

Chapter02.

Tensor Handling

작성자 : 김진성

목차

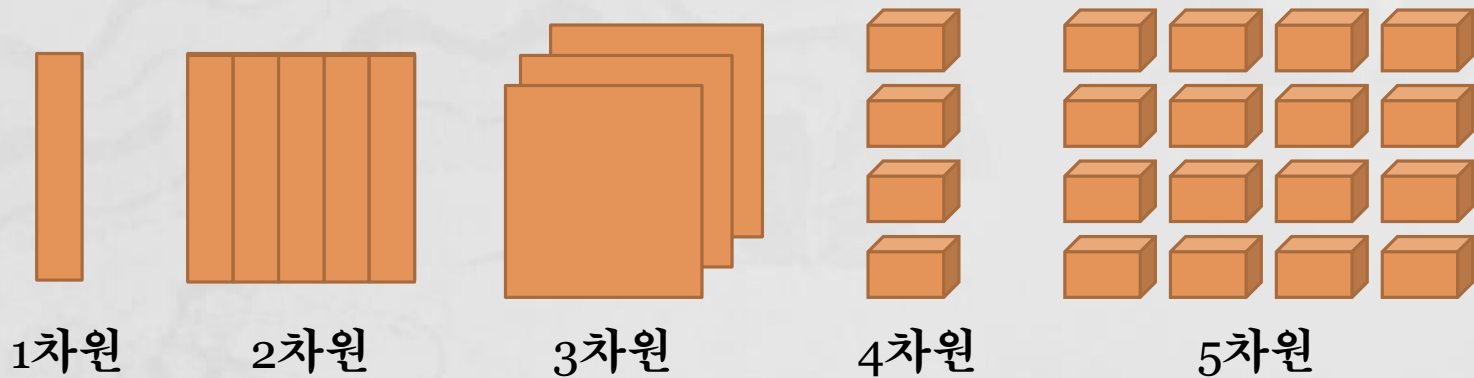
1. **Tensor?**
2. **Tensor 처리 함수**
3. **Tensor 다루기**

1. Tensor?

- Tensorflow의 모든 자료는 Tensor로 표현
 - Tensor : N 차원 배열
- Tensor 속성
 - Rank : Tensor 차원 수
 - Shape : Tensor 모양
 - Size : Tensor 원소 수
- Data types
 - Int16/32/64, float32/64, string, bool ...

Tensor는 0 ~ n 차원까지 갖는 자료구조

- 0차원 : Scalar
- 1차원 : Vector
- 2차원 : Vector 행렬 구조
- 3차원 : 2차원 Tensor 일렬 구성
- N차원 : n-1차 Tensor 일렬 구성



● Rank, Shape 예

Python code	Rank	Shape	Meth Entity
120	0	[]	Scalar
[1, 2, 3]	1	[3]	Vector
[[1, 2, 3], [4, 5, 6]]	2	[2, 3]	Matrix
[[[1 2] [3 4] [5 6]] [[7 8] [9 10] [11 12]]]	3	[2, 3, 2]	3-Tensor
	n	$[d_0, d_1, \dots d_{n-1}]$	N-Tensor

● Numpy vs Tensorflow

Numpy	Tensorflow
<code>np.zeros((3,3)), np.ones((3,3))</code>	<code>tf.zeros((3,3)), tf.ones((3, 3))</code>
<code>np.sum(data, axis=1)</code>	<code>tf.reduce_sum(data, axis=1]</code>
<code>var.shape()</code>	<code>var.get_shape()</code>
<code>np.reshape(data, (2, 3))</code>	<code>tf.reshape(data, (2, 3))</code>
<code>np.dot(x, y)</code>	<code>tf.matmul(x, y)</code>
<code>data[o, o]</code> <code>data[: , o]</code> <code>data[o, :]</code>	<code>data[o, o]</code> <code>data[: , o]</code> <code>data[o, :]</code>

2. Tensor 처리 함수

● 주요 수학 관련 함수

tf.add()

tf.subtract()

tf.multiply()

tf.div()

tf.mod() : 나머지

tf.abs() : 절대값

tf.square() : 제곱

tf.sqrt() : 제곱근

tf.round() : 반올림

tf.pow() : 거듭제곱

tf.exp() : 지수값

tf.log() : 로그값

tf.cumsum() : 누적합

tf.cumprod() : 누적곱

● 행렬 연산 함수

tf.diag : 대각행렬 반환

tf.transpose : 전치행렬 반환

tf.matmul : 두 텐서의 행렬곱 반환

tf.matrix_determinant : 정방행렬 행렬식 반환

tf.matrix_inverse : 정방행렬 역행렬 반환

● Transform 관련 함수

tf.shape() : 차원 모양 변경

tf.squeeze() : 차원 크기가 1인 경우 제거

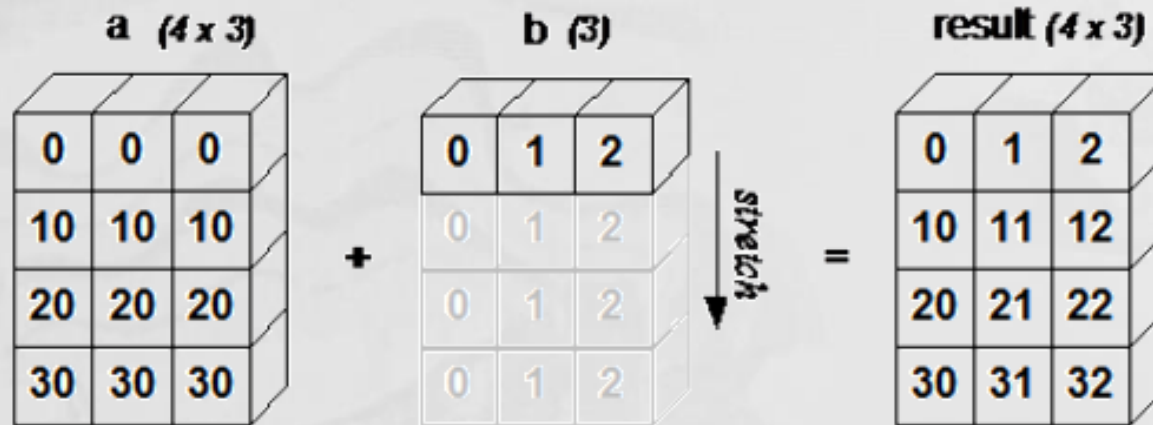
tf.slice() : tensor 자르기

tf.expand_dims() : 축 단위 차원 추가

❖ 브로드캐스팅

행렬 연산 (덧셈, 뺄셈, 곱셈)에서 차원이 맞지 않을 때, 행렬을 자동으로 늘려줘서 (Stretch) 차원을 맞춰주는 기법

- 늘리는 것은 가능하지만 줄이는 것은 불가능하다.



❖ 행렬 연산 함수

● tf.diag : 대각행렬 반환

주 대각선 성분 이외의 모든 성분 0행렬

X =

```
[[ 0.43340078  0.8103979 ]  
 [ 0.78758866  0.20491953]]
```



X 대각행렬

```
[[  
  [[ 0.43340078  0.    ]  
   [ 0.    0.    ]]  
  
  [[ 0.    0.8103979 ]  
   [ 0.    0.    ]]  
  
  [[[ 0.    0.    ]  
   [ 0.78758866  0.    ]]  
  
  [[ 0.    0.    ]  
   [ 0.    0.20491953]]  
]]
```

❖ 행렬 연산 함수

- `tf.matrix_determinant` : 정방행렬 행렬식 반환

$$X = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$X \text{ 정방행렬 행렬식} = ad - bc$$

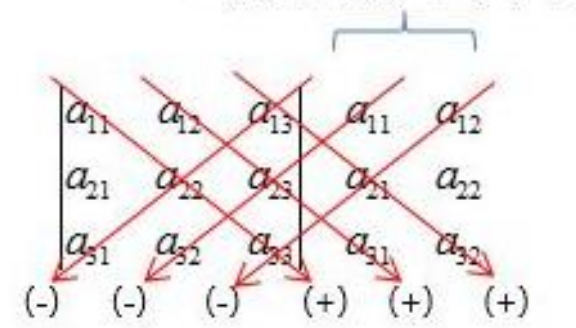
$$\begin{bmatrix} 0.43340078 & 0.8103979 \\ 0.78758866 & 0.20491953 \end{bmatrix}$$



$$-0.549447907134$$

- 3차 행렬식 (determinant of 3rd order)

처음 두 열을 복사하여 붙임

$$\begin{aligned}
 \det \mathbf{A} &= \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} \\
 &= a_{11}a_{22}a_{33} - a_{11}a_{23}a_{32} + a_{12}a_{23}a_{31} - a_{12}a_{21}a_{33} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} \\
 &= a_{11} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} + a_{12} \left(- \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} \right) + a_{13} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{32} \end{vmatrix}
 \end{aligned}$$


❖ 행렬 연산 함수

- tf.matrix_inverse : 정방행렬 역행렬 반환

X 정방행렬 역행렬

$$X = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow A^{-1} = \frac{1}{ad-bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

$$\begin{bmatrix} 0.43566236 & 0.06891132 \\ 0.96319346 & 0.6941725 \end{bmatrix} \Rightarrow \begin{bmatrix} 2.94078713 & -0.2919354 \\ -4.08046547 & 1.84563674 \end{bmatrix}$$

$$\begin{bmatrix} 0.6941725 & -0.06891132 \\ -0.96319346 & 0.43566236 \end{bmatrix} \xleftarrow{4.3402 *}$$

❖ 행렬 연산 함수

- tf.matmul : 두 텐서의 행렬곱 반환

X =

```
[[ 0.43340078  0.8103979 ]  
 [ 0.78758866  0.20491953]]
```

Y =

```
[[ 0.84916386  0.63782171]  
 [ 0.66741557  0.0765041 ]]
```

X*Y 행렬곱

```
[[ 0.90890046  0.33843119]  
 [ 0.80555832  0.51801833]]
```

❖ 행렬 연산 함수

- `tf.matmul` : 두 텐서 행렬곱 연산(입력 : 2개, hidden node : 3개)

$X(1,2)$

x1	x2
----	----



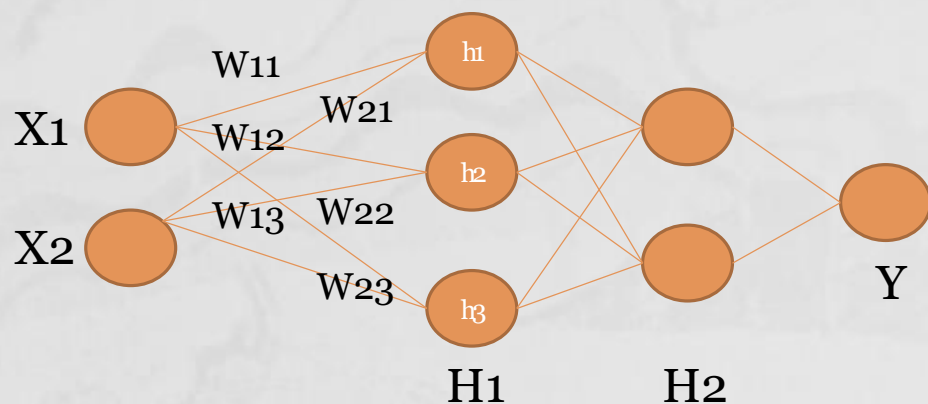
$W(2,3)$

w11	w12	w13
w21	w22	w23



$H1(3,1)$

$x1w11+x2w21$	=h1
$x1w12+x2w22$	=h2
$x1w13+x2w23$	=h3



❖ 텐서 플로우 행렬 연산

```
import tensorflow as tf
input_data = [ [1,1,1],[2,2,2] ]
x = tf.placeholder(dtype=tf.float32,shape=[2, 3])
w = tf.Variable([[2],[2],[2]],dtype=tf.float32)
b = tf.Variable([4],dtype=tf.float32)
y = tf.matmul(x,w)+b
print(x.get_shape())

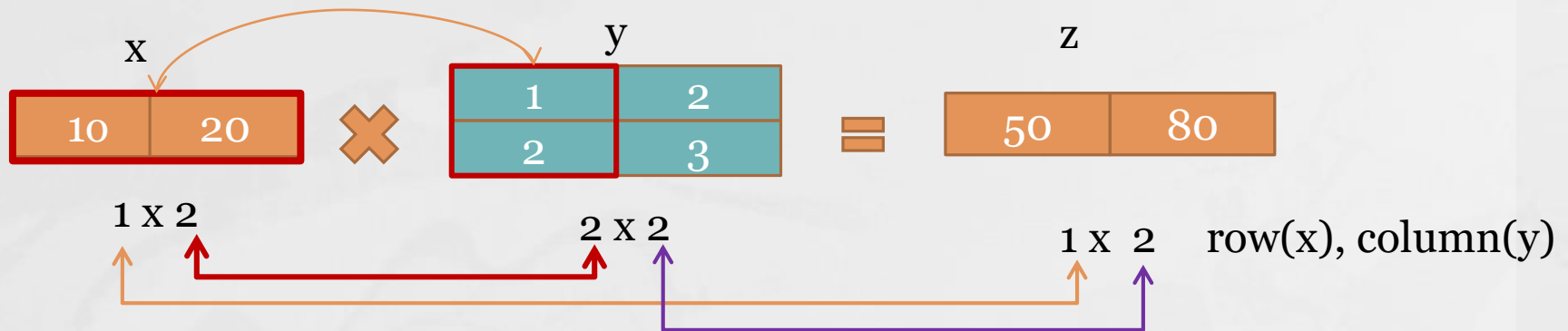
sess = tf.Session()
init = tf.global_variables_initializer()
sess.run(init)
result = sess.run(y,feed_dict={x:input_data})
print(result)
```

❖ 텐서 플로우 행렬 실습

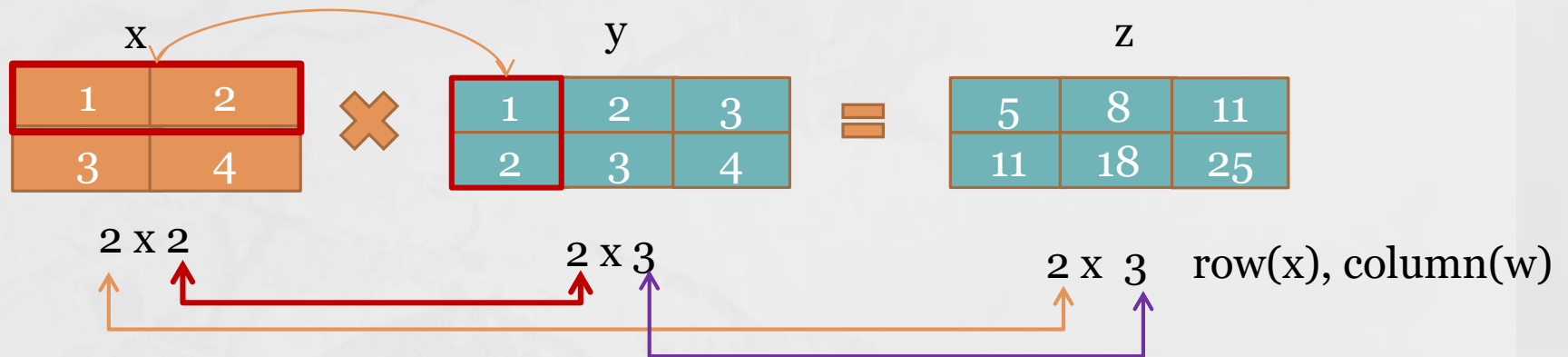
$$\begin{bmatrix} 1. & 2. & 3. \\ 2. & 3. & 4. \end{bmatrix}$$
$$\begin{bmatrix} 0. & 1. \\ 1. & 2. \\ 2. & 3. \end{bmatrix}$$
$$\begin{bmatrix} 8. & 14. \\ 11. & 20. \end{bmatrix}$$

x, y 행렬 곱

1. $x(1, 2) * y(2, 2) = z(1, 2) \rightarrow \text{row}(x), \text{column}(y)$

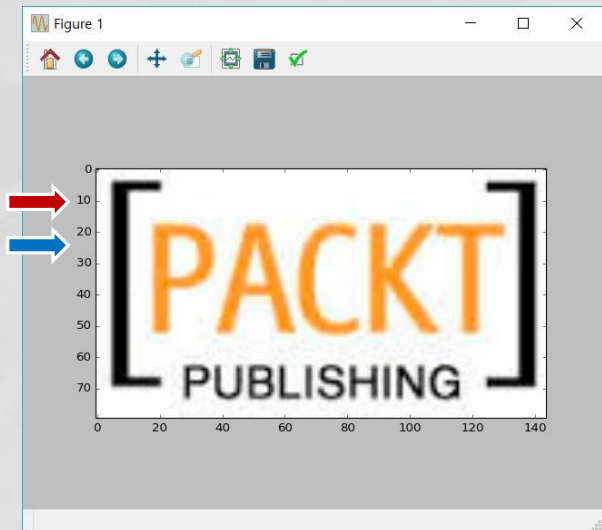


2. $x(2, 2) * y(2, 3) = z(2, 3) \rightarrow \text{row}(x), \text{column}(y)$



3. Tensor 다루기

(80, 144, 3)



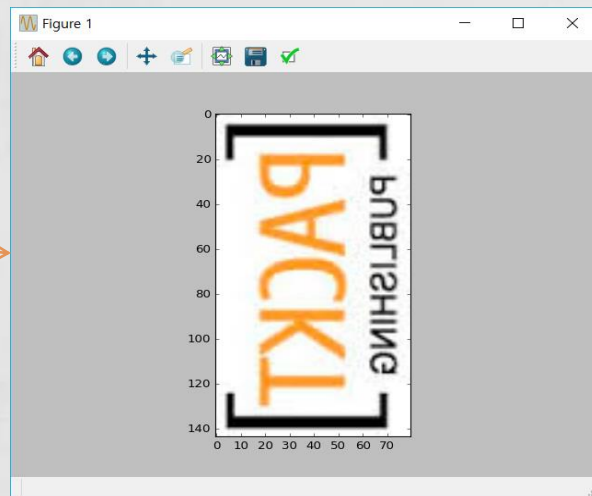
(16, 144, 3)



자르기

$[10, 0, 0], [16, -1, -1]$

$\text{perm}=[1, 0, 2]$



축 변경