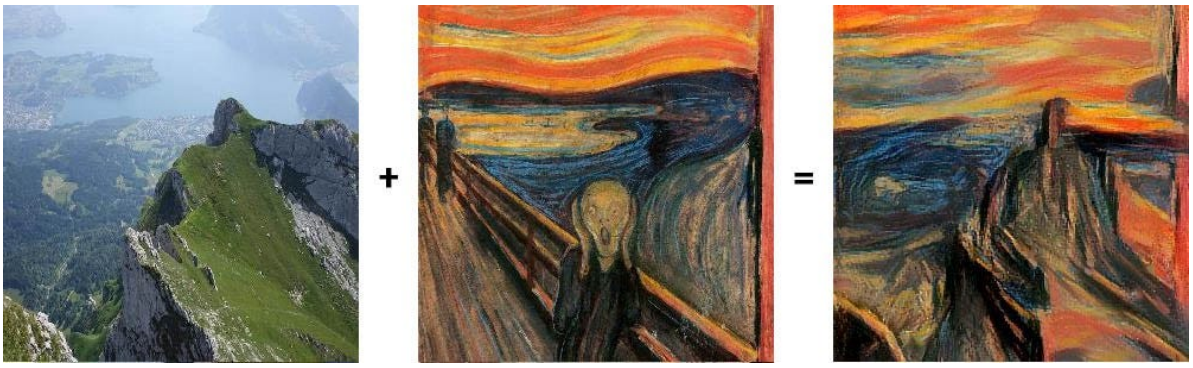
Numpy - Gram Matrix in Style transfer

Hoàng-Nguyên Vũ

1 Giới thiệu

Gram Matrix là một công cụ quan trọng trong bài toán *Style Transfer*. Nó được sử dụng để trích xuất thông tin về phong cách của một hình ảnh dựa trên ma trận đặc trưng (*feature map*) của một mạng nơ-ron.



2 Vai trò của Gram Matrix trong Style Transfer

Gram Matrix giúp nắm bắt mối quan hệ tương quan giữa các feature maps và đại diện cho cấu trúc tổng thể của hình ảnh, chẳng hạn như các mẫu và kết cấu, chứ không phải các chi tiết cục bộ. Trong Style Transfer, Gram Matrix của hình ảnh phong cách được so sánh với Gram Matrix của hình ảnh kết quả để đảm bảo rằng phong cách của hình ảnh gốc được chuyển sang hình ảnh mới.

3 Công thức Gram Matrix

Giả sử chúng ta có một ma trận đặc trưng F của một ảnh, với kích thước:

$$F \in \mathbb{R}^{C \times H \times W}$$

trong đó:

- C là số kênh (channels) của ảnh.
- H là chiều cao (height) của ảnh.
- W là chiều rộng (width) của ảnh.

Gram Matrix được tính theo công thức:

$$G = FF^T$$

trong đó:

- $G \in \mathbb{R}^{C \times C}$ là Gram Matrix.
- F được reshape về kích thước $(C, H \times W)$.

Để tránh giá trị quá lớn, Gram Matrix thường được chuẩn hóa bằng số điểm ảnh:

$$G = \frac{FF^T}{H \times W}$$

4 Cài đặt bằng NumPy

Dưới đây là cách cài đặt Gram Matrix bằng NumPy:

```python
import numpy as np

def compute_gram_matrix(feature_map: np.ndarray) -> np.ndarray:
    """
    Tính Gram Matrix từ feature map.

    Args:
        feature_map (np.ndarray): Ma trận đặc trưng có kích thước (C, H, W).

    Returns:
        np.ndarray: Gram Matrix có kích thước (C, C).
    """
    # Lấy kích thước đầu vào
    C, H, W = feature_map.shape

    # Chuyển đổi ma trận về dạng (C, H*W)
    F = feature_map.reshape(C, H * W)

    # Tính Gram Matrix G = F @ F.T
    G = np.dot(F, F.T)

    # Chuẩn hóa bằng số điểm ảnh
    G /= (H * W)

    return G

# Tạo dữ liệu giả lập với kích thước (3, 4, 4)
np.random.seed(42)
feature_map = np.random.rand(3, 4, 4)

# Tính Gram Matrix
gram_matrix = compute_gram_matrix(feature_map)
```

```

# In kết quả
print("Gram Matrix:\n", gram_matrix)
```

5 So sánh với PyTorch (Đọc thêm)

Có thể kiểm tra tính chính xác bằng cách so sánh với PyTorch:

```
import torch

feature_map_torch = torch.tensor(feature_map, dtype=torch.float32)
F_torch = feature_map_torch.view(3, -1)
gram_matrix_torch = torch.mm(F_torch, F_torch.t()) / (4 * 4)

print("\nGram Matrix PyTorch:\n", gram_matrix_torch.numpy())
```

6 Kết luận

Gram Matrix giúp mô hình Style Transfer học phong cách của một hình ảnh bằng cách đo tương quan giữa các kênh đặc trưng. Công thức này có thể được triển khai dễ dàng bằng NumPy và có thể kiểm tra với PyTorch để đảm bảo tính chính xác.

7 Bài tập

Bài tập 1: Tính Gram Matrix từ Feature Map

Mô tả bài toán

Cho một ma trận đặc trưng F có kích thước:

$$F \in \mathbb{R}^{C \times H \times W}$$

trong đó:

- C là số kênh (channels).
- H là chiều cao của ảnh.
- W là chiều rộng của ảnh.

Gram Matrix được tính theo công thức:

$$G = \frac{F \cdot F^T}{H \times W}$$

Cài đặt bằng NumPy

Dưới đây là code Python để tính Gram Matrix từ feature map:

```python
import numpy as np

def compute_gram_matrix(feature_map: np.ndarray) -> np.ndarray:
    # Your code here #

# Feature Map đầu vào cố định
feature_map = np.array([
    [[1, 2], [3, 4]],  # Kênh 1
    [[5, 6], [7, 8]],  # Kênh 2
    [[9, 10], [11, 12]]  # Kênh 3
])

# Tính Gram Matrix
gram_matrix = compute_gram_matrix(feature_map)
print(gram_matrix)
```

Kết quả mong muốn

Nếu bạn chạy đoạn code trên, kết quả đầu ra sẽ là:

$$G = \begin{bmatrix} 7.5 & 17.5 & 27.5 \\ 17.5 & 43.5 & 69.5 \\ 27.5 & 69.5 & 111.5 \end{bmatrix}$$

Bài tập 2: Đo độ tương đồng giữa hai ảnh bằng Gram Matrix

Mô tả bài toán

Cho hai ảnh với ma trận đặc trưng khác nhau, hãy đo **độ tương đồng phong cách** giữa chúng bằng Gram Matrix sử dụng công thức:

$$\text{Similarity} = \frac{\sum(G_1 \cdot G_2)}{\sqrt{\sum G_1^2} \cdot \sqrt{\sum G_2^2}}$$

trong đó:

- G_1, G_2 là Gram Matrix của hai ảnh.
- Giá trị đầu ra nằm trong khoảng [0,1], càng gần 1 thì hai ảnh càng giống nhau về phong cách.

Cài đặt bằng NumPy

```python
def compute_similarity(gram1: np.ndarray, gram2: np.ndarray) -> float:
    """
    Tính độ tương đồng giữa hai Gram Matrix.

    Args:
        gram1 (np.ndarray): Gram Matrix ảnh 1.
        gram2 (np.ndarray): Gram Matrix ảnh 2.

    Returns:
        float: Độ tương đồng trong khoảng [0,1].
    """
    # Your code here #

# Feature Map của hai ảnh
feature_map1 = np.array([
    [[1, 2], [3, 4]],
    [[5, 6], [7, 8]],
    [[9, 10], [11, 12]]
])

feature_map2 = np.array([
    [[2, 4], [6, 8]],
    [[1, 3], [5, 7]],
    [[0, 2], [4, 6]]
])

# Tính Gram Matrix của hai ảnh
gram1 = compute_gram_matrix(feature_map1)
gram2 = compute_gram_matrix(feature_map2)

# Tính độ tương đồng
similarity = compute_similarity(gram1, gram2)
print("Similarity Score:", similarity)
```

Kết quả mong muốn

Sau khi chạy đoạn code trên, kết quả đầu ra sẽ là:

$$\text{Similarity Score} = 0.67$$

Điều này cho thấy hai ảnh có phong cách khá giống nhau với độ tương đồng 67%.