Basic Python - Einsum in Numpy

Hoàng-Nguyên Vũ

1 Khái niệm `einsum` trong Python

`einsum` là một hàm trong thư viện NumPy của Python, viết tắt của "Einstein summation", giúp chúng ta thực hiện các phép tính trên ma trận một cách ngắn gọn và hiệu quả. `einsum` cho phép bạn chỉ định các phép toán trên các mảng đa chiều bằng cách sử dụng ký hiệu Einstein. Ký hiệu này cho phép bạn mô tả các phép tính như tổng, nhân chéo, và nhiều phép toán phức tạp khác một cách trực quan và ngắn gọn. Cú pháp của `einsum` là:

```
numpy.einsum(subscripts, *operands, out=None)
```

Trong đó: - `subscripts` là một chuỗi mô tả các phép toán. - `operands` là các mảng đầu vào mà bạn muốn thực hiện phép tính.

Ví dụ cơ bản

1. Nhân ma trận (Matrix multiplication)

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
C = np.einsum('ij,jk->ik', A, B)
print(C)  # Output: [[19 22] [43 50]]
```

2. Tính tổng của các phần tử trong một mảng

```
a = np.array([1, 2, 3, 4])
sum_a = np.einsum('i->', a)
print(sum_a)  # Output: 10
```

3. Nhân chéo (Outer product)

```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
outer_product = np.einsum('i,j->ij', a, b)
print(outer_product)  # Output: [[ 4  5  6] [ 8 10 12] [12 15 18]]
```

2 Bài tập thực hành

Bài tập 1: Tính tổng của các cột trong ma trận

Sử dụng `einsum` để tính tổng của từng cột trong ma trận 2D.

```
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
column_sums = np.einsum('ij->j', A)
print(column_sums)  # Output: [12 15 18]
```

Bài tập 2: Tính tích vô hướng của hai ma trận

Sử dụng `einsum` để tính tích vô hướng (dot product) của hai ma trận 2D.

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
dot_product = # Your code here #
print(dot_product)  # Output: 70
```

Bài tập 3: Tính tổng các phần tử trên đường chéo chính của ma trận

Sử dụng `einsum` để tính tổng các phần tử trên đường chéo chính của ma trận 2D.

```
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
diagonal_sum = # Your code here #
print(diagonal_sum)  # Output: 15
```

Bài tập 4: Nhân hai ma trận 2D

Sử dụng `einsum` để thực hiện phép nhân hai ma trận 2D.

```
import numpy as np

A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])
C = # Your code here #
print(C)  # Output: [[19 22] [43 50]]
```

Bài tập 5: Tính tích ngoài của hai vector

Sử dụng `einsum` để tính tích ngoài (outer product) của hai vector.

```python
import numpy as np

a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
outer_product = # Your code here #
print(outer_product)  # Output: [[ 4  5  6] [ 8 10 12] [12 15 18]]
```

Bài tập 6: Tính ma trận Gram của một tensor 3D

Sử dụng `einsum` để tính ma trận Gram của một tensor 3D có kích thước (channels, height, width).

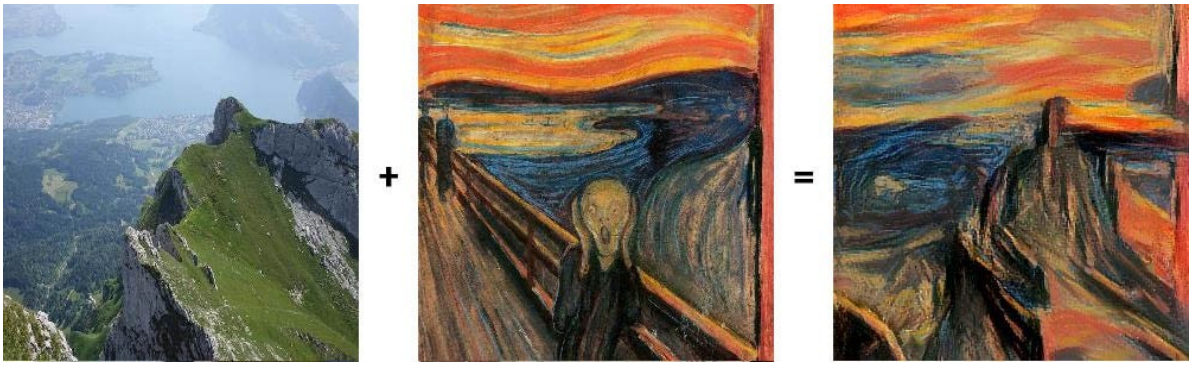
```python
import numpy as np

def gram_matrix(tensor):
    # tensor có shape (channels, height, width)
    channels, height, width = tensor.shape
    features = tensor.reshape(channels, height * width)
    gram = # Your code here #
    return gram

tensor = np.random.rand(3, 4, 4)  # ví dụ tensor ngẫu nhiên
gram = gram_matrix(tensor)
print(gram)
```

Đọc thêm về Gram matrix:

Trong lĩnh vực Style Transfer, Gram Matrix là một công cụ quan trọng để nắm bắt và đại diện cho phong cách của một hình ảnh. Gram Matrix được tính từ các đặc trưng (feature maps) của một mạng nơ-ron tích chập (Convolutional Neural Network - CNN).



Cách tính Gram Matrix

Giả sử bạn có một tensor F đại diện cho các feature maps của một hình ảnh sau khi đi qua một lớp tích chập. Tensor này có kích thước (C, H, W), trong đó:

- C là số lượng các kênh (channels).

- H là chiều cao của feature map.
- W là chiều rộng của feature map.

Gram Matrix G có kích thước (C, C) và được tính bằng công thức:

$$G_{ij} = \sum_k F_{ik} F_{jk}$$

trong đó F_{ik} và F_{jk} là các phần tử của tensor F tại kênh i và j, với k chạy qua tất cả các vị trí không gian (spatial locations).

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.