

圖像壓縮使用線性代數

圖像壓縮是數位圖像處理中的一項基本任務，旨在減少圖像的存儲大小，同時保持其視覺質量。實現這一目標的一種強大方法是通過線性代數，特別是奇異值分解（SVD）。這種技術使我們能夠以更緊湊的形式表示圖像矩陣，有效地丟棄不重要的信息，同時保留基本特徵。

以下的 Python 代碼示範了如何使用 SVD 壓縮圖像。該過程涉及將圖像分解為其組成部分，通過保留最重要特徵的子集來壓縮這些組成部分，然後重建壓縮圖像。這種方法可以應用於灰度圖像和彩色圖像，提供了一種靈活且數學上可靠的方法來減少圖像大小。

```
import numpy as np
from PIL import Image
import argparse
import os

def compress_image(image_path, compression_factor=0.1):
    # 打開圖像並將其轉換為 numpy 陣列
    img = Image.open(image_path)
    img_array = np.array(img, dtype=float)

    # 檢查圖像是灰度圖像還是彩色圖像
    if len(img_array.shape) == 2: # 灰度圖像
        # 在圖像陣列上執行 SVD
        U, S, Vt = np.linalg.svd(img_array, full_matrices=False)

        # 通過保留最顯著的奇異值來壓縮圖像
        k = int(compression_factor * min(img_array.shape))
        S_compressed = np.diag(S[:k])
        U_compressed = U[:, :k]
        Vt_compressed = Vt[:k, :]

        # 重建壓縮圖像
        img_compressed = np.dot(U_compressed, np.dot(S_compressed, Vt_compressed))
    else: # 彩色圖像
        # 分別在每個通道上執行 SVD
        img_compressed = np.zeros_like(img_array)
        for i in range(img_array.shape[2]): # 迭代每個通道
            channel = img_array[:, :, i]
            U, S, Vt = np.linalg.svd(channel, full_matrices=False)

            # 通過保留最顯著的奇異值來壓縮通道
```

```

    k = int(compression_factor * min(channel.shape))
    S_compressed = np.diag(S[:k])
    U_compressed = U[:, :k]
    Vt_compressed = Vt[:k, :]

    # 重建壓縮通道
    img_compressed[:, :, i] = np.dot(U_compressed, np.dot(S_compressed, Vt_compressed))

# 將值剪切在 0 和 255 之間，並轉換回 uint8
img_compressed = np.clip(img_compressed, 0, 255).astype(np.uint8)

# 通過在原始文件名後添加 '_compressed' 來生成輸出路徑
file_name, file_extension = os.path.splitext(image_path)
output_path = f"{file_name}_compressed{file_extension}"

# 保存壓縮圖像
compressed_img = Image.fromarray(img_compressed)
compressed_img.save(output_path)

return output_path

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description=" 使用 SVD 壓縮圖像。")
    parser.add_argument("input_file", help=" 輸入圖像文件的路徑")
    parser.add_argument("--compression_factor", type=float, default=0.1, help=" 壓縮因子（預設：0.1）")
    args = parser.parse_args()

    output_file = compress_image(args.input_file, args.compression_factor)
    print(f" 壓縮圖像已保存為：{output_file}")

```