Chapter 2 - Tensor Flow

Ex2: Operations

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
from __future__ import print_function
import tensorflow as tf
import numpy as np
```

Arithmetic Operators

- Làm quen cách lập trình có Arithmetic Operators bằng TensorFlow

Ghi chú: Cách đặt tên biến

- _x, _y, _z, ...: NumPy 0-d or 1-d arrays
- _X, _Y, _Z, ...: NumPy 2-d or higer dimensional arrays
- x, y, z, ...: 0-d or 1-d tensors
- X, Y, Z, ...: 2-d or higher dimensional tensors

Cộng

In [3]:
```
_x = np.array([1, 2, 3])
_y = np.array([-1, -2, -3])

x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)

out1 = tf.add(x, y)
out2 = x + y

tf.print("Tensor Add:", out1)
_out = np.add(_x, _y)
tf.print("Numpy Add:", _out)
```

```
Tensor Add: [0 0 0]
Numpy Add: array([0, 0, 0])
```

Trừ

In [4]:
```
_x = np.array([3, 4, 5])
_y = np.array(3)
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)
out1 = tf.subtract(x, y)
out2 = x - y
tf.print("Tensor Sub:", out1)
_out = np.subtract(_x, _y)
tf.print("Numpy Sub:", _out)
```

```
Tensor Sub: [0 1 2]
Numpy Sub: array([0, 1, 2])
```

Nhân

In [5]:
```
_x = np.array([3, 4, 5])
_y = np.array([1, 0, -1])
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)
out1 = tf.multiply(x, y)
out2 = x * y
tf.print("Tensor Mul:", out1)
_out = np.multiply(_x, _y)
tf.print("Numpy Mul:", _out)
```

```
Tensor Mul: [3 0 -5]
Numpy Mul: array([ 3,  0, -5])
```

Chia

In [6]:
```
_x = np.array([10, 20, 30], np.int32)
_y = np.array([2, 3, 5], np.int32)
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)
out1 = tf.divide(x, y)
out2 = tf.truediv(x, y)
tf.print("Tensor Divide:", out1)
tf.print("Tensor True Div:", out2)
```

```
Tensor Divide: [5 6.666666666666667 6]
Tensor True Div: [5 6.666666666666667 6]
```

In [7]:
```
_out1 = _x / _y
_out2 = _x // _y
tf.print("Numpy Div:", _out2)
tf.print("Numpy True Div:", _out1)
```

```
Numpy Div: array([5, 6, 6], dtype=int32)
Numpy True Div: array([5.        , 6.66666667, 6.        ])
```

Các hàm toán học cơ bản

In [8]:
```
_x = np.array([1, 2, 3], np.int32)
_y = np.array([4, 5, 6], np.int32)
_z = np.array([7, 8, 9], np.int32)
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)
z = tf.convert_to_tensor(_z)

out1 = tf.add_n([x, y, z])
out2 = x + y + z
tf.print(out1)
tf.print(out2)
```

```
[12 15 18]
[12 15 18]
```

In [9]:
```
_X = np.array([[1, -1], [3, -3]])
X = tf.convert_to_tensor(_X)

out = tf.abs(X)
tf.print(out)
```

```
[[1 1]
 [3 3]]
```

In [10]:
```
_x = np.array([1, -1])
x = tf.convert_to_tensor(_x)

out1 = tf.negative(x)
out2 = -x
tf.print(out1)
tf.print(out2)
```

```
[-1 1]
[-1 1]
```

In [11]:
```
_x = np.array([1, 3, 0, -1, -3])
x = tf.convert_to_tensor(_x)

out = tf.sign(x)
tf.print(out)

_out = np.sign(_x)
tf.print(_out)
```

```
[1 1 0 -1 -1]
array([ 1,  1,  0, -1, -1])
```

In [12]:
```
_x = np.array([1, 2, -1])
x = tf.convert_to_tensor(_x)

out1 = tf.square(x)
out2 = x * x
tf.print(out1)
_out = np.square(_x)
tf.print(_out)
```

```
[1 4 1]
array([1, 4, 1], dtype=int32)
```

In [13]:
```
_x = np.array([2.1, 1.5, 2.5, 2.9, -2.1, -2.5, -2.9])
x = tf.convert_to_tensor(_x)

out1 = tf.round(x)
out2 = tf.floor(x)
tf.print(out1)
tf.print(out2)
```

```
[2 2 2 ... -2 -2 -3]
[2 1 2 ... -3 -3 -3]
```

In [14]:
```
_x = np.array([1, 4, 9], dtype=np.float32)
x = tf.convert_to_tensor(_x)

out = tf.sqrt(x)
tf.print(out)
```

```
[1 2 3]
```

In [15]:
```
_x = np.array([[1, 2], [3, 4]])
_y = np.array([[1, 2], [1, 2]])
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)

out = tf.pow(x, y)
tf.print(out)
```

```
[[1 4]
 [3 16]]
```

In [16]:
```
_x = np.array([1., 2., 3.], np.float32)
x = tf.convert_to_tensor(_x)

out1 = tf.exp(x)
out2 = tf.pow(np.e, x) #np.e = 2.718281828459045
tf.print(out1)
```

```
[2.71828175 7.38905621 20.085537]
```

In [17]:
```
_x = np.array([2, 3, 4])
_y = np.array([1, 5, 2])
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)

out1 = tf.maximum(x, y)
out2 = tf.where(x > y, x, y)

tf.print(out1)
tf.print(out2)
```

```
[2 5 4]
[2 5 4]
```

In [18]:
```
_x = np.array([2, 3, 4])
_y = np.array([1, 5, 2])
x = tf.convert_to_tensor(_x)
y = tf.convert_to_tensor(_y)

out1 = tf.minimum(x, y)
out2 = tf.where(x < y, x, y)

tf.print(out1)
tf.print(out2)
```

```
[1 3 2]
[1 3 2]
```

In [19]:
```
### Matrix Math Function
```

In [20]:
```
_X = np.array([[1, 2, 3], [4, 5, 6]])
_Y = np.array([[1, 1], [2, 2], [3, 3]])
X = tf.convert_to_tensor(_X)
Y = tf.convert_to_tensor(_Y)

out = tf.matmul(X, Y)
tf.print(out)
```

```
[[14 14]
 [32 32]]
```

In [21]:
```
# Multiply X and Y. The first axis represents batches.
_X = np.arange(1, 13, dtype=np.int32).reshape((2, 2, 3))
_Y = np.arange(13, 25, dtype=np.int32).reshape((2, 3, 2))
display(_X, _Y)
```

```
array([[[ 1,  2,  3],
        [ 4,  5,  6]],

       [[ 7,  8,  9],
        [10, 11, 12]]])

array([[[13, 14],
        [15, 16],
        [17, 18]],

       [[19, 20],
        [21, 22],
        [23, 24]]])
```

In [22]:
```
X = tf.convert_to_tensor(_X)
Y = tf.convert_to_tensor(_Y)
out = tf.matmul(X, Y)
tf.print(out)
```

```
[[[94 100]
  [229 244]]

 [[508 532]
  [697 730]]]
```

In [23]:
```
out_re = tf.reduce_sum(out)
out_re_0 = tf.reduce_sum(out, 0)
out_re_1 = tf.reduce_sum(out, 1)
```

In [24]:
```
tf.print(out_re)
```

```
3134
```

In [25]:
```
tf.print(out_re_0)
```

```
[[602 632]
 [926 974]]
```

In [26]:
```
tf.print(out_re_1)
```

```
[[323 344]
 [1205 1262]]
```

In [27]:
```

```