객체지향 프로그래밍 [CSED232-01]

Assignment #2

Due date: ~ 10.30(wed.) 23:59

Objective

본 과제에서는 Convolution과 padding을 class 상속을 이용해 구현한다.

Background – Convolution and padding

Convolution은 이미지 및 행렬 연산에 가장 많이 쓰이는 처리 기법으로, input matrix의 각 patch를 kernel matrix로 원소별 곱셈을 한 후 합을 해서 새로운 행렬을 만들어 내는 과정이다. 이 process는 image blurring 또는 image edge extraction에 사용되며 최근 deep learning에선 이 kernel matrix를 학습 시키는 convolution neural network에서도 사용된다.

Input matrix와 kernel matrix의 (x, y) 원소를 각각 $\mathrm{input}[\mathrm{x},\mathrm{y}], \mathrm{kernel}[\mathrm{x},\mathrm{y}]$라 하고 kernel matrix의 크기는 $\mathrm{k} \times \mathrm{k}$라 하자. 이때 k는 홀수이다. 그러면 출력 matrix의 (x′, y′)인 $\mathrm{output}[\mathrm{x}',\mathrm{y}']$는 다음과 같이 정의된다.

$$\mathrm{output}[\mathrm{x}',\mathrm{y}'] = \sum_{i=0}^{k}\sum_{j=0}^{k} \mathrm{input}[x'+i, y'+j] * \mathrm{kernel}[i,j]$$

예를 들어서 input과 kernel이 아래와 같다고 하자.

$$\mathrm{input} = \begin{bmatrix} 3 & 3 & 2 & 1 \\ 0 & 0 & 1 & 3 \\ 3 & 1 & 2 & 2 \\ 2 & 0 & 0 & 2 \end{bmatrix}, \mathrm{kernel} = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$

그러면 output은 아래와 같다.

$$\mathrm{output} = \begin{bmatrix} 12 & 12 \\ 10 & 17 \end{bmatrix}$$

$\mathrm{output}[0,0] = 12$ 인 이유는 $\mathrm{output}[0,0] = 3*0 + 3*1 + 2*2 + 0*2 + 0*2 + 1*0 + 3*0 + 1*1 + 2*2$이기 때문이다. 다음 링크의 첫번째 이미지를 보면 convolution에 대해 이해를 돕는 그림이 있다.

위 예시에서도 보면 알 수 있지만 convolution의 단점은 input matrix와 output matrix의 크기가 다르다는 것이다. Input matrix의 크기가 $\mathrm{h} \times \mathrm{w}$, kernel matrix의 크기가 $\mathrm{k} \times \mathrm{k}$ 라고 하면 output matrix의 크기는 $(\mathrm{h}-\mathrm{k}+1) \times (\mathrm{w}-\mathrm{k}+1)$이다.

이 단점을 보안해 주는 것이 바로 padding이다. Padding은 convolution의 전처리작업으로 convolution을 하기 전에 input matrix의 가장자리에 추가로 값을 붙여서 convolution을 하고 난

이후에도 input matrix와 output matrix의 크기를 같게 한다.

위 과제에서는 3가지 padding을 구현한다. 이 padding은 tensorflow의 tf.pad 함수에서 가져왔다.

Zero padding

첫 번째 padding은 가장 기본적인 zero padding이다. Zero padding은 input matrix의 가장자리에 0을 붙이는 padding이다. 이전에 본 input matrix에 두께가 1인 zero padding을 하면 다음과 같이 된다. padding으로 추가된 값들은 빨간색으로 적었다.

$$\text{padded input} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 & 3 & 0 \\ 0 & 3 & 1 & 2 & 2 & 0 \\ 0 & 2 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

두께가 2인 padding을 하면 다음과 같이 된다.

$$\text{padded input} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 3 & 0 & 0 \\ 0 & 0 & 3 & 1 & 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

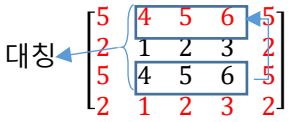
이렇게 padding을 해 주면 matrix의 크기가 커지지 때문에 padding을 해준 input matrix에 convolution을 하면 input matrix와 output matrix의 크기가 같아지게 된다. Input matrix와 output matrix의 크기가 같아지려면 kernel matrix의 크기는 $k \times k$라 할 때 zero padding의 두께는 $(k-1)/2$가 되어야 한다.

Reflect padding

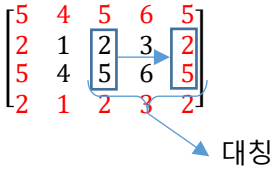
두 번째 padding은 reflect padding이다. Reflect padding은 padding을 할 때 input matrix의 값을 복제 및 뒤집기를 해서 만든다. 이때 input matrix의 가장자리에 있는 원소들은 사용되지 않는다. Input matrix와 두께가 1인 reflected padding을 한 input matrix는 다음과 같다.

$$\text{input} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \text{padded input} = \begin{bmatrix} 5 & 4 & 5 & 6 & 5 \\ 2 & 1 & 2 & 3 & 2 \\ 5 & 4 & 5 & 6 & 5 \\ 2 & 1 & 2 & 3 & 2 \end{bmatrix}$$

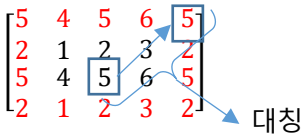
위 예시에서 위쪽 padding인 padded input의 첫번째 row 중간 부분은 입력 행렬의 두번째 row (padded input의 세번째 row)를 대칭으로 복사해서 붙여 넣은 것이다. 이렇게 하면 padded input의 두번째 row를 중심으로 대칭 구조를 이루게 된다. 아래쪽 padding인 padded input의 네번째 row도 같은 식으로 한다.



같은 식으로 Padded input의 오른쪽 padding인 다섯 번째 column의 중간 부분은 입력 행렬의 두 번째 column (padded input의 세번째 column)을 복사해서 붙여 넣은 것이다. 이렇게 하면 padded input의 네번째 column을 중심으로 대칭을 이루게 된다. 왼쪽 padding인 첫번째 column 도 동일하게 한다.



Padded input의 코너 부분 원소는 input의 코너를 중심으로 대칭점을 찾아서 구한다. 예를 들어 padded input의 row 1, column 5의 원소는 input matrix의 오른쪽 상단 코너인 row 1, column 3 (padded input의 row2, column 4)를 중심으로 대칭점인 input matrix의 row 2, column 2 (padded input의 row3, column 3) 의 원소와 같게 된다.



Input matrix의 크기가 $\mathrm{h} \times \mathrm{w}$ 라 하면 세로 padding의 두께는 최대 $\mathrm{h}-1$까지만 되고 가로 padding의 두께는 최대 $\mathrm{w}-1$까지만 된다. Input matrix와 output matrix의 크기가 같아지게 하는 padding의 두께는 $(\mathrm{k}-1)/2$ 이므로 $\frac{\mathrm{k}-1}{2} \leq \mathrm{w}-1$, $\frac{\mathrm{k}-1}{2} \leq \mathrm{h}-1$ 이어야 padding을 해 줄 수 있다.

Symmetric padding

세 번째 padding은 symmetric padding이다. 이는 reflect padding처럼 위, 아래, 양 옆에 padding 을 할 때 input matrix의 값을 복제 및 뒤집기를 해서 만드는데 input matrix의 가장자리에 잇는 원소들도 사용된다. Input matrix와 두께가 2인 symmetric padding을 한 input matrix는 다음과 같다.

$$\mathrm{input} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \mathrm{padded\ input} = \begin{bmatrix} 5 & 4 & 4 & 5 & 6 & 6 & 5 \\ 2 & 1 & 1 & 2 & 3 & 3 & 2 \\ 2 & 1 & 1 & 2 & 3 & 3 & 2 \\ 5 & 4 & 4 & 5 & 6 & 6 & 5 \\ 5 & 4 & 4 & 5 & 6 & 6 & 5 \\ 2 & 1 & 1 & 2 & 3 & 3 & 2 \end{bmatrix}$$

Symmetric padding은 input matrix의 가장자리 행렬도 padding에 사용된다는 점만 reflect padding과 다르지 padding을 만드는 방식은 reflect padding과 비슷하게 input matrix를 복사 및 대칭 시켜서 만들면 된다.

Input matrix의 크기가 $h \times w$ 라 하면 세로 padding의 두께는 최대 h까지만 되고 가로 padding의 두께는 최대 w까지만 된다. 따라서 Input matrix와 output matrix의 크기가 같아지게 하는 padding의 두께는 $(k-1)/2$ 이므로 $\frac{k-1}{2} \leq w$, $\frac{k-1}{2} \leq h$ 이어야 padding을 해 줄 수 있다

Program Explanation

이 과제에서는 padding없이 convolution을 하는 class 1개와 이 class를 상속 받아 padding을 하면서 convolution하는 class 3개를 만들어야 한다. Padding 없이 convolution 하는 class는 kernel matrix와 크기를 입력 받아 저장하는 함수와 input matrix를 받고 미리 저장해둔 kernel matrix로 convolution해서 결과를 보여주는 함수를 만들어야 한다.

상속받는 class 3개는 input matrix를 받고 이를 padding 한 결과와 padding한 input matrix를 미리 저장해둔 kernel matrix로 convolution해서 만든 결과를 보여줘야한다. 각 클래스는 zero padding, reflect padding, symmetric padding을 한다.

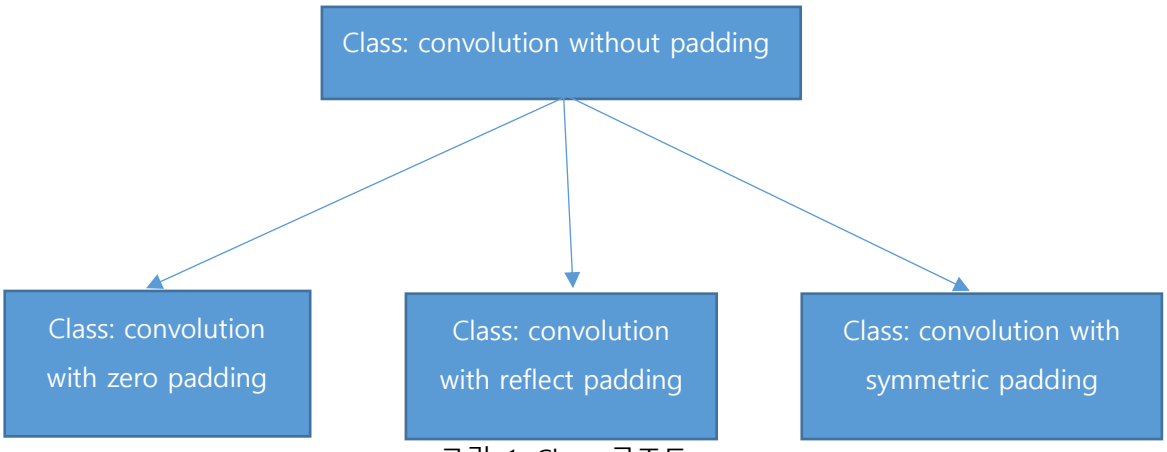


그림 1. Class 구조도

프로그램을 실행하면 kernel matrix와 input matrix를 보여주고, padding 없는 convolution 결과, 3 가지 padding을 한 input matrix와 이를 가지고 convolution한 결과를 보여준다.

또한 한 개의 Kernel matrix로 만든 결과들을 보여주고 난 뒤엔 크기가 다른 새로운 kernel matrix 로 만든 결과들도 보여준다. Input matrix는 같아도 되고 달라도 된다.

프로그램 예시

그림 2. Kernel, input, result matrix 출력 결과

위 결과물은 3 x 3 kernel matrix와 input matrix, 그리고 padding 없이 convolution한 결과물을 보여주고 있다. 코드를 실행하면 이런 식으로 어떤 kernel matrix와 input matrix를 썼는지, 그리고 그 convolution 결과물은 무엇인지를 보여줘야한다.

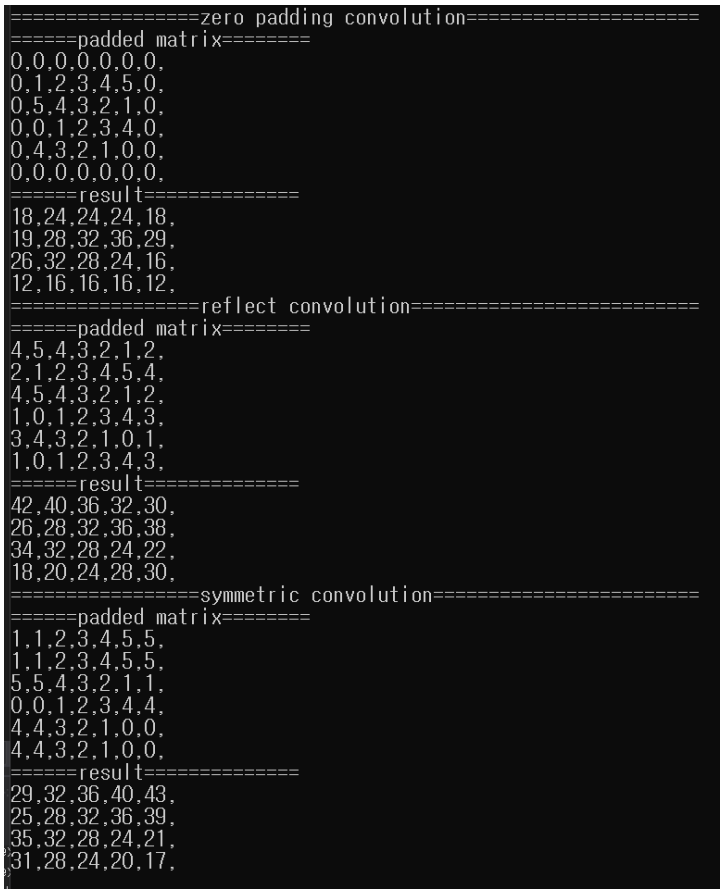


그림 3. Padded input matrix와 result matrix 출력 결과

위 결과물은 각 padding을 한 입력과 convolution의 결과물이다. 위에서부터 zero padding, reflect padding, symmetric padding의 결과물이다.

그림 4. 다른 크기의 input matrix와 kernel matrix의 결과

3x3 kernel matrix로 결과를 보여주고 나면 다른 크기의 kernel matrix (여기선 5 x 5)로 만든 결과를 똑같이 보여준다.

Requirements

- 본 과제의 프로그램은 리눅스 환경(프로그래밍 실습 서버 또는 MinGW 또는 맥에서의 터미널 등 그와 유사한 환경) 또는 윈도우 환경 (visual studio 2019)에서 컴파일 및 구동 가능하도록 작성한다.
- 리눅스 환경에서 컴파일 할 시 반드시 Makefile을 작성하여 컴파일이 가능하도록 해야 하며, 작성된 Makefile 역시 제출해야 한다. (make를 통한 컴파일 실패 시, 컴파일이 안되는 것으로 간주)
- 실습 서버의 접속 방법은 첨부된 [프로그래밍 실습환경 사용가이드]의 '실습 시스템 접속 방법'을 참조한다.
- 상기한 환경에서의 프로그램 컴파일 및 실행 과정에 대한 설명과 이미지 모두 보고서에 포함해야 한다
- 상속(Inheritance)을 반드시 사용해야 한다.
- Kernel matrix의 크기를 $k \times k$라고 했을 때 k는 1 이상의 홀수 이어야 한다.
- Input matrix의 크기는 10×10을 넘지 않도록 한다.
- Kernel matrix의 높이가 input matrix의 높이보다 크거나 Kernel matrix의 너비가 input matrix의 너비보다 큰 경우는 고려하지 않는다.
- 보고서 및 소스코드는 LMS의 과제 디렉토리에 업로드한다.
- 채점 기준은 AssnReadMe_Fall_2019.pdf 파일을 참고한다.