trabalho_pib

August 3, 2024

[ ]:
```
import pandas as pd
import cupy as cp
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import os
import cv2
from cuml.cluster import KMeans
import torch
from torchclustermetrics import silhouette
```

[ ]:
```
# Função para carregar todas as imagens de um diretório
def load_images_from_folder(folder):
    images = []
    for filename in os.listdir(folder):
        img = mpimg.imread(os.path.join(folder, filename))
        if img is not None:
            images.append(img)
    return images
```

[ ]:
```
# Função para calcular silhouette_score médio para um dado k
def calculate_silhouette(images, k, color_space='RGB'):
    silhouettes = []
    for img in images:
        if color_space == 'HSV':
            img = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
        img_reshaped = img.reshape((-1, 3)).astype('float32')  # Converter para␣
 ↪float32
        img_reshaped_gpu = cp.asarray(img_reshaped)
        kmeans = KMeans(n_clusters=k, random_state=42).fit(img_reshaped_gpu)
        labels = kmeans.labels_
        img_tensor = torch.tensor(cp.asnumpy(img_reshaped_gpu)).float().cuda()
        labels_tensor = torch.tensor(labels).cuda()
        silhouette_avg = silhouette.score(img_tensor, labels_tensor)  # Chamar␣
 ↪o método estático score
        silhouettes.append(silhouette_avg)
    return cp.mean(cp.array(silhouettes))
```

[ ]:
```python
# Aplicar borramento e recalcular silhouette_score
def calculate_blurred_silhouette(images, k, color_space, blur_size):
    silhouettes = []
    for img in images:
        if color_space == 'HSV':
            img = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
        img_blurred = cv2.blur(img, (blur_size, blur_size))
        img_reshaped = img_blurred.reshape((-1, 3)).astype('float32')
        img_reshaped_gpu = cp.asarray(img_reshaped)
        kmeans = KMeans(n_clusters=k, random_state=42).fit(img_reshaped_gpu)
        labels = kmeans.labels_
        img_tensor = torch.tensor(cp.asnumpy(img_reshaped_gpu)).float().cuda()
        labels_tensor = torch.tensor(labels).cuda()
        silhouette_avg = silhouette.score(img_tensor, labels_tensor)
        silhouettes.append(silhouette_avg)
    return cp.mean(cp.array(silhouettes))
```

[ ]:
```python
# Visualização dos dados
def visualize_clusters(images, k, color_space='RGB'):
    for img in images:
        if color_space == 'HSV':
            img = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
        img_reshaped = img.reshape((-1, 3)).astype('float32')  # Converter para␣
 ↪float32
        img_reshaped_gpu = cp.asarray(img_reshaped)
        kmeans = KMeans(n_clusters=k, random_state=42).fit(img_reshaped_gpu)
        labels = kmeans.labels_
        clustered_img = labels.get().reshape(img.shape[:2])

        plt.figure(figsize=(10, 5))
        plt.subplot(1, 2, 1)
        plt.title('Original Image')
        plt.imshow(img)
        plt.axis('off')

        plt.subplot(1, 2, 2)
        plt.title('Clustered Image')
        plt.imshow(clustered_img, cmap='viridis')
        plt.axis('off')

        plt.show()
```

[ ]:
```python
# Visualizar imagens borradas
def visualize_blurred_images(images, blur_size):
    for img in images:
        img_blurred = cv2.blur(img, (blur_size, blur_size))
```

```
        plt.figure(figsize=(10, 5))
        plt.subplot(1, 2, 1)
        plt.title('Original Image')
        plt.imshow(img)
        plt.axis('off')

        plt.subplot(1, 2, 2)
        plt.title(f'Blurred Image ({blur_size}x{blur_size})')
        plt.imshow(img_blurred)
        plt.axis('off')

        plt.show()
```

[ ]:
```
# Carregar imagens
folder_path = '/home/bsfn19/PIB/ALL_IDB2/images'
images = load_images_from_folder(folder_path)
```

[ ]:
```
# Testar k=2 e k=3 para RGB
silhouette_rgb_k2 = calculate_silhouette(images, 2, 'RGB')
print("Média do Silhouette Score para k=2 e RGB: ", silhouette_rgb_k2)
silhouette_rgb_k3 = calculate_silhouette(images, 3, 'RGB')
print("Média do Silhouette Score para k=3 e RGB: ", silhouette_rgb_k3)
```

```
Média do Silhouette Score para k=2 e RGB:  0.7319810727467904
Média do Silhouette Score para k=3 e RGB:  0.6869305686308788
```

[ ]:
```
# Testar k=2 e k=3 para HSV
silhouette_hsv_k2 = calculate_silhouette(images, 2, 'HSV')
print("Média do Silhouette Score para k=2 e HSV: ", silhouette_hsv_k2)
silhouette_hsv_k3 = calculate_silhouette(images, 3, 'HSV')
print("Média do Silhouette Score para k=3 e HSV: ", silhouette_hsv_k3)
```

```
Média do Silhouette Score para k=2 e HSV:  0.7470214111300615
Média do Silhouette Score para k=3 e HSV:  0.7725789397954941
```

[ ]:
```
# Testar borramento 11x11 e 13x13 e k=2
silhouette_blur_11_rgb_k2 = calculate_blurred_silhouette(images, 2, 'RGB', 11)
print("Média do Silhouette Score para k=2 e RGB com borramento 11x11: ",␣
 ↪silhouette_blur_11_rgb_k2)
silhouette_blur_13_rgb_k3 = calculate_blurred_silhouette(images, 3, 'RGB', 13)
print("Média do Silhouette Score para k=3 e RGB com borramento 13x13: ",␣
 ↪silhouette_blur_13_rgb_k3)
```

```
Média do Silhouette Score para k=2 e RGB com borramento 11x11:
0.742155432013365
Média do Silhouette Score para k=3 e RGB com borramento 13x13:
0.6774886165673916
```

[ ]:
```
# Testar borramento 11x11 e 13x13 e k=2
silhouette_blur_11_hsv_k2 = calculate_blurred_silhouette(images, 2, 'HSV', 11)
print("Média do Silhouette Score para k=2 e HSV com borramento 11x11: ",␣
 ↪silhouette_blur_11_hsv_k2)
silhouette_blur_13_hsv_k3 = calculate_blurred_silhouette(images, 3, 'HSV', 13)
print("Média do Silhouette Score para k=3 e HSV com borramento 13x13: ",␣
 ↪silhouette_blur_13_hsv_k3)
```

```
Média do Silhouette Score para k=2 e HSV com borramento 11x11:
0.7837382827813809
Média do Silhouette Score para k=3 e HSV com borramento 13x13:
0.7025265967616668
```

[ ]:
```
#Garbage Collection
torch.cuda.empty_cache()
```