DEEP LEARNING

Mạng nơ ron tích chập (CNN)

Phạm Nguyên Khang
pnkhang@cit.ctu.edu.vn

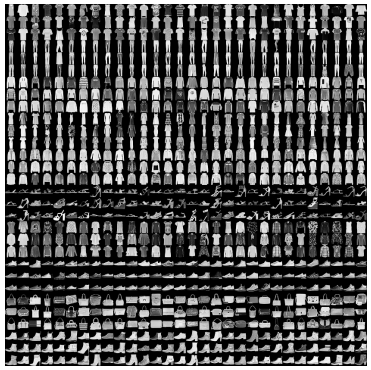
CAN THO, 22/12/2022

Nội dung

- Phân lớp ảnh với mạng nơ ron
- Nhược điểm của mạng nơ ron truyền thẳng (FNN) trong xử lý ảnh
- Giới thiệu CNN
- Kiến trúc mạng CNN
- Phân lớp ảnh MNIST với CNN

Phân lớp ảnh với mạng nơ ron

- Phân lớp ảnh MNIST Fashion (https://www.tensorflow.org/tutorials/keras/classification)
  - 70.000 ảnh xám
  - 10 lớp



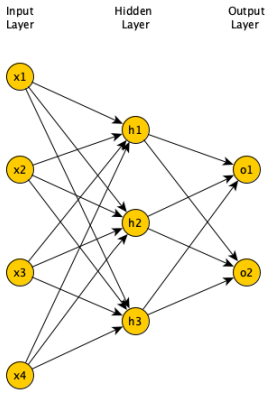
Phân lớp ảnh với mạng nơ ron

- Phân lớp ảnh MNIST Fashion (https://www.tensorflow.org/tutorials/keras/classification)

```
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(10)
])
```

Nhược điểm của mạng nơ ron truyền thẳng trong xử lý ảnh

- Trong mạng FNN đầy đủ (Dense), mỗi nút ở tầng t sẽ kết nối với tất cả các nút ở tầng t+1.
  - Vd:
    - Input: ảnh xám 64x64x1 = 4096
    - Tầng ẩn 1: 500 nút => 4.096 x 500 = 2.048.000 trọng số!
- Trải dài ảnh 2D (flatten) => mất thông tin không gian



Giới thiệu mạng CNN

- Năm 1959, Hubel và Wiesel thực hiện các thí nghiệm để hiểu các thần kinh thị giác của bộ não xử lý các thông tin hình ảnh như thế nào.
  - Chiếu ánh sáng phía trước 1 con mèo và quan sát phản ứng của các tế bào thần kinh thị giác
    - Một số tế bào phản ứng mạnh khi ánh sáng chiếu theo một góc nào đó => tế bào đơn (simple cells)
    - Một số tế bào khác phản ứng mạnh khi ánh sáng chiếu nhưng không quan tâm tới góc chiếu và dường như khi có sự chuyển động của ánh sáng => tế bào phức (complex cells)
    - Dường các dự tế bào phức nhận tín hiệu từ các tế bào đơn và nên một cấu trúc phân cấp
- Nhận giải Nobel vào năm 1981 vì phát hiện này

Giới thiệu mạng CNN

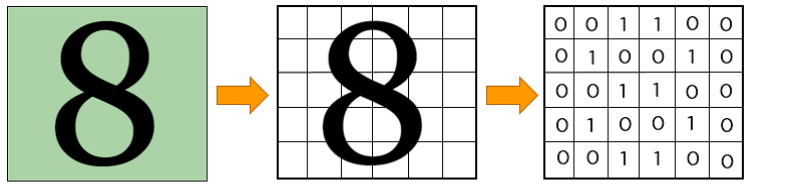
- Năm 1980, dựa trên đề xuất về cấu trúc phân cấp của các tế bào, Fukushima đề xuất Neocognitron dùng để nhận dạng ký tự Nhật viết tay. Có thể nói Neocognitron là mạng CNN đầu tiên.
- Năm 1989, Yann Lecun đề xuất CNN có thể được huấn luyện bằng thuật toán lan truyền ngược. CNN vượt xa các mô hình khác tại kỳ thi ILSVRC (ImageNet Large Scale Visual Recognition Challenge)
- Các mô hình CNN thắng tại ILSVRC
  - AlexNet (2012), ZFNet (2013), GoogLeNet và VGG (2014), ResNet (2015).

Kiến trúc mạng CNN

- Mạng CNN tiêu biểu gồm có 4 tầng
  - Đầu vào (Input)
  - Tích chập (Convolution)
  - Giảm kích thước (Pooling)
  - Kết nối đầy đủ (Fully connected)

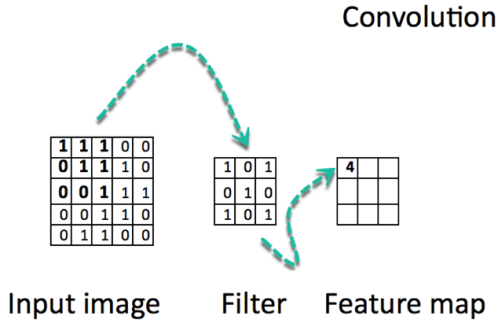
Tầng đầu vào

- Ảnh 2D xám hoặc màu
  - Không làm phẳng thành 1D để giữ lại thông tin không gian



Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)

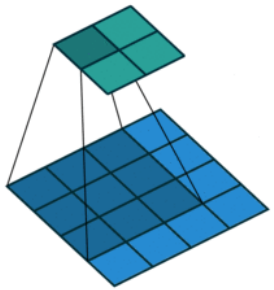


Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - Kích thước mặt nạ (Filter size)
  - Mở rộng đầu vào (Padding)
  - Bước nhảy (Stride)
  - Co (Dilation)
  - Hàm kích hoạt (Activation function)

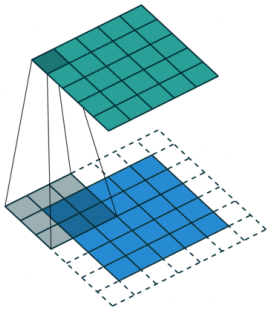
Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - **Kích thước mặt nạ (Filter size)**
    - Ảnh 4x4 chập bằng mặt nạ 3x3 => ảnh kết quả 2x2



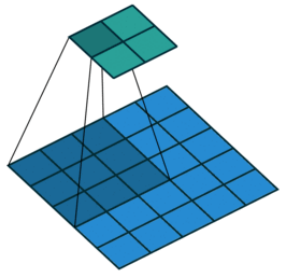
Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - **Mở rộng đầu vào (Padding)**
    - Ảnh 5x5 chập bằng mặt nạ 3x3, mở rộng 1 pixel =>kết quả 5x5



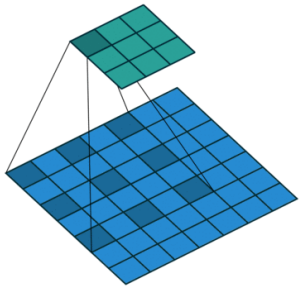
Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - **Bước nhảy (Stride)**
    - 5x5 chập bằng 3x3, bước nhảy 2 => kết quả 2x2



Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - **Co (Dilation)**
    - Ảnh 7x7 chập bằng 3x3, co 2 => kết quả 3x3



Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - **Hàm kích hoạt (Activation function)**
    - Thông thường là ReLU hoặc biến thể của nó

0	1	-4
0	2	0
-1	4	3

Filter output

—>

0	1	0
0	2	0
0	4	3

Filter output after ReLU

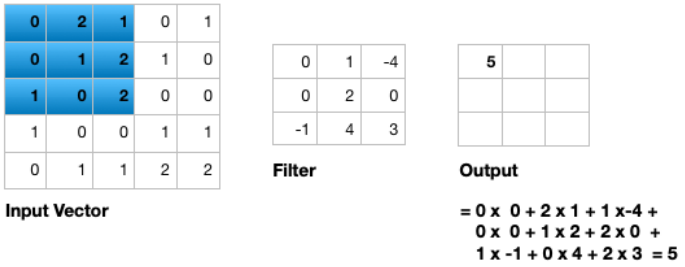
Tầng tích chập

- Phép tích chập (xem lại trong học phần Xử lý ảnh)
  - Kích thước đầu ra

$$\frac{input\ size\ -\left(filter\ size + (filter\ size\ -1) * (dilation - 1)\right) + 2 * padding}{stride} + 1$$

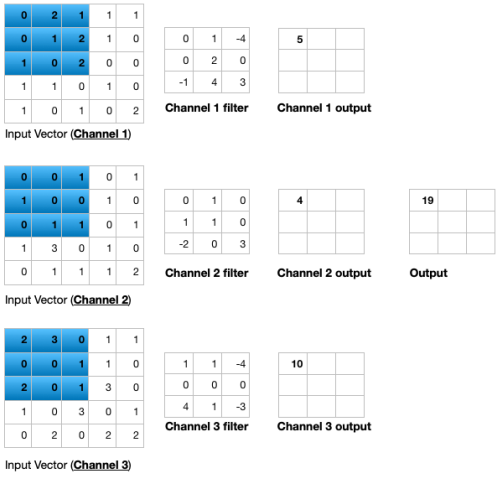
Tầng tích chập

- Phép tích chập 2D trên đầu vào chỉ có 1 kênh
  - Có thể cộng thêm bias vài tổng



Tầng tích chập

- Phép tích chập 2D trên đầu vào có nhiều kênh
  - Mỗi kênh có 1 mặt nạ khác nhau
  - Có thể cộng thêm bias vài tổng sau cùng

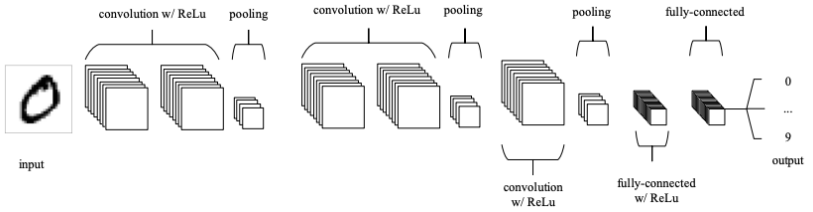


Tầng giảm kích thước (pooling)

- Thực hiện lấy mẫu giảm (giảm kích thước đầu vào), làm giảm số lượng tham số của mô hình, giảm overfitting
  - Lớn nhất: Max Pooling
  - Trung bình: Average Pooling

Tầng kết nối đầy đủ

- Liên kết đầu ra của tất cả các nơ ron từ tầng trước đó đến các nơ ron của tầng này, thực hiện phân lớp
  - Thông thường trước, khi đến tầng này, ta thêm tầng làm phẳng (Flatten).



Xây dựng mô hình CNN với Keras

- Lớp Conv2D

```
tf.keras.layers.Conv2D(filters,
        kernel_size,
        strides=(1, 1),
        padding="valid",
        data_format=None,
        dilation_rate=(1, 1),
        groups=1,
        activation=None,
        use_bias=True,
        …
)
```


Xây dựng mô hình CNN với Keras

- Lớp Conv2D
  - Dữ liệu đầu vào
    - 4+D tensor: batch_shape + (channels, rows, cols) if data_format='channels_first'
    - 4+D tensor: batch_shape + (rows, cols, channels) if data_format='channels_last'.
  - Dữ liệu đầu ra
    - 4+D tensor: batch_shape + (filters, new_rows, new_cols) if data_format='channels_first'
    - 4+D tensor: batch_shape + (new_rows, new_cols, filters) if data_format='channels_last'.
  - Số lượng tham số = (số channels*filter size + bias)*số filters

Xây dựng mô hình CNN với Keras

- Lớp Conv2D

```
# Đầu vào là ảnh màu RGB kích thước 28x28 với định dạng
dữ liệu `channels_last` và batch size 4.

input_shape = (4, 28, 28, 3)
x = tf.random.normal(input_shape)
y = tf.keras.layers.Conv2D(2, 3, activation='relu',
                           input_shape=input_shape[1:])(x)
print(y.shape)

Kết quả: (4, 26, 26, 2)
```

Xây dựng mô hình CNN với Keras

- Lớp MaxPooling2D

```
tf.keras.layers.MaxPooling2D(
        pool_size=(2, 2),
        strides=None,
        padding="valid",
        data_format=None,
        **kwargs
)
```

- Số lượng tham số = 0

Xây dựng mô hình CNN với Keras

- Lớp MaxPooling2D

Padding="valid"
output_shape = math.floor((input_shape - pool_size) / strides) + 1 (khi input_shape >= pool_size)

Padding="same"
output_shape = math.floor((input_shape - 1) / strides) + 1

Xây dựng mô hình CNN với Keras

- Lớp Flatten

```
tf.keras.layers.Flatten(
        data_format=None,
        **kwargs
)
```

- Số lượng tham số = 0

Xây dựng mô hình CNN với Keras

- Lớp Dense

```
tf.keras.layers.Dense(
   units,
   activation=None,
   use_bias=True,
   kernel_initializer="glorot_uniform",
   …
)
```

- Số lượng tham số = số đầu vào * số units + bias

Thực hành 1

- Xây dựng mô hình phân lớp tập dữ liệu ảnh chữ số viết tay MNIST: https://keras.io/examples/vision/mnist_convnet/

Thực hành 2

- Thu thập một tập dữ liệu ảnh màu (tối thiểu 3 lớp, tối thiểu 500 ảnh/lớp)
- Xây dựng một mô hình CNN để phân lớp tập dữ liệu này
  - Thực nghiệm với nhiều kiến trúc mạng khác nhau
  - So sánh độ chính xác trên tập test giữa các kiến trúc mạng khác nhau
  - Soạn slides để báo cáo

THANK YOU