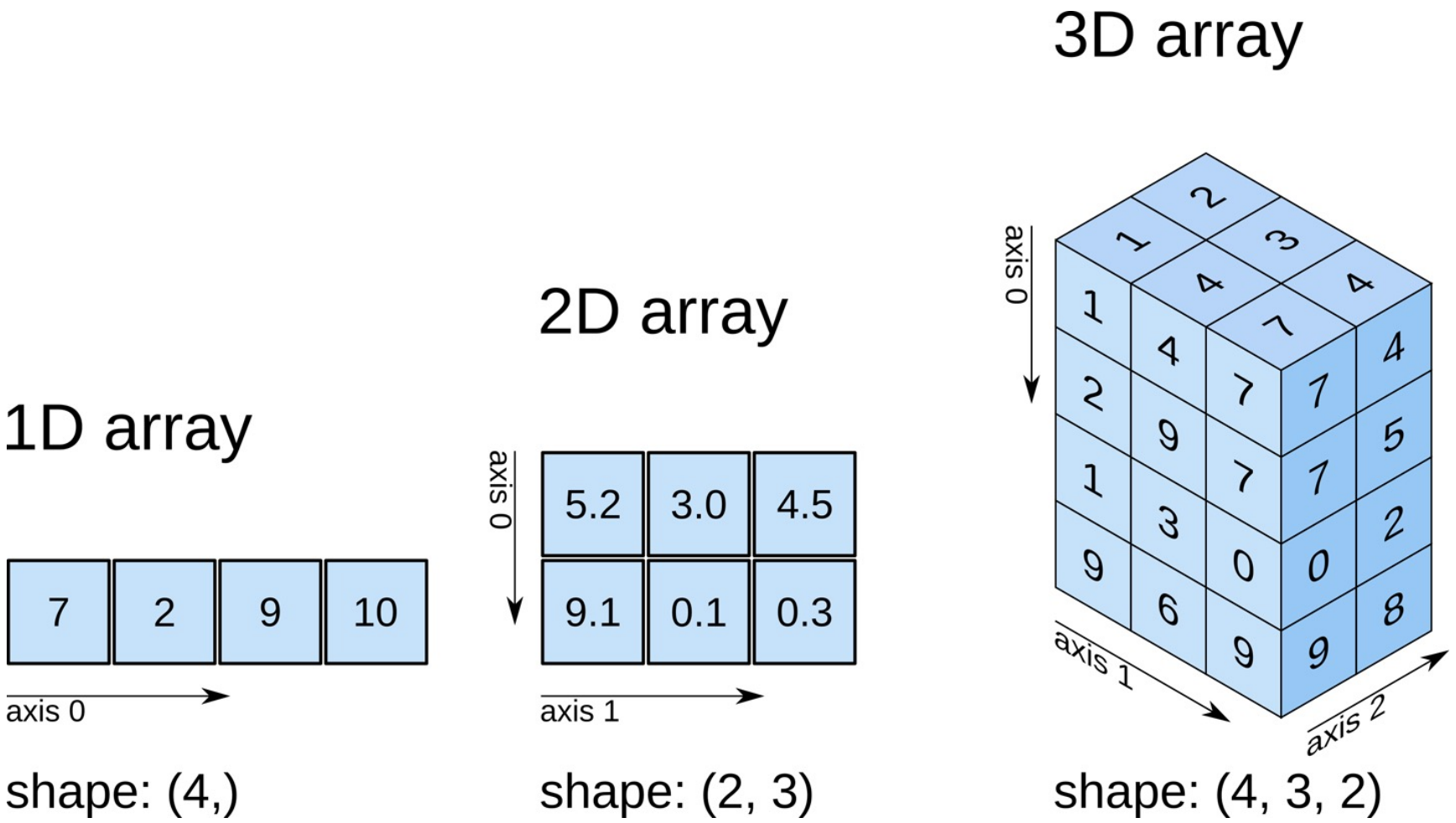
신경망 데이터 표현

김영욱

Hello AI

텐서(Tensor)


1D array
7
2
9
10
axis 0
shape: (4,)
2D array
axis 0
5.2
3.0
4.5
9.1
0.1
0.3
axis 1
shape: (2, 3)
3D array
axis 0
axis 1
axis 2
shape: (4, 3, 2)

텐서(Tensor)

스칼라(0차원 텐서)
- 하나의 숫자를 담고 있는 텐서(tensor)
- 형상은 없음

벡터의 곱

- $A = (x_1,\ x_2,\ x3,\ \ldots,\ x_n)$
  $B = (y_1,\ y_2,\ y3,\ \ldots,\ y_n)$ 일 때,
- 원소곱
  - 같은 형상(shape)일 때, 각 원소별로 계산

$$A \times B = (x_1,\ x_2,\ x_3,\ \ldots,\ x_n) \times (y_1,\ y_2,\ y_3,\ \ldots,\ y_n)$$
$$= (x_1 y_1,\ x_2 y_2,\ x_3 y_3,\ \ldots\ ,\ x_n y_n)$$

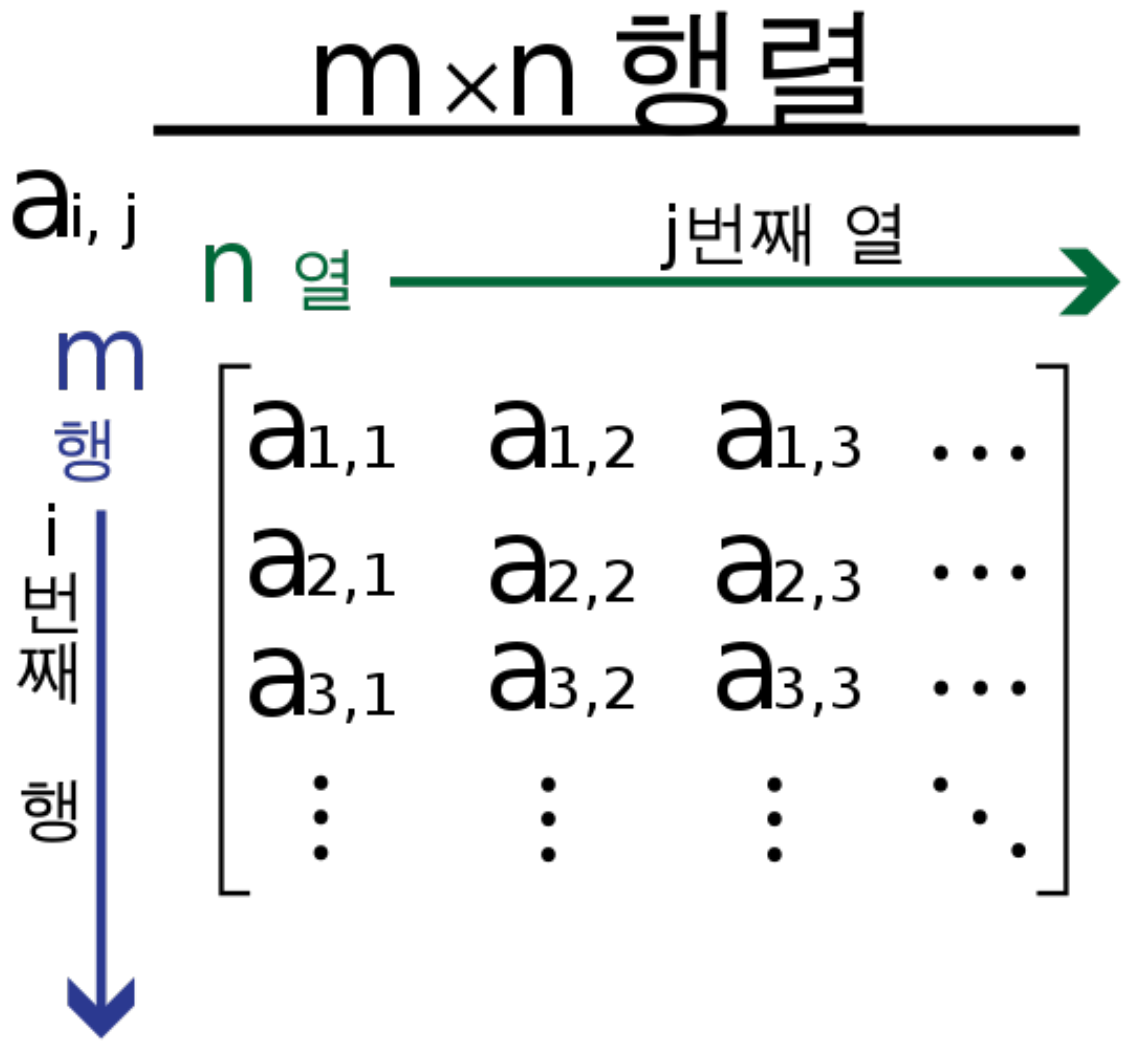
- 벡터곱(product, dot)
  - 두 1차원 벡터가 있을 때 **각각의 성분끼리의 곱을 모두 더하는 계산**

$$A \bullet B \Rightarrow A \times B^T = (x_1,\ x_2,\ x_3,\ \ldots,\ x_n) \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \ldots \\ y_n \end{pmatrix}$$

$$= (x_1 y_1 + x_2 y_2 + x_3 y_3 + \ldots + x_n y_n)$$

2차원 텐서(행렬)

- 2차원 텐서는 행렬로 생각할 수 있음
  - (m, n) 형상의 배열



전치행렬

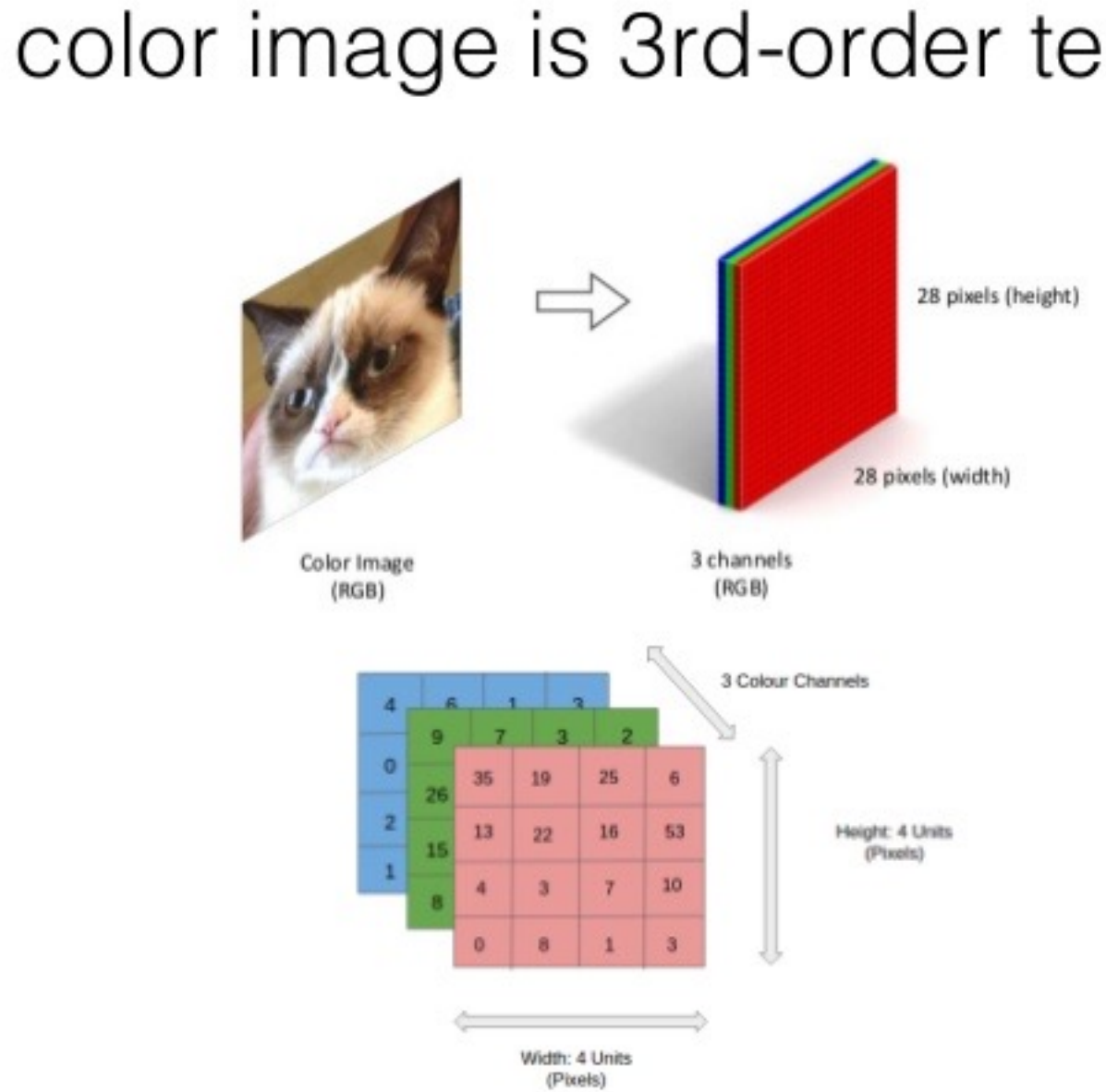
- 행과 열을 바꾼 배열의 형태

$$A$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

3차원 텐서

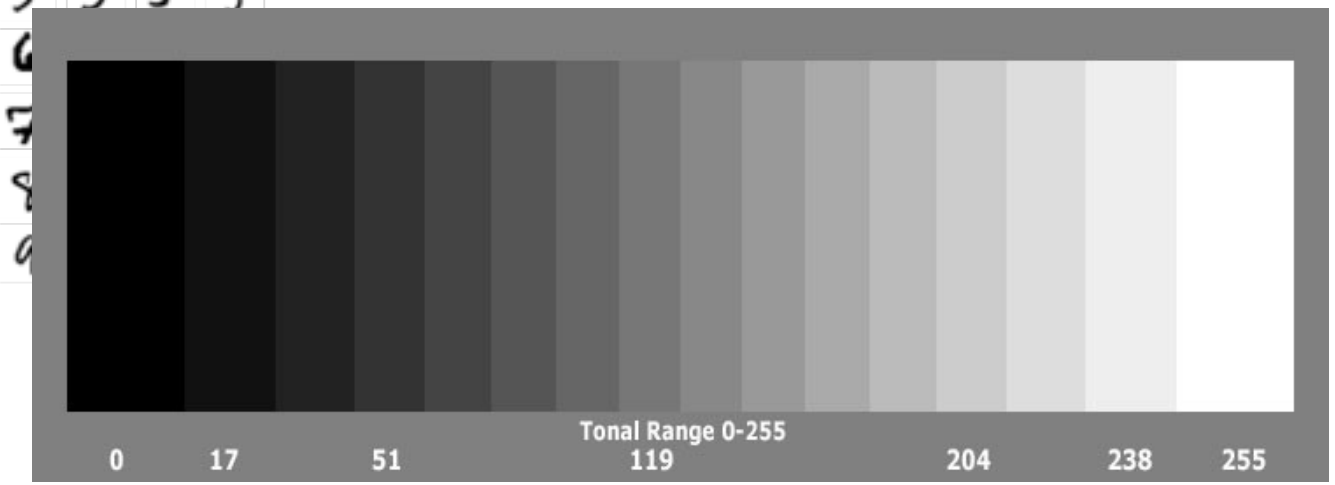
- 보통 이미지를 나타낼 때 사용되는 텐서
  - (width, height, channels)
  - 일반적으로 Numpy array로 표현



3차원 텐서 활용 예시(이미지)

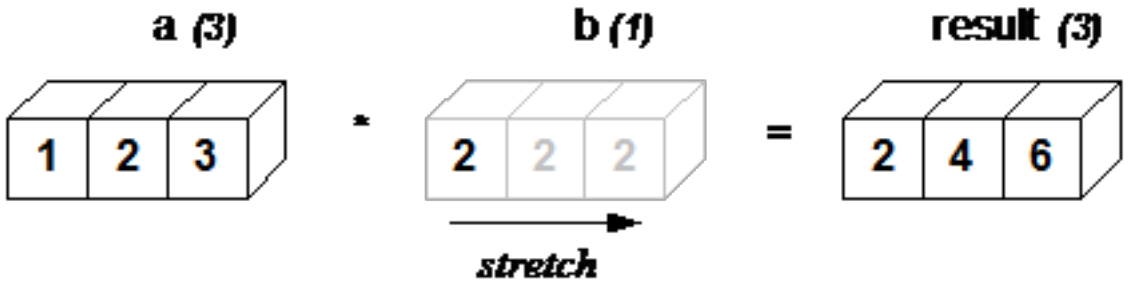
- MNIST Dataset
- 28x28 사이즈의 gray scale 이미지들로 구성

gray scale: 0~255의 값을 통해 밝기를 표현 0으로 갈수록 어두워지고, 255로 갈수록 밝아짐

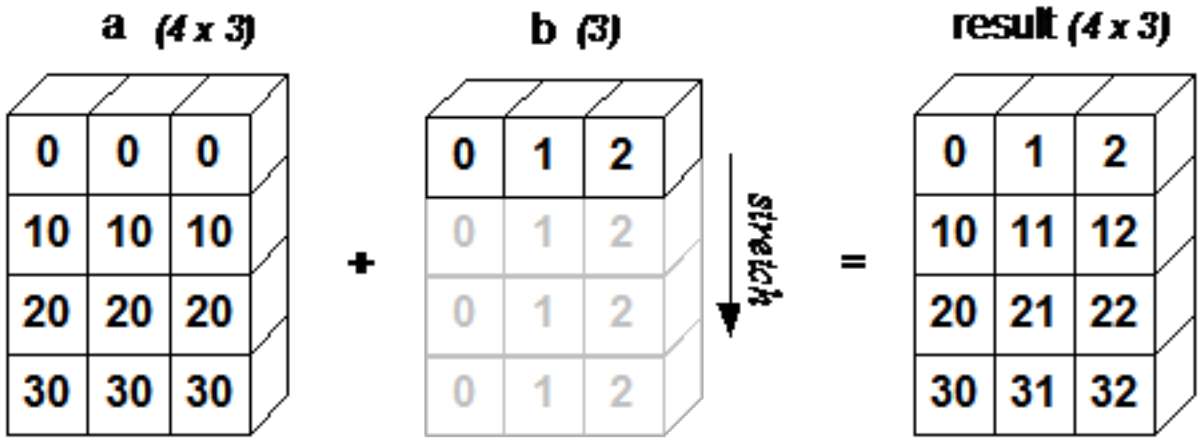


브로드캐스팅(broadcasting)

- 넘파이에서 다른 형상(shape)끼리 계산 가능
- 1차원 텐서



•2차원 텐서



4, 5차원 텐서

- Color Image Datasets(4차원)
  - (samples, height, width, channels) (Keras, Tensorflow)
  - (samples, channels, height, width) (Pytorch)

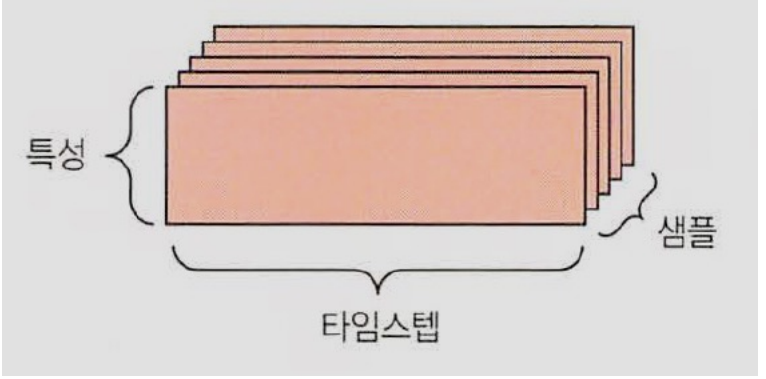
- 동영상(5차원)
  1. (samples, frames, height, width, channels)
  2. (samples, frames, channels, height, width)
  - 예시 1) (4, 300, 1920, 1080, 3)
    - 1920x1080 사이즈 3채널의 300프레임 수를 가진 배치가 4개

- **벡터 데이터**: (samples, features) 크기의 2D 텐서
- **시계열 데이터** 또는 **시퀀스**(sequence) **데이터**: (samples, timesteps, features) 크기의 3D 텐서
- **이미지**: (samples, height, width, channels) 또는 (samples, channels, height, width) 크기의 4D 텐서
- **동영상**: (samples, frames, height, width, channels) 또는 (samples, frames, channels, height, width) 크기의 5D 텐서

벡터 데이터

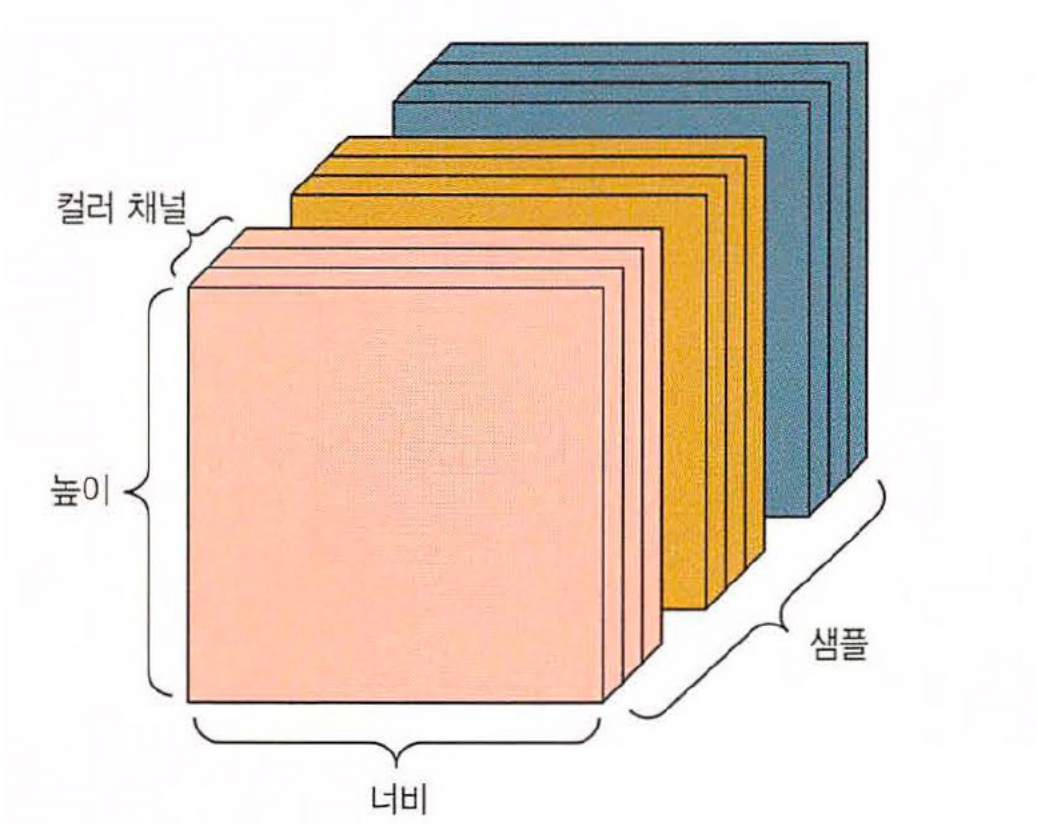
- 사람의 나이, 우편 번호, 소득으로 구성된 인구 통계 데이터. 각 사람은 3개의 값을 가진 벡터로 구성되고 10만 명이 포함된 전체 데이터셋은 (100000, 3) 크기의 텐서에 저장될 수 있습니다.
- (공통 단어 2만 개로 만든 사전에서) 각 단어가 등장한 횟수로 표현된 텍스트 문서 데이터셋. 각 문서는 2만 개의 원소(사전에 있는 단어마다 하나의 원소에 대응합니다)를 가진 벡터로 인코딩될 수 있습니다. 500개의 문서로 이루어진 전체 데이터셋은 (500, 20000) 크기의 텐서로 저장됩니다.

시계열 데이터 또는 시퀀스 데이터



- **주식 가격 데이터셋**: 1분마다 현재 주식 가격, 지난 1분 동안에 최고 가격과 최소 가격을 저장합니다. 1분마다 데이터는 3D 벡터로 인코딩되고 하루 동안의 거래는 (390, 3) 크기의 2D 텐서로 인코딩됩니다(하루의 거래 시간은 390분입니다[13]). 250일치의 데이터는 (250, 390, 3) 크기의 3D 텐서로 저장될 수 있습니다. 여기에서 1일치 데이터가 하나의 샘플이 됩니다.
- **트윗 데이터셋**: 각 트윗은 128개의 알파벳으로 구성된 280개의 문자 시퀀스입니다. 여기에서는 각 문자가 128개의 크기인 이진 벡터로 인코딩될 수 있습니다(해당 문자의 인덱스만 1이고 나머지는 모두 0인 벡터). 그러면 각 트윗은 (280, 128) 크기의 2D 텐서로 인코딩될 수 있습니다. 100만 개의 트윗으로 구성된 데이터셋은 (1000000, 280, 128) 크기의 텐서에 저장됩니다.

이미지 데이터



256 x 256 사이즈 이미지 128장의 경우

흑백 (128,256,256,1)
컬러 (128,256,256,3)

(Samples, height, width, color_depth)

비디오 데이터

비디오 데이터는 현실에서 5D 텐서가 필요한 몇 안 되는 데이터 중 하나입니다.
하나의 비디오는 프레임의 연속이고 각 프레임은 하나의 컬러 이미지입니다.

프레임이 (height, width, color_depth)의 3D 텐서로 저장될 수 있기 때문에 프레임의 연속은 (frames, height, width, color_depth)의 4D 텐서로 저장될 수 있습니다.
여러 비디오의 배치는 (samples, frames, height, width, color_depth)의 5D 텐서로 저장될 수 있습니다.
예를 들어 60초짜리 144×256 유튜브 비디오 클립을 초당 4프레임으로 샘플링하면 240프레임이 됩니다. 이런 비디오 클립을 4개 가진 배치는 (4, 240, 144, 256, 3) 크기의 텐서에 저장될 것입니다. 총 106,168,320개의 값이 있습니다! 이 텐서의 dtype을 float32로 했다면[16] 각 값이 32비트로 저장될 것이므로 텐서의 저장 크기는 405MB가 됩니다. 아주 크네요! 실생활에서 접하는 비디오는 float32 크기로 저장되지 않기 때문에 훨씬 용량이 적고, 일반적으로 높은 압축률로 (MPEG 포맷 같은 방식을 사용하여) 압축되어 있습니다.