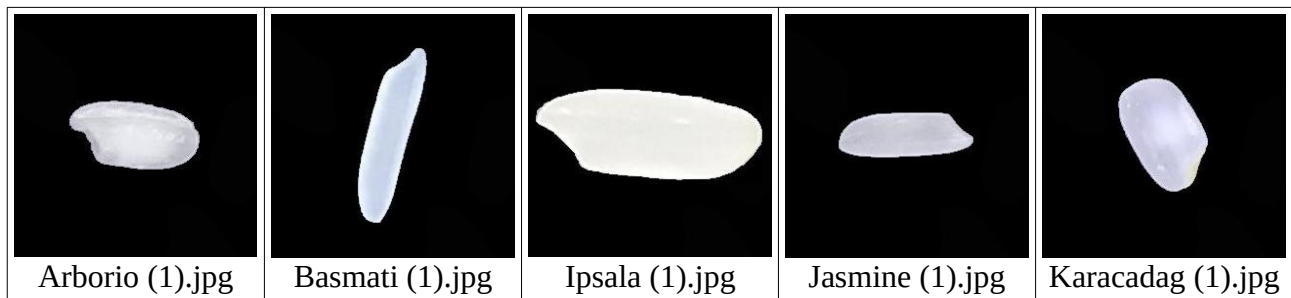


Figura 1: Os 5 tipos de grãos de arroz do conjunto de dados.

Peguei esse conjunto (que é muito grande) e armazenei somente as primeiras 1000 imagens de cada tipo, gerando o subconjunto abaixo:

[www.lps.usp.br/hae/psi5796/ep1-2022/rice\\_1to1000.zip](http://www.lps.usp.br/hae/psi5796/ep1-2022/rice_1to1000.zip)

### Parte 1 (4 pontos): Alinhar os grãos horizontalmente - “alinha.cpp”

Esta parte pode ser feita em C++ ou em Python.

1) Transforme a imagem original colorida (RGB) numa imagem binária e ache o seu centro de massa e a sua orientação, como mostra a segunda linha da figura 2. Para facilitar o trabalho de vocês, estou passando a função *findCenterAndOrientation* que acha o centro de massa e a orientação de uma imagem binária (uma imagem uint8 que só tem pixels com valores 0 ou 255). Para maiores detalhes, veja a discussão em:

<https://stackoverflow.com/questions/14720722/binary-image-orientation>

```
#include <cekeikon.h>
(...)
Point3d findCenterAndOrientation(Mat_<GRY> src) {
    Moments m = moments(src, true);
    double cen_x = m.m10/m.m00;
    double cen_y = m.m01/m.m00;
    double theta = 0.5 * atan2( 2*m.mu11, m.mu20-m.mu02);
    return Point3d(cen_x, cen_y, theta);
}
```

A chamada desta função, para uma imagem binária *a*, é:

```
Point3d p=findCenterAndOrientation(a);
printf("%f %f %f\n", p.x, p.y, p.z);
```

Após a chamada, *p.x* e *p.y* conterão o centro de massa da imagem *a* e *p.z* representará o ângulo em radianos da inclinação da figura.

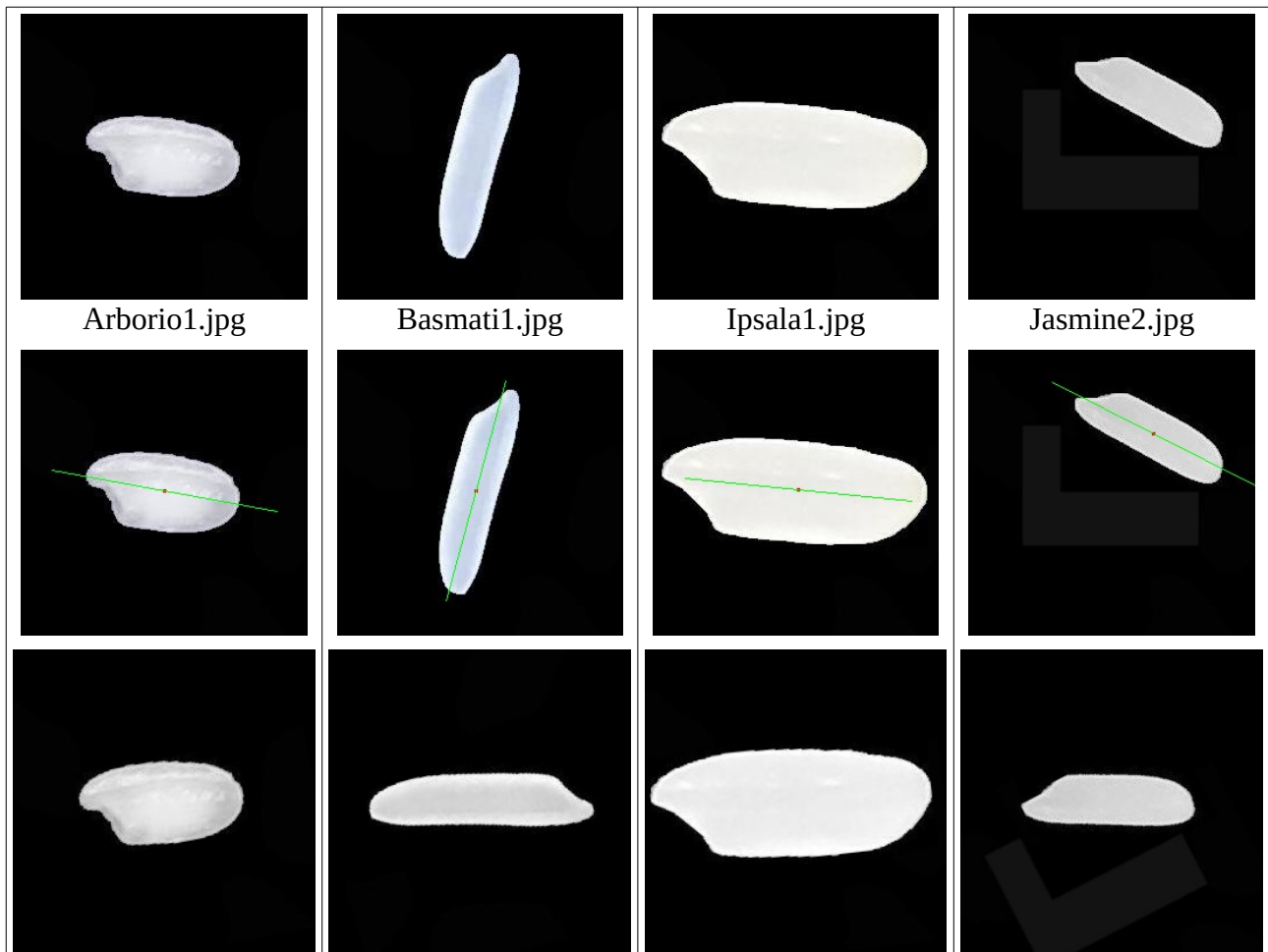


Figura 2: Achar o centro de massa e a orientação da imagem e transformá-la para que fique centrada e horizontal. Estas imagens estão no subdiretório *orientacao*.

2) Rotacione a imagem, para que a imagem esteja centrada e a orientação dos grãos seja horizontal, como mostra a terceira linha da figura 2. Teste obrigatoriamente o seu programa *alinha.cpp* para a imagem *Jasmine2.jpg* (do diretório *orientacao*), para se certificar de que o seu programa está colocando corretamente o centro de massa no centro da imagem.

Depois de verificar que o seu programa funciona, rotacione as 5000 imagens *.jpg* do conjunto de dados, para que todos eles fiquem centralizadas e alinhadas horizontalmente. O meu programa encontra todos os arquivos *.jpg* do diretório atual (“.”), os alinha horizontalmente e grava as imagens resultantes com extensão *.png*. Usei a extensão *.png* para denotar as imagens alinhadas (distinguindo-as das imagens originais com o mesmo nome mas com extensão *.jpg*). Sugiro que vocês façam o mesmo.

## Parte 2 (6 pontos): Classificar as imagens originais e alinhadas horizontalmente

Pode ser feita em Keras/Python.

O objetivo desta parte do exercício é classificar os 5 tipos de arroz usando rede neural convolucional, verificando se alinhar os grãos horizontalmente ajuda (ou não) nessa tarefa para qual número  $n_a$  de amostras de treinamento.

### 2.1) `classifica_jpg.py`

- Tome as  $n_a$  primeiras imagens originais de cada tipo de arroz como amostras de treinamento ( $ax$ ,  $ay$ ). Inicialmente, adote  $n_a=100$ .
- Se quiser, redimensione as imagens para diminuir a resolução, pois a resolução original  $250 \times 250$  pode ser grande demais.
- Treine uma rede neural convolucional com estrutura, parâmetros e *data augmentation* apropriadas para conseguir a maior taxa de acertos.
- Teste a sua rede nas imagens 101-1000 de todos os tipos de arroz ( $qx$ ,  $qy$ ).
- Qual foi a melhor taxa de acerto que conseguiu obter? Registre esta informação no vídeo e no relatório.

Nota: A função abaixo lê as imagens `.png` de *inic* até *fim* do conjunto de dados nos 5 subdiretórios ["Arborio", "Basmati", "Ipsala", "Jasmine", "Karacadag"], redimensiona-as para  $nl \times nc$  pixels e armazena-as em *AX*. Atribui as classes 0 a 4 para os 5 tipos de arroz e armazena esses rótulos em *AY*.

```
import cv2, sys
(...)
def le(diretorio, nl, nc, inic, fim):
    nclasses=len(diretorio)
    n=nclasses*(fim-inic+1)
    AX=np.empty((n, nl, nc), np.uint8);
    AY=np.empty((n, ), np.uint8);

    j=0; k=0
    for nome in diretorio:
        for i in range(inic, fim+1):
            st = nome + " (" + str(i) + ")."+"png"; st = nome+"/"+st; print(st)
            a=cv2.imread(st, 0)
            if a is None: print("Erro leitura", st); sys.exit()
            a=cv2.resize(a, (nl, nc), interpolation=cv2.INTER_AREA)
            AX[k, :, :]=a; AY[k]=j
            k+=1
        j+=1
    return AX, AY

diretorio=["Arborio", "Basmati", "Ipsala", "Jasmine", "Karacadag"]
AX, AY=le(diretorio, nl, nc, 1, 100)
```

### 2.2) `classifica_png.py`

Repita a mesma operação usando as  $n_a=100$  imagens alinhadas horizontalmente. Qual foi a melhor taxa de acerto que conseguiu obter? Registre esta informação no vídeo e no relatório.

### 2.3) Determinar $n_a$

Diminua o número de amostras  $n_a$  até que a taxa de erro das imagens não-alinhadas seja mais de quádruplo das imagens alinhadas. Qual é o valor de  $n_a$ ? Conclua (no relatório e no vídeo) se alinhar as imagens ajudou (ou não) a aumentar a taxa de acerto quando a quantidade de dados de treino  $n_a$  é pequena.

- Obs. 1:** Entregue os programas-fontes (alinha.cpp, classifica\_jpg.py, classifica\_png.py). Você pode enviar links para notebooks Google Colab em vez de programas .py.
- Obs. 2:** Entregue um documento PDF de no máximo 5 páginas (relatorio.pdf) descrevendo o funcionamento dos seus programas e os resultados obtidos. O envio do relatório é obrigatório (veja o anexo).
- Obs. 3:** Entregue um vídeo de no máximo três minutos explicando o funcionamento dos seus programa e resultados. No vídeo deve aparecer em algum momento o seu rosto e um documento seu. De preferência envie link para vídeo e não o vídeo em si, para não sobrecarregar o meu HD.
- Obs. 4:** Envie os arquivos e links em edisciplinas.

## **Anexo: Relatórios dos exercícios programas**

O mais importante numa comunicação escrita é que o leitor entenda, sem esforço e inequivocamente, o que o escritor quis dizer. O texto ficar “bonito” é um aspecto secundário. Se uma (pseudo) regra de escrita dificultar o entendimento do leitor, essa regra está indo contra a finalidade primária da comunicação. No site do governo americano [1], há regras denominadas de “plain language” para que comunicações governamentais sejam escritas de forma clara. As ideias por trás dessas regras podem ser usadas em outros domínios, como na escrita científica. Resumo abaixo algumas dessas ideias.

(1) Escreva para a sua audiência. No caso do relatório, a sua audiência será o professor ou o monitor que irá corrigir o seu exercício. Você deve focar na informação que o seu leitor quer conhecer. Não precisa escrever informações que são inúteis ou óbvias para o seu leitor.

(2) Organize a informação. Você é livre para organizar o relatório como achar melhor, porém sempre procurando facilitar o entendimento do leitor. Seja breve. Quebre o texto em seções com títulos claros. Use sentenças curtas. Elimine as frases e palavras que podem ser retiradas sem prejudicar o entendimento. Use sentenças em ordem direta (sujeito-verbo-predicado).

(3) Use palavras simples. Use o tempo verbal o mais simples possível. Evite cadeia longa de nomes, substituindo-os por verbos (em vez de “desenvolvimento de procedimento de proteção de segurança de trabalhadores de minas subterrâneas” escreva “desenvolvendo procedimentos para proteger a segurança dos trabalhadores em minas subterrâneas”). Minimize o uso de abreviações (para que o leitor não tenha que decorá-las). Use sempre o mesmo termo para se referir à mesma realidade (pode confundir o leitor se usar termos diferentes para se referir a uma mesma coisa). O relatório não é obra literária, não tem problema repetir várias vezes a mesma palavra.

(4) Use voz ativa. Deixe claro quem fez o quê. Se você utilizar oração com sujeito indeterminado ou na voz passiva, o leitor pode não entender quem foi o responsável (Ex: “Criou-se um novo algoritmo” - Quem criou? Você? Ou algum autor da literatura científica?). O site diz: “Passive voice obscures who is responsible for what and is one of the biggest problems with government writing.”

(5) Use exemplos, diagramas, tabelas, figuras e listas. Ajudam bastante o entendimento.

### **O relatório deve conter pelo menos as seguintes informações:**

#### *Identificação*

Nome, número USP, nome da disciplina, etc.

#### *Breve enunciado do problema*

Apesar do enunciado do problema ser conhecido ao professor/monitor, descreva brevemente o problema que está resolvendo. Isto tornará o documento compreensível para alguma pessoa que não tem o enunciado do EP à mão.

#### *Técnica(s) utilizada(s) para resolver o problema*

Descreva quais técnicas você usou para resolver o problema. Se você mesmo inventou a técnica, descreva a sua ideia, deixando claro que a ideia foi sua. Se você utilizou alguma técnica já conhecida, utilize o nome próprio da técnica (por exemplo, filtragem Gaussiana, algoritmo SIFT, etc.) juntamente com alguma referência bibliográfica onde a técnica está descrita. Use elementos gráficos como imagens intermediárias e diagramas, pois ajudam muito a compreensão. Não “copie-e-cole” código-fonte, a não ser que seja relevante. Use preferencialmente o pseudo-código.

---

1 <https://plainlanguage.gov/guidelines/>

### *Ambiente de desenvolvimento utilizado*

Em qual plataforma você desenvolveu o programa? Como o professor/monitor pode compilar o programa? Você utilizou que bibliotecas?

### *Operação*

Como o professor/monitor pode executar o programa? Que argumentos são necessários para a execução do programa? Há parâmetros que devem ser configurados? Quais arquivos de entrada são necessários? Quais arquivos de saída são gerados?

### *Resultados Obtidos*

Descreva os resultados obtidos. Qual é o tempo de processamento típico? O problema foi resolvido de forma satisfatória?

### *Referências*

Descreva o material externo utilizado, como livros/artigos consultados, websites visitados, etc.