IMT3870 Bimestre 2021-3
Tarea 2

Elwin van 't Wout

August 21, 2021

Introducción

Una parte clave en una red neuronal convolucional es la aplicación de un filtro en una capa convolucional. Esta operación de una convolución se explica mejor a traves de ejemplos.

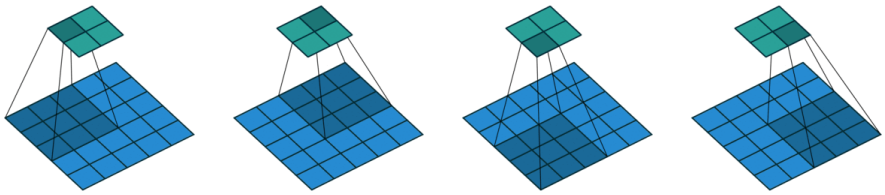

Figure 1: Operación de convolución sobre una matriz de 5×5 (celeste) con un filtro de 3×3 (azul) y un stride de 2, que genera un output de 2×2 (verde).

En el ejemplo dibujado en Figura 1 se inicia con una matriz de 5×5. Se aplica un filtro de 3×3, es decir se reduce una parte de 3×3 del input a un único número como output. Esta parte de 3×3 se va moviendo sobre el input con pasos. El stride es el paso que toma el filtro sobre el input. Es decir, con un stride de dos, se mueve el filtro dos elementos horizontalmente o verticalmente. Como otro ejemplo, consideren la matriz dado por

$$\begin{bmatrix} 5 & 0 & 3 & 3 & 7 & 9 & 3 \\ 5 & 2 & 4 & 7 & 6 & 8 & 8 \\ 1 & 6 & 7 & 7 & 8 & 1 & 5 \\ 9 & 8 & 9 & 4 & 3 & 0 & 3 \\ 5 & 0 & 2 & 3 & 8 & 1 & 3 \\ 3 & 3 & 7 & 0 & 1 & 9 & 9 \\ 0 & 4 & 7 & 3 & 2 & 7 & 2 \end{bmatrix}$$

la cual tiene un tamaño 7×7. Se aplica el filtro dado por

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

lo cual tiene un tamaño 3×3. Aplicando el filtro a la parte superior izquierda resulta en

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \star \begin{bmatrix} 5 & 0 & 3 \\ 5 & 2 & 4 \\ 1 & 6 & 7 \end{bmatrix} =$$
$$(1 \cdot 5 + 2 \cdot 0 + 1 \cdot 3 + 2 \cdot 5 + 4 \cdot 2 + 2 \cdot 4 + 1 \cdot 1 + 2 \cdot 6 + 1 \cdot 7)/16 = 3.375$$

donde se multiplican cada elemento de la matriz con cada elemento del filtro que está en el mismo lugar, se suman, y dividen por la suma de los elementos del filtro (16) para obtener el promedio ponderado. Ahora, si tenemos un stride de dos, el segundo elemento del output será

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \star \begin{bmatrix} 3 & 3 & 7 \\ 4 & 7 & 6 \\ 7 & 7 & 8 \end{bmatrix} =$$
$$(1 \cdot 3 + 2 \cdot 3 + 1 \cdot 7 + 2 \cdot 4 + 4 \cdot 7 + 2 \cdot 6 + 1 \cdot 7 + 2 \cdot 7 + 1 \cdot 8)/16 = 5.8125,$$

moviendo hacia derecha, o

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \star \begin{bmatrix} 1 & 6 & 7 \\ 9 & 8 & 9 \\ 5 & 0 & 2 \end{bmatrix} =$$
$$(1 \cdot 1 + 2 \cdot 6 + 1 \cdot 7 + 2 \cdot 9 + 4 \cdot 8 + 2 \cdot 9 + 1 \cdot 5 + 2 \cdot 0 + 1 \cdot 2)/16 = 5.9375$$

hacia abajo. Al final, el resultado es

$$\begin{bmatrix} 3.375 & 5.8125 & 6.4375 \\ 5.9375 & 5.3125 & 2.5 \\ 3.375 & 2.9375 & 5.4375 \end{bmatrix}$$

una matriz de 3×3. Nótense que la operación de convolución como acá es un especie de promedio ponderado, entonces el resultado tiene valores en el mismo rango del input (entre 0 y 9 en este ejemplo).

Tarea

Este tarea contempla una implementación de Numba para la operación de convolución.

1. Vamos a utilizar dos imágenes disponible en `sklearn`. Corren los comandos siguientes.

   (a) `from sklearn import datasets`
   (b) `dataset = datasets.load_sample_images()`
   (c) `china, flower = dataset.images`

2. Analizen el formato de `china` y `flower`. ¿Cuales son las dimensiones de los arreglos? ¿Los elementos son números enteros o reales?

3. Dibujen las imágenes con el comando `imshow()` de `matplotlib`.

4. Programen la operación de convolución.
   (a) Definen un filtro 2D (debe ser a menos 5×5).
   (b) Definen un valor para el paso (*stride*) (debe ser entre 2 y 5).
   (c) Programen una función que toma como input el imágen, el filtro y el stride. El output es una nueva imagen generado por la operación de convolución. El filtro se aplica de forma separada a cada canal de color (RGB, rojo-verde-azul).
   (d) Dibujen el imagen filtrado.

5. Programen la operación de convolución (paso 4(c) arriba) en los paradigmas siguientes.
   (a) Con bucles (for-loops) de Python y sin operaciones de cálculo por `numpy` o `scipy`. (Se puede utilizar `np.zeros()` para crear una matriz, pero no se puede utilizar comandos tales como `np.dot()`, `np.sum()`, etc.).
   (b) Con acceleración de Numba.
   (c) Con paralelización en Numba por medio de `prange()`.

6. Comparen el tiempo de cómputo entre los tres paradigmas. Además, eligen distintos números de hilos para Numba (a menos 1, 2, 3, y 4).

7. Expliquen las diferencias en tiempo de cómputo en un informe separado o adentro el Jupyter Notebook en una celda de *markdown*.
   (a) En la respuesta, incluye el número de núcleos físicos que tiene su computador.
   (b) Expliquen cuales de los bucles en la operación paralelizaron con Numba y por qué.

8. Opcionalmente, carga otra imagen o fotografía y aplica el filtro (pueden usar `plt.imread()`).

Evaluación

Entreguen todo el código de Python (script o Jupyter notebook) y el informe (pdf o Jupyter notebook) en Canvas.

Los reglamientos del curso se puede encontrar en Canvas. Se destaca que las tareas deben ser hechas de forma individual.

Sugerencias

Filtros comúnes son líneas verticales $\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, líneas horizontales $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, y filtros completos $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$.

En esta tarea no es necesario usar *padding*.

Uno de los errores más comunes a la hora de programar Numba es utilizar un tipo equivocado. Como por ejemplo, utilizar un *integer* en lugar de un *float*, o un *list* en lugar de un *tuple*. Numba analiza el código para detectar el tipo de datos, con el objetivo de crear código de bajo nivel. Sin embargo, la detección automática del tipo de dato no es siempre posible. En este caso, el programador debe especificar el tipo de datos.

A la hora de crear una matriz en Numba, por ejemplo, `np.zeros(size, dtype=np.float_)`, asegúrense que `size` es un *tuple*, no una *list*.

La tarea se puede preparar en un computador personal o en Google Colab (`https://colab.research.google.com`). Para este tarea se requiere una versión reciente de Python y las bibliotecas `numpy`, `scipy`, `sklearn` y `numba`.

Se recomienda crear un nuevo *environment* de Anaconda para esta tarea con el objetivo de evitar problemas de distintas versiones the Python y bibliotecas. Ver, por ejemplo, `https://towardsdatascience.com/a-guide-to-conda-environments-bc6180fc533`, `https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html`, o `https://docs.anaconda.com/anaconda/navigator/tutorials/manage-environments/`.