

핸즈온 머신러닝

PART1. 머신러닝

- 머신러닝이란? 데이터에서부터 학습하도록 컴퓨터를 프로그래밍하는 과학

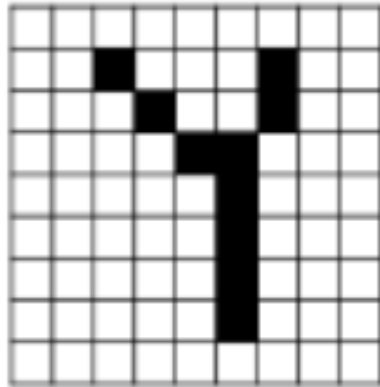
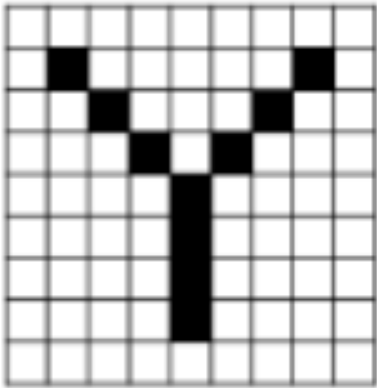
ex) 스팸 필터: 전통적인 프로그래밍 방식의 스팸 필터는 스팸 메일 발송자가 스팸 필터에 대항해 계속 단어를 바꾸면 영원히 새로운 규칙을 추가해야 한다. 그러나 머신러닝 기반의 스팸 필터는 스팸으로 지정한 메일에 특정 단어가 자주 나타는 것을 자동으로 인식하고 별도의 작업을 하지 않아도 자동으로 이 단어를 스팸으로 분류한다.

- 데이터마이닝: 머신러닝을 통해 배울 수도 있다. 머신러닝 기술을 적용해서 대용량의 데이터를 분석하면 겉으로 보이지 않는 패턴을 발견할 수 있다.
- 애플리케이션 사례
 1. 생산 라인에서 제품 이미지를 분석해 자동으로 분류하기: 이미지 분류 작업.
일반적으로 합성곱 신경망(CNN)을 사용하여 수행한다.

핸즈온 머신러닝

PART1. 머신러닝

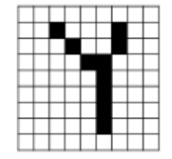
- 합성곱 신경망(CNN: Convolutional Neural Network)은 이미지 처리에 탁월한 성능을 보이는 신경망이다. 이미지 처리를 하기 위해서 다층 퍼셉트론을 사용할 수는 있지만 한계가 있었다. 예를 들어, 알파벳 손글씨를 분류하는 어떤 문제가 있다고 해보면 아래의 그림은 알파벳 Y를 비교적 정자로 쓴 손글씨와 다소 휘갈겨 쓴 손글씨 두 개를 2차원 텐서인 행렬로 표현한 것이다.



기계가 보기에 는 각 픽셀마다 가진 값이 거의 상이하므로 완전히 다른 값을 가진 입력이다. 그런데 이미지라는 것은 위와 같이 같은 대상이라도 휘어지거나, 이동되었거나, 방향이 뒤틀렸거나 등 다양한 변형이 존재한다. 다층 퍼셉트론은 몇 가지 픽셀만 값이 달라져도 민감하게 예측에 영향을 받는다는 단점이 있다.

핸즈온 머신러닝

PART1. 머신러닝



↓ 변환

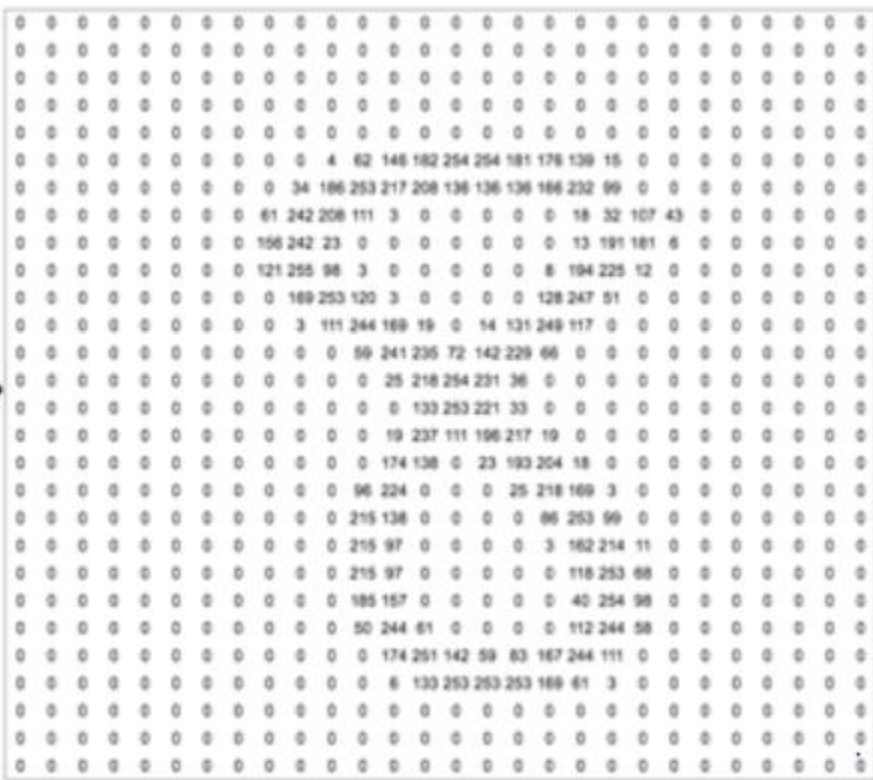


- 손글씨를 다층 퍼셉트론으로 분류한다고 하면, 이미지를 1차원 텐서인 벡터로 변환하고 다층 퍼셉트론의 입력층으로 사용해야 한다. 1차원으로 변환된 결과는 사람이 보기에 이가 원래 어떤 이미지였는지 알아보기가 어렵다. 이는 기계도 마찬가지이다. 위와 같이 결과는 변환 전에 가지고 있던 공간적인 구조(spatial structure) 정보가 유실된 상태이다.
- 여기서 공간적인 구조 정보라는 것은 거리가 가까운 어떤 픽셀들끼리는 어떤 연관이 있고, 어떤 픽셀들끼리는 값이 비슷하거나 등을 포함하고 있다. 결국 이미지의 공간적인 구조 정보를 보존하면서 학습할 수 있는 방법이 필요해졌고, 이를 위해 사용하는 것이 합성곱 신경망입니다.
- 이미지 처리의 기본적인 용어: 기계는 글자나 이미지보다 숫자. 다시 말해, 텐서를 더 잘 처리할 수 있습니다. 이미지는 **(높이, 너비, 채널)**이라는 3차원 텐서입니다. 여기서 높이는 이미지의 세로 방향 픽셀 수, 너비는 이미지의 가로 방향 픽셀 수, 채널은 색 성분을 의미합니다.

PART1. 머신러닝

- 8

28 x 28
784 pixels



핸즈온 머신러닝

PART1. 머신러닝

- 합성곱 신경망(CNN: Convolutional Neural Network) 이란?

2) CNN은 필터링 기법을 인공신경망에 적용함으로써 이미지를 더욱 효과적으로 처리하기 위해 처음 소개되었다. 현재는 딥러닝에서 이용되고 있는 형태의 CNN이 제안되었다. 기존의 필터링 기법이 앞의 그림과 같이 고정된 필터를 이용하여 이미지를 처리했다. CNN의 기본 개념은 "행렬로 표현된 필터의 각 요소가 데이터 처리에 적합하도록 자동으로 학습되게 하자"이다.

EX) 이미지 분류 알고리즘을 개발하고자 할 때 우리는 필터링 기법을 이용하여 분류 정확도를 향상시킬 수 있다. 사람의 직관이나 반복적인 실험을 통해 알고리즘에 이용될 필터를 결정해야 하는 문제점을 CNN을 통해 자동으로 이미지 분류 정확도를 최대화 하는 필터를 학습할 수 있다.

<https://untitledtblog.tistory.com/150>

합성곱 신경망은 크게 **합성곱층과(Convolution layer)**와 **풀링층(Pooling layer)**으로 구성된다.

핸즈온 머신러닝

PART1. 머신러닝







- 합성곱 신경망(CNN: Convolutional Neural Network) 은)은 이미지 처리에 탁월한 성능을 보이는 신경망. 합성곱 신경망으로도 자연어 처리를 할 수 있습니다.

1) 필터링은 이미지 처리 분야에서 광범위하게 이용되는 기법으로써, 이미지에서 테두리 부분을 추출하거나 이미지를 흐릿하게 만드는 등의 기능을 수행하기 위해 이용된다. 필터링은 행렬의 형태로 표현된 이미지에 대해 행렬로 표현된 필터를 동일하게 적용함으로써 수행된다.

$$G_{ij} = (F * X)(i, j) = \sum_{m=0}^{F_H-1} \sum_{n=0}^{F_W-1} F_{m,n} X_{(i-m),(j-n)}.$$

일반적으로 행렬로 표현된 필터링된 이미지의 i번째 행, j번째 열의 픽셀인 G_{ij} 는 다음과 같이 원본 이미지 X 와 필터 F 의 합성곱 (convolution)으로 계산된다:

위의 식 (1)에서 F_h 와 F_w 는 각각 필터의 높이 (행의 수)와 너비 (열의 수)이다. 그림은 다양한 종류의 필터와 각 필터의 기능을 보여준다.

Operation	Filter	Filtered image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

PART1. 머신러닝

- 합성곱 신경망(CNN: Convolutional Neural Network) 구조

일반적인 인공신경망은 그림 2와 같이 affine으로 명시된 fully-connected 연산과 ReLU와 같은 비선형 활성화 함수 (nonlinear activation function)의 합성으로 정의된 계층을 여러 층 쌓은 구조이다.

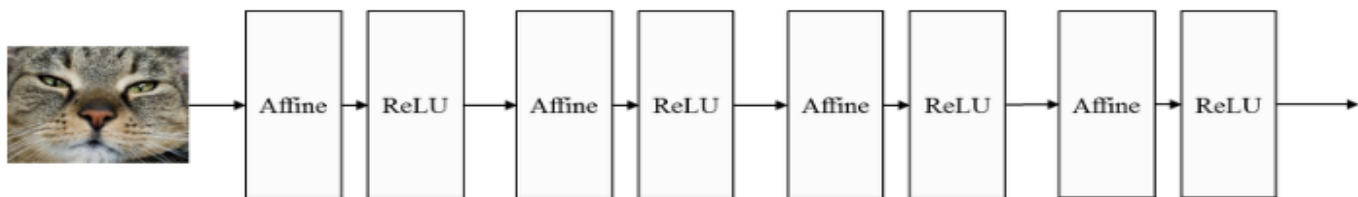


그림 2. 일반적인 인공신경망의 구조.

CNN은 그림 3과 같이 합성곱 계층 (convolutional layer)과 풀링 계층 (pooling layer)이라고 하는 새로운 층을 fully-connected 계층 이전에 추가함으로써 원본 이미지에 필터링 기법을 적용한 뒤에 필터링된 이미에 대해 분류 연산이 수행되도록 구성된다.

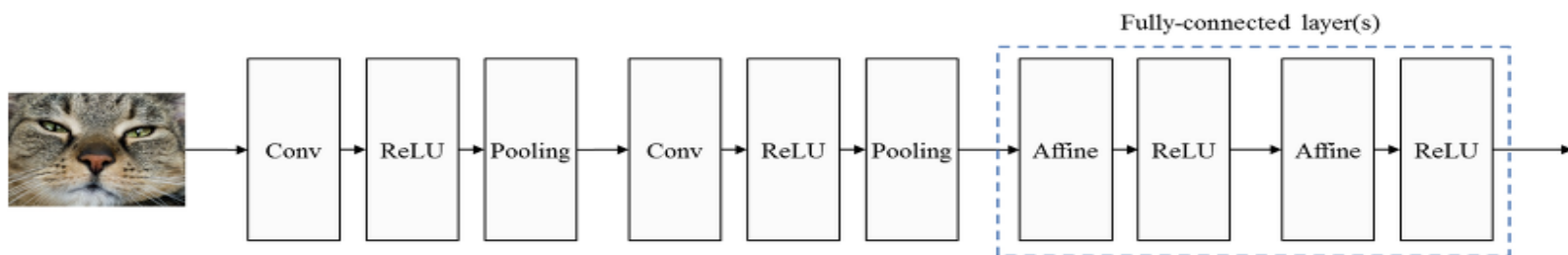


그림 3. CNN의 구조.

핸즈온 머신러닝

PART1. 머신러닝

• 합성곱 신경망 계층이란?

이미지 데이터는 높이x너비x채널의 3차원 텐서 (tensor)로 표현될 수 있다.

만약, 이미지의 색상이 RGB 코드로 표현되었다면, 채널의 크기는 3이 되며 각각의 채널에는 R, G, B 값이 저장된다. 하나의 합성곱 계층에는 입력되는 이미지의 채널 개수만큼 필터가 존재하며, 각 채널에 할당된 필터를 적용함으로써 합성곱 계층의 출력 이미지가 생성된다.

예를 들어, 높이x너비x채널이 4X4X1인 텐서 형태의 입력 이미지에 대해 3X3 크기의 필터를 적용하는 합성곱 계층에서는 그림 5와 같이 이미지와 필터에 대한 **합성곱 연산**을 통해 2X2X1 텐서 형태의 이미지가 생성된다.

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	25

그림 5. 하나의 채널에 대한 합성곱 계층의 동작.

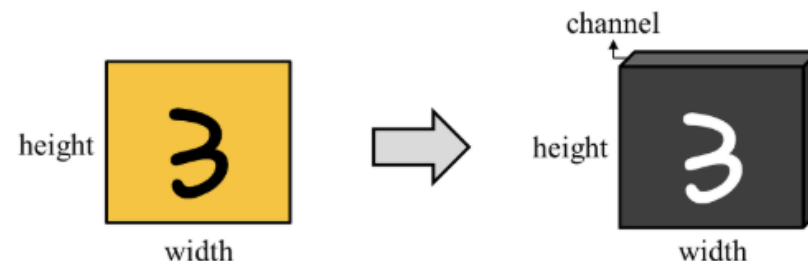
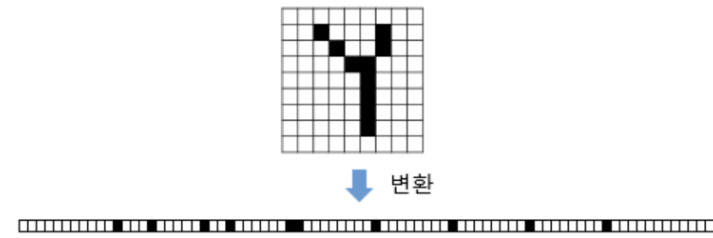


그림 4. 컬러 이미지 데이터에 대한 텐서 표현.

그림 5의 예시에서는 bias를 더하는 것이 생략되었는데, 실제 구현에서는 합성곱을 통해 생성된 행렬 형태의 이미지에 bias라는 스칼라값을 동일하게 더하도록 구현되기도 한다.

핸즈온 머신러닝

PART1. 머신러닝

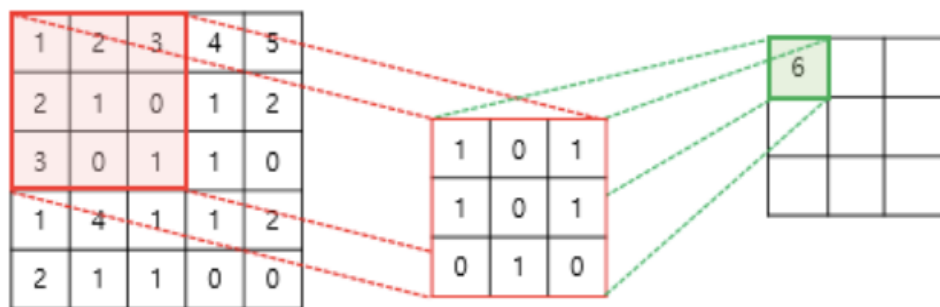


- **합성곱 연산이란?** 합성곱층은 합성곱 연산을 통해서 **이미지의 특징을 추출**하는 역할을 합니다.
- 합성곱은 영어로 컨볼루션이라고도 불리는데, **커널(kernel)** 또는 **필터(filter)**라는 $n \times m$ 크기의 행렬로 높이(height) \times 너비(width) 크기의 이미지를 처음부터 끝까지 겹치며 훑으면서 $n \times m$ 크기의 겹쳐지는 부분의 각 이미지와 커널의 원소의 값을 곱해서 모두 더한 값을 출력으로 하는 것을 말합니다. 이때, 이미지의 가장 왼쪽 위부터 가장 오른쪽까지 순차적으로 훑습니다.

커널(kernel)은 일반적으로 3×3 또는 5×5 를 사용합니다.

아래는 3×3 크기의 커널로 5×5 의 이미지 행렬에 합성곱 연산을 수행하는 과정을 보여줍니다.
한 번의 연산을 1 스텝(step)이라고 하였을 때, 합성곱 연산의 네번째 스텝까지 이미지와 식으로 표현해봤습니다.

1. 첫번째 스텝



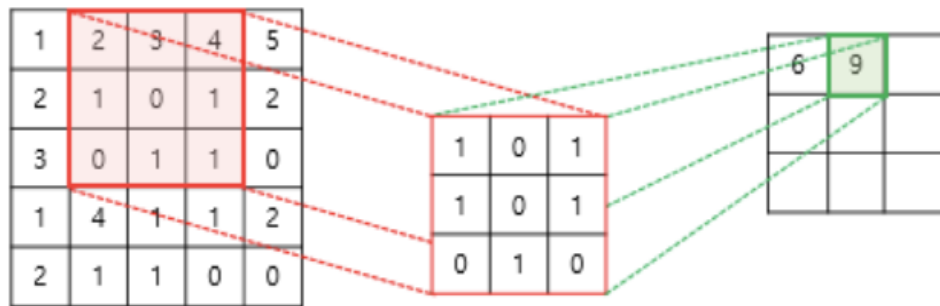
입력

커널

출력

$$(1 \times 1) + (2 \times 0) + (3 \times 1) + (2 \times 1) + (1 \times 0) + (0 \times 1) + (3 \times 0) + (0 \times 1) + (1 \times 0) = 6$$

2. 두번째 스텝



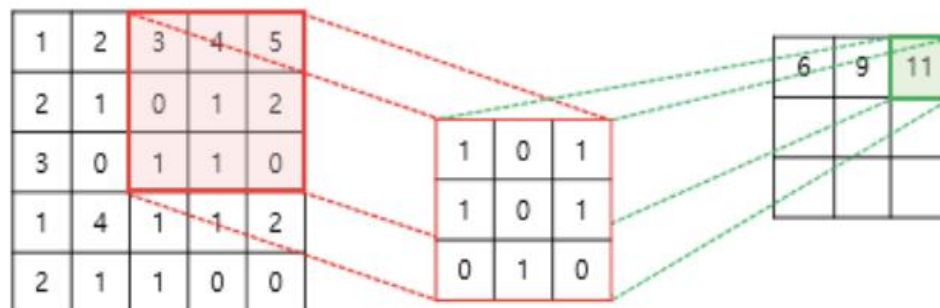
입력

커널

출력

$$(2 \times 1) + (3 \times 0) + (4 \times 1) + (1 \times 1) + (0 \times 0) + (1 \times 1) + (0 \times 0) + (1 \times 1) + (1 \times 0) = 9$$

3. 세번째 스텝



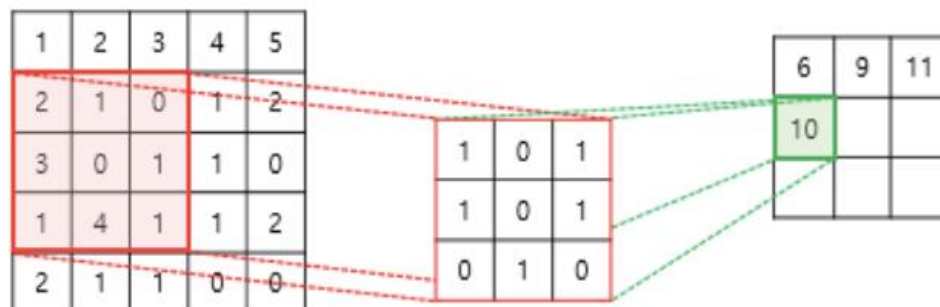
입력

커널

출력

$$(3 \times 1) + (4 \times 0) + (5 \times 1) + (0 \times 1) + (1 \times 0) + (2 \times 1) + (1 \times 0) + (1 \times 1) + (0 \times 0) = 11$$

4. 네번째 스텝



입력

커널

출력

$$(2 \times 1) + (1 \times 0) + (0 \times 1) + (3 \times 1) + (0 \times 0) + (1 \times 1) + (1 \times 0) + (4 \times 1) + (1 \times 0) = 10$$

핸즈온 머신러닝

PART1. 머신러닝

- 위 연산을 총 9번의 스텝까지 마쳤다고 가정하였을 때, 최종 결과는 아래와 같습니다.

6	9	11
10	4	4
7	7	4

특성 맵(feature map)

위와 같이 입력으로부터 커널을 사용하여 합성곱 연산을 통해 나온 결과를 **특성 맵(feature map)**이라고 합니다.

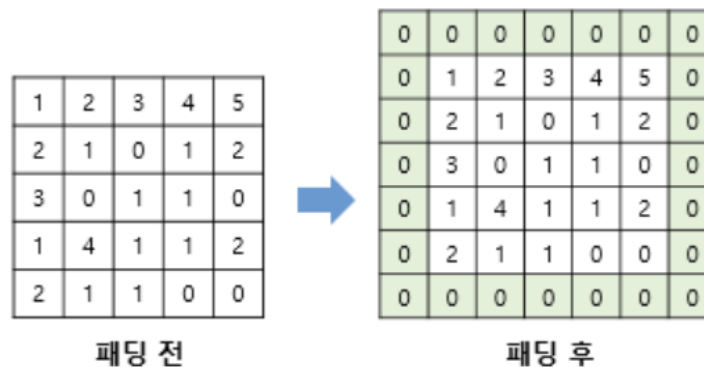
위의 예제에서는 커널의 크기가 3×3 이었지만, 커널의 크기는 사용자가 정할 수 있습니다. 또한 커널의 이동 범위가 위의 예제에서는 한 칸이었지만, 이 또한 사용자가 정할 수 있습니다. 이러한 이동 범위를 **스트라이드(stride)**라고 합니다.

핸즈온 머신러닝

PART1. 머신러닝

- **패딩(Padding):** 위의 예에서 5×5 이미지에 3×3 의 커널로 합성곱 연산을 하였을 때, 스트라이드가 1일 경우에는 3×3 의 특성 맵을 얻었습니다. 이와 같이 합성곱 연산의 결과로 얻은 특성 맵은 입력보다 크기가 작아진다는 특징이 있습니다. 만약, 합성곱 층을 여러개 쌓았다면 최종적으로 얻은 특성 맵은 초기 입력보다 매우 작아진 상태가 되버립니다. **합성곱 연산 이후에도 특성 맵의 크기가 입력의 크기와 동일하게 유지되도록 하고 싶다면 패딩(padding)을 사용하면 됩니다.**

패딩은 (합성곱 연산을 하기 전에) 입력의 가장자리에 지정된 개수의 폭만큼 행과 열을 추가해주는 것을 말합니다. 좀 더 쉽게 설명하면 지정된 개수의 폭만큼 테두리를 추가합니다. 주로 값을 0으로 채우는 제로 패딩(zero padding)을 사용합니다. 위의 그림은 5×5 이미지에 1폭짜리 제로 패딩을 사용하여 위, 아래에 하나의 행을 좌, 우에 하나의 열을 추가한 모습을 보여줍니다.

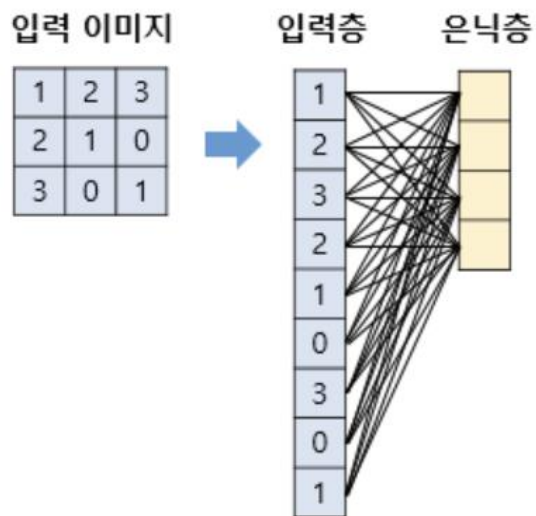


핸즈온 머신러닝

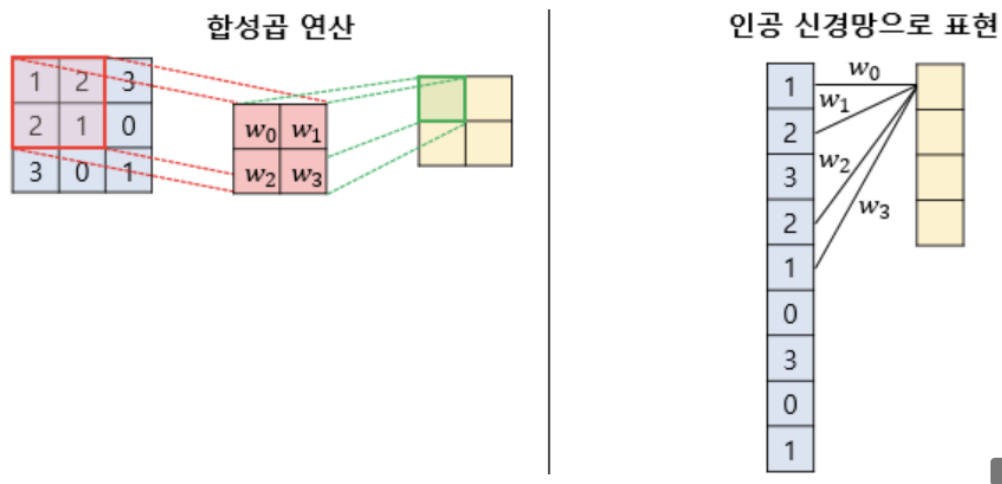
PART1. 머신러닝

• 합성곱 신경망의 가중치

다층 퍼셉트론으로 3×3 이미지를 처리한다고 가정해보겠습니다. 우선 이미지를 1차원 텐서인 벡터로 만들면, $3 \times 3 = 9$ 가 되므로 입력층은 9개의 뉴론을 가집니다. 그리고 4개의 뉴론을 가지는 은닉층을 추가한다고 해보겠습니다. 이는 아래의 그림과 같습니다. 각 연결선은 가중치를 의미하므로, 위의 그림에서는 $9 \times 4 = 36$ 개의 가중치를 가집니다.



이제 비교를 위해 합성곱 신경망으로 3×3 이미지를 처리한다고 해보겠습니다. 2×2 커널을 사용하고, 스트라이드는 1로 합니다.



핸즈온 머신러닝

PART1. 머신러닝

- 합성곱 신경망의 가중치

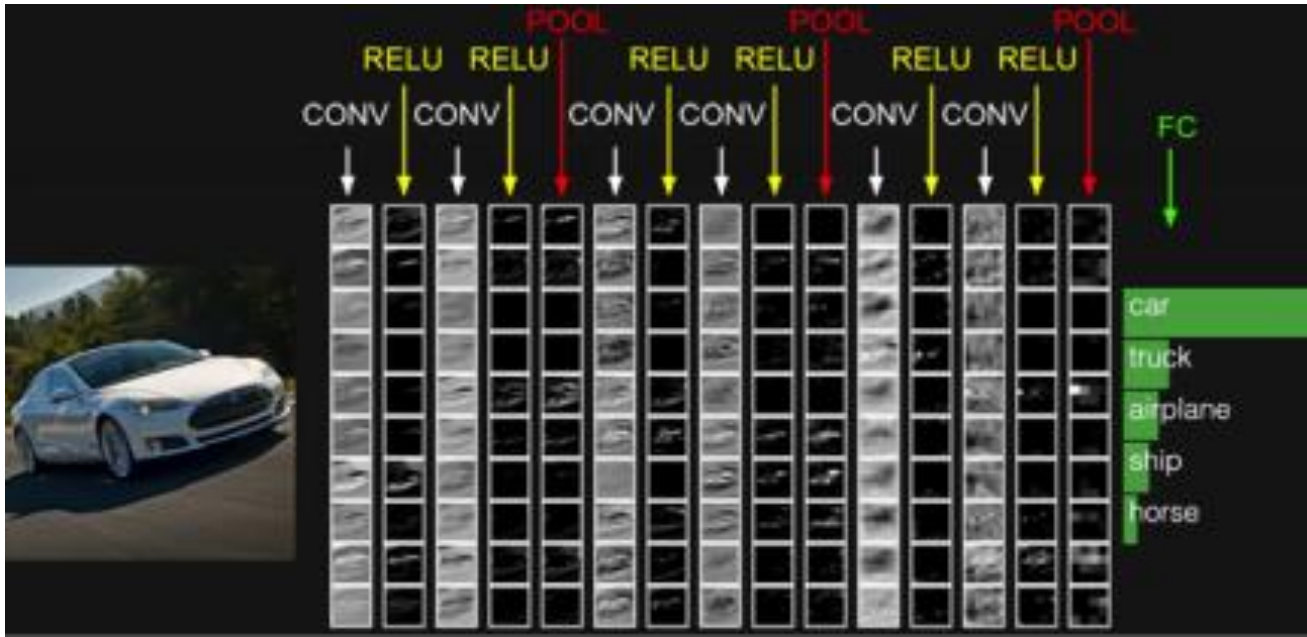
다층 퍼셉트론으로 3×3 이미지를 처리한다고 가정해보겠습니다. 우선 이미지를 1차원 텐서인 벡터로 만들면, $3 \times 3 = 9$ 가 되므로 입력층은 9개의 뉴론을 가집니다. 그리고 4개의 뉴론을 가지는 은닉층을 추가한다고 해보겠습니다. 이는 아래의 그림과 같습니다. 각 연결선은 가중치를 의미하므로, 위의 그림에서는 $9 \times 4 = 36$ 개의 가중치를 가집니다.

최종적으로 특성 맵을 얻기 위해서는 동일한 커널로 이미지 전체를 훑으며 합성곱 연산을 진행합니다. 결국 이미지 전체를 훑으면서 사용되는 가중치는 $w_0w_0, w_1w_1, w_2w_2, w_3w_3$ 4개 뿐입니다. 그리고 각 합성곱 연산마다 이미지의 모든 픽셀을 사용하는 것이 아니라, 커널과 맵핑되는 픽셀만을 입력으로 사용하는 것을 볼 수 있습니다. 결국 합성곱 신경망은 다층 퍼셉트론을 사용할 때보다 훨씬 적은 수의 가중치를 사용하며 공간적 구조 정보를 보존한다는 특징이 있습니다.

다층 퍼셉트론의 은닉층에서는 가중치 연산 후에 비선형성을 추가하기 위해서 활성화 함수를 통과시켰습니다. 합성곱 신경망의 은닉층에서도 마찬가지입니다. 합성곱 연산을 통해 얻은 특성 맵은 다층 퍼셉트론때와 마찬가지로 비선형성 추가를 위해서 활성화 함수를 지나게 됩니다. 이때 렐루 함수나 렐루 함수의 변형들이 주로 사용됩니다.

PART1. 머신러닝

- 합성곱 신경망(CNN: Convolutional Neural Network) 구조



위의 그림에서 CONV는 합성곱 연산을 의미하고, 합성곱 연산의 결과가 활성화 함수 ReLU를 지납니다. 이 두 과정을 합성곱층이라고 합니다. 그 후에 POOL이라는 구간을 지나는데 이는 풀링 연산을 의미하며 풀링층이라고 합니다.

합성곱 계층은 이미지에 필터링 기법이 적용하고, 풀링 계층은 이미지의 국소적인 부분들을 하나의 대표적인 스칼라 값으로 변환함으로써 이미지의 크기를 줄이는 등의 다양한 기능들을 수행한다.

핸즈온 머신러닝

PART1. 머신러닝

- 합성곱 신경망 계층이란?

이미지 데이터는 높이x너비x채널의 3차원 텐서 (tensor)로 표현될 수 있다.

만약, 이미지의 색상이 RGB 코드로 표현되었다면, 채널의 크기는 3이 되며 각각의 채널에는 R, G, B 값이 저장된다. 하나의 합성곱 계층에는 입력되는 이미지의 채널 개수만큼 필터가 존재하며, 각 채널에 할당된 필터를 적용함으로써 합성곱 계층의 출력 이미지가 생성된다.

예를 들어, 높이x너비x채널이 4X4X1인 텐서 형태의 입력 이미지에 대해 3X3 크기의 필터를 적용하는 합성곱 계층에서는 그림 5와 같이 이미지와 필터에 대한 합성곱 연산을 통해 2X2X1 텐서 형태의 이미지가 생성된다.

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	25

그림 5. 하나의 채널에 대한 합성곱 계층의 동작.

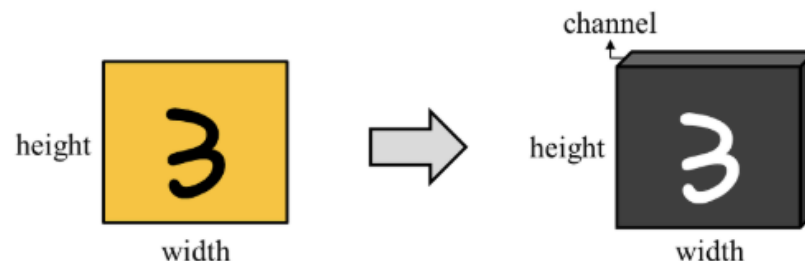


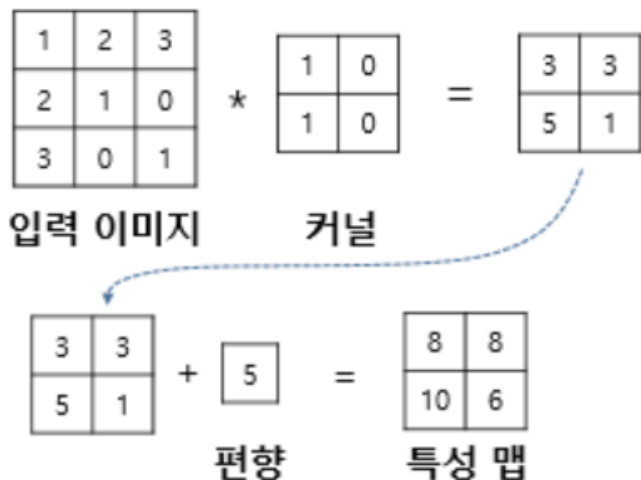
그림 4. 컬러 이미지 데이터에 대한 텐서 표현.

그림 5의 예시에서는 bias를 더하는 것이 생략되었는데, 실제 구현에서는 합성곱을 통해 생성된 행렬 형태의 이미지에 bias라는 스칼라값을 동일하게 더하도록 구현되기도 한다.

핸즈온 머신러닝

PART1. 머신러닝

- 합성곱 신경망의 편향



합성곱 신경망에도 편향(bias)를 당연히 추가할 수 있습니다. 만약, 편향을 사용한다면 커널을 적용한 뒤에 더해집니다. 편향은 하나의 값만 존재하며, 커널이 적용된 결과의 모든 원소에 더해 집니다.

입력값이 0이 들어가면 어떠한 학습률을 넣어도 이후 가중치에 더해지는 값이 없다.

천번의 가중치 조정은 의미가 없다.

이러한 경우를 방지하고자 **편향**이라는 개념이 등장했다. 의미 그대로 입력으로는 늘 한 쪽으로 치우쳐진 고정 값이며 입력으로 받은 값이 0인 경우에도 아무것도 학습하지 못하는 것을 방지한다.

핸즈온 머신러닝

PART1. 머신러닝

• 특성 맵의 크기 계산 방법

입력의 크기와 커널의 크기, 그리고 스트라이드의 값만 알면 합성곱 연산의 결과인 특성 맵의 크기를 계산할 수 있습니다.

- I_h : 입력의 높이
- I_w : 입력의 너비
- K_h : 커널의 높이
- K_w : 커널의 너비
- S : 스트라이드
- O_h : 특성 맵의 높이
- O_w : 특성 맵의 너비

이에 따라 특성 맵의 높이와 너비는 다음과 같습니다.

$$O_h = \text{floor}(\frac{I_h - K_h}{S} + 1)$$

패딩의 폭을 P 라고 하고, 패딩까지 고려한 식은 다음과 같습니다.

$$O_h = \text{floor}(\frac{I_h - K_h + 2P}{S} + 1)$$

$$O_w = \text{floor}(\frac{I_w - K_w}{S} + 1)$$

$$O_w = \text{floor}(\frac{I_w - K_w + 2P}{S} + 1)$$

여기서 *floor* 함수는 소수점 발생 시 소수점 이하를 버리는 역할을 합니다. 예를 들어 위의 첫 번째 예제의 경우 5×5 크기의 이미지에 3×3 커널을 사용하고 스트라이드 1로 합성곱 연산을 했습니다. 이 경우 특성 맵의 크기는 $(5 - 3 + 1) \times (5 - 3 + 1) = 3 \times 3$ 임을 알 수 있습니다. 이는 또한 총 9번의 스텝이 필요함을 의미하기도 합니다.

핸즈온 머신러닝

PART1. 머신러닝

- 합성곱 신경망 계층이란?

이미지 데이터는 높이x너비x채널의 3차원 텐서 (tensor)로 표현될 수 있다.

만약, 이미지의 색상이 RGB 코드로 표현되었다면, 채널의 크기는 3이 되며 각각의 채널에는 R, G, B 값이 저장된다.

하나의 합성곱 계층에는 입력되는 이미지의 채널 개수만큼 필터가 존재하며, 각 채널에 할당된 필터를 적용함으로써 합성곱 계층의 출력 이미지가 생성된다.

예를 들어, 높이x너비x채널이 4x4x1인 텐서 형태의 입력 이미지에 대해 3x3 크기의 필터를 적용하는 합성곱 계층에서는 그림 5와 같이 이미지와 필터에 대한 합성곱 연산을 통해 2x2x1 텐서 형태의 이미지가 생성된다.

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	

0	1	7	5
5	5	6	6
5	3	3	0
1	1	1	2

 \otimes

1	0	1
1	2	0
3	0	1

 $=$

40	32
26	25

그림 5. 하나의 채널에 대한 합성곱 계층의 동작.

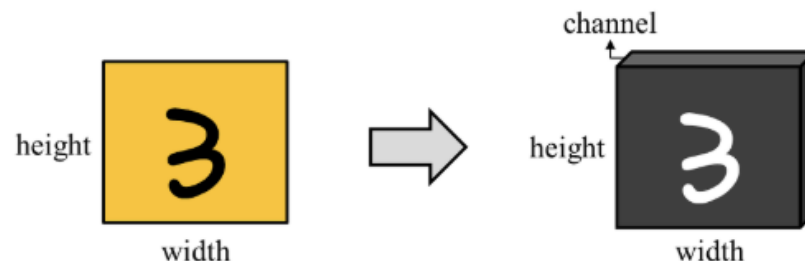


그림 4. 컬러 이미지 데이터에 대한 텐서 표현.

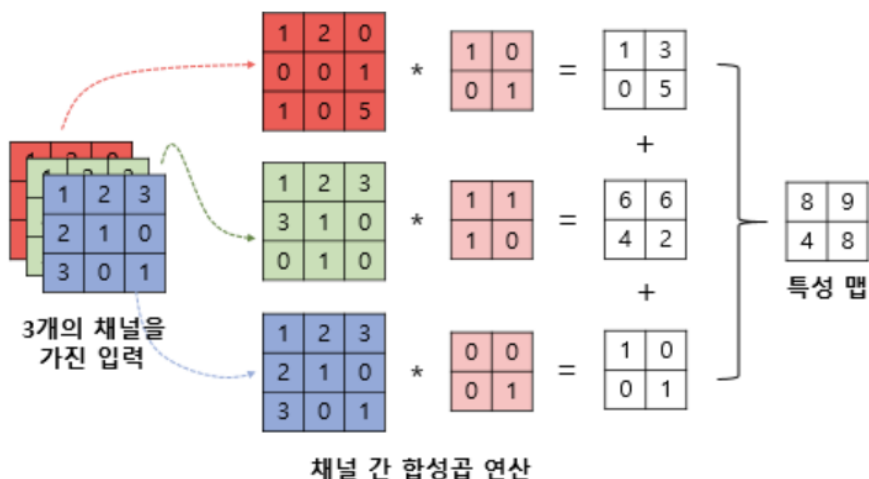
그림 5의 예시에서는 bias를 더하는 것이 생략되었는데, 실제 구현에서는 합성곱을 통해 생성된 행렬 형태의 이미지에 bias라는 스칼라값을 동일하게 더하도록 구현되기도 한다.

핸즈온 머신러닝

PART1. 머신러닝

- 다수의 채널을 가질 경우의 합성곱 연산(3차원 텐서의 합성곱 연산)

지금까지는 채널(channel) 또는 깊이(depth)를 고려하지 않고, 2차원 텐서를 가정하고 설명했습니다. 하지만 실제로 합성곱 연산의 입력은 '다수의 채널을 가진' 이미지 또는 이전 연산의 결과로 나온 특성 맵일 수 있습니다. 만약, 다수의 채널을 가진 입력 데이터를 가지고 합성곱 연산을 한다고 하면 **커널의 채널 수도 입력의 채널 수만큼 존재**해야 합니다. 다시 말해 **입력 데이터의 채널 수와 커널의 채널 수는 같아야 합니다**. 채널 수가 같으므로 합성곱 연산을 채널마다 수행합니다. 그리고 그 결과를 모두 더하여 최종 특성 맵을 얻습니다.



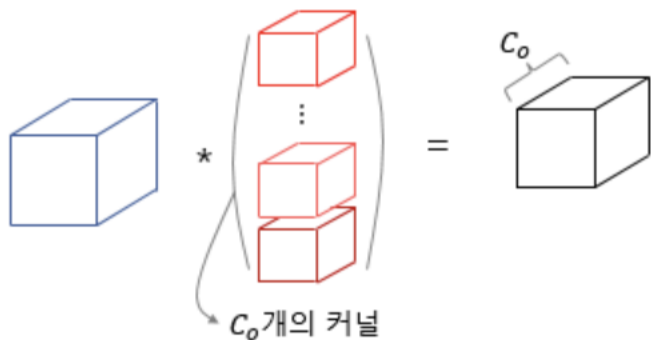
커널의 각 채널끼리의 크기는 같아야 합니다. 각 채널 간 합성곱 연산을 마치고, 그 결과를 모두 더해서 하나의 채널을 가지는 특성 맵을 만듭니다.

위 그림은 높이 3, 너비 3, 채널 3의 입력이 높이 2, 너비 2, 채널 3의 커널과 합성곱 연산을 하여 높이 2, 너비 2, 채널 1의 특성 맵을 얻는다는 의미입니다. 합성곱 연산의 결과로 얻은 특성 맵의 채널 차원은 RGB 채널 등과 같은 컬러의 의미를 담고 있지는 않습니다.

핸즈온 머신러닝

PART1. 머신러닝

• 3차원 텐서의 합성곱 연산

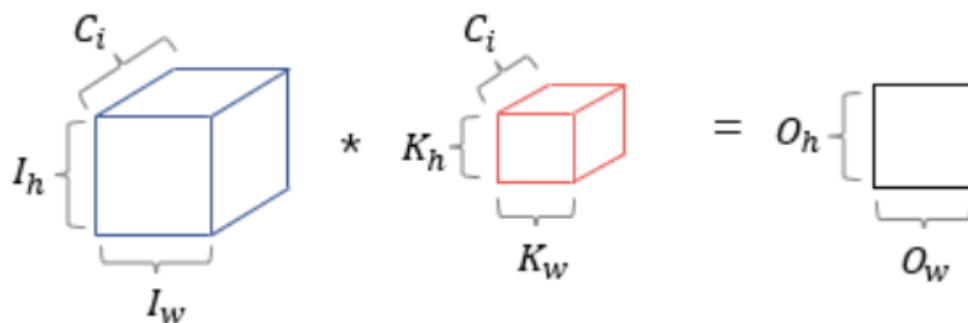


합성곱 연산에서 다수의 커널을 사용할 경우, 사용한 커널 수는 합성곱 연산의 결과로 나오는 특성 맵의 채널 수가 됩니다.

일반화를 위해 사용하는 각 변수가 의미하는 바는 다음과 같습니다.

- I_h : 입력의 높이
- I_w : 입력의 너비
- K_h : 커널의 높이
- K_w : 커널의 너비
- O_h : 특성 맵의 높이
- O_w : 특성 맵의 너비
- C_i : 입력 데이터의 채널

다음은 3차원 텐서의 합성곱 연산을 보여줍니다.



높이 I_h , 너비 I_w , 채널 C_i 의 입력 데이터는 동일한 채널 수 C_i 를 가지는 높이 K_h , 너비 K_w 의 커널과 합성곱 연산을 하여 높이 O_h , 너비 O_w , 채널 1의 특성 맵을 얻습니다. 그런데 하나의 입력에 여러 개의 커널을 사용하는 합성곱 연산을 할 수도 있습니다.

핸드온 머신러닝
 PART1. 머신러닝

이를 이해했다면 커널의 크기와 입력 데이터의 채널 수 C_i 와 특성 맵(출력 데이터)의 채널 수 C_o 가 주어졌을 때, 가중치 매개변수의 총 개수를 구할 수 있습니다. 가중치는 커널의 원소들이므로 하나의 커널의 하나의 채널은 $K_i \times K_o$ 개의 매개변수를 가지고 있습니다. 그런데 합성곱 연산을 하려면 커널은 입력 데이터의 채널 수와 동일한 채널 수를 가져야 합니다. 이에 따라 하나의 커널이 가지는 매개변수의 수는 $K_i \times K_o \times C_i$ 입니다. 그런데 이러한 커널이 총 C_o 개가 있어야 하므로 가중치 매개변수의 총 수는 다음과 같습니다.

가중치 매개변수의 총 수 : $K_i \times K_o \times C_i \times C_o$

9. 풀링(Pooling)

일반적으로 합성곱 층(합성곱 연산 + 활성화 함수) 다음에는 풀링 층을 추가하는 것이 일반적입니다. 풀링 층에서는 특성 맵을 다운샘플링하여 특성 맵의 크기를 줄이는 풀링 연산이 이루어집니다. 풀링 연산에는 일반적으로 최대 풀링(max pooling)과 평균 풀링(average pooling)이 사용됩니다. 우선 최대 풀링을 통해서 풀링 연산을 이해해봅시다.

