모델

Part I. 자신만의 모델 생성

개요

- 자신만의 모델을 만드는 방법 (*오늘 강의)

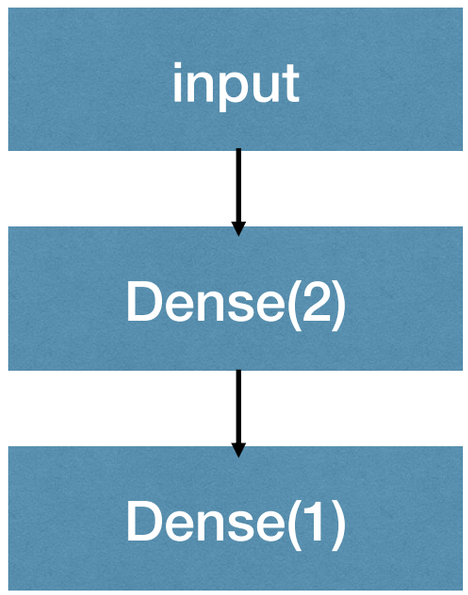
모델 생성

- 간단한 모델링을 Keras의 Sequential 모델을 통해서 학습
- 조금 더 다양한 모델을 다룰수 있는 기법을 다루고자 함

모델 생성 : Sequential

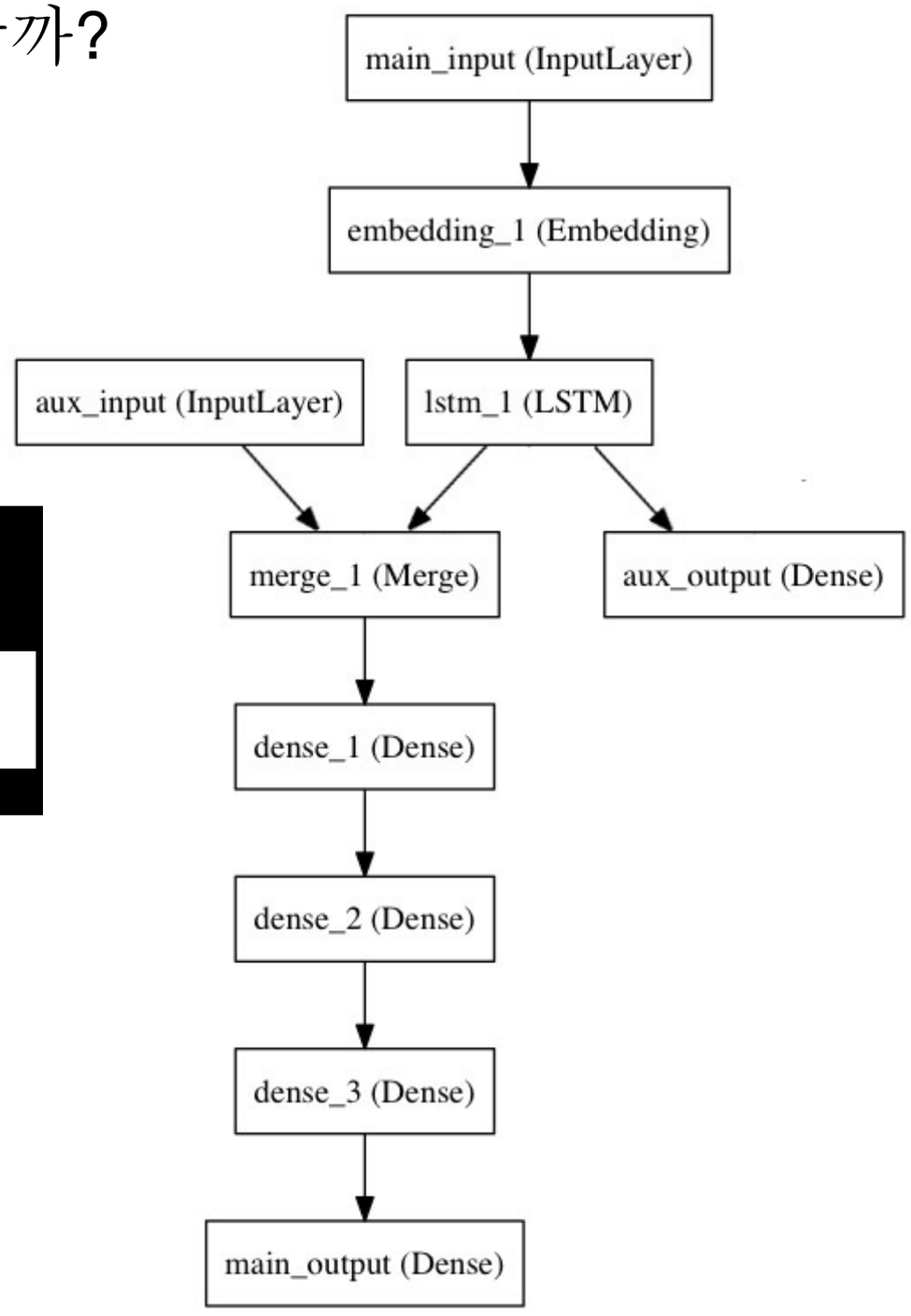
- 하나의 입력과 하나의 출력을 가지는 모델로, 순차적으로 Layer들을 쌓아나가는 방식

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add( Dense(2, input_dim = 1) )
model.add( Dense(1) )
```



모델 생성 : Functional

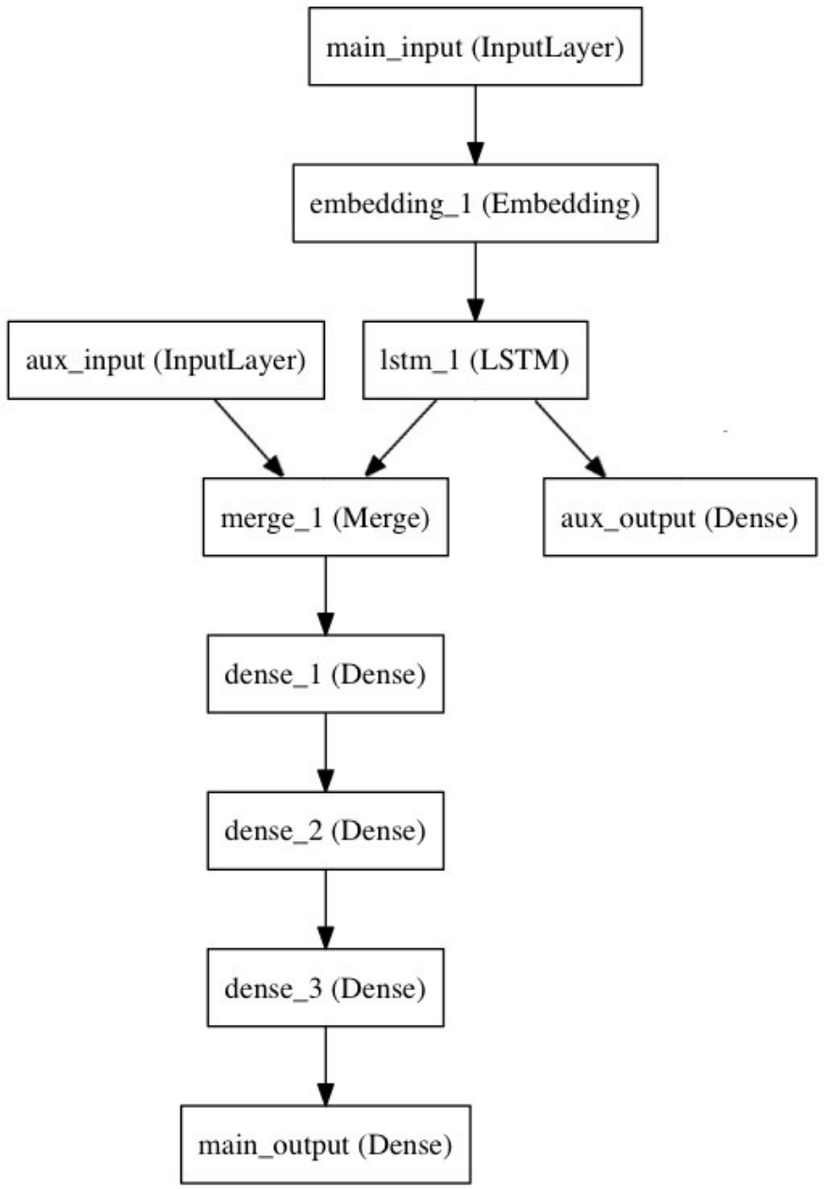
- 꼭 하나의 입력과 하나의 출력이어야만 할까?
- 두 개의 입력 하나의 출력은?
- 하나의 입력, 두 개의 출력은?




```
model = Model(inputs=[main_input, auxiliary_input],
              outputs=[main_output, auxiliary_output])
```

모델 생성 : Functional

- 꼭 하나의 입력과 하나의 출력이어야만 할까?
- 두 개의 입력 하나의 출력은?
- 하나의 입력, 두 개의 출력은?




```
model = Model(inputs=[main_input, auxiliary_input],
              outputs=[main_output, auxiliary_output])
```

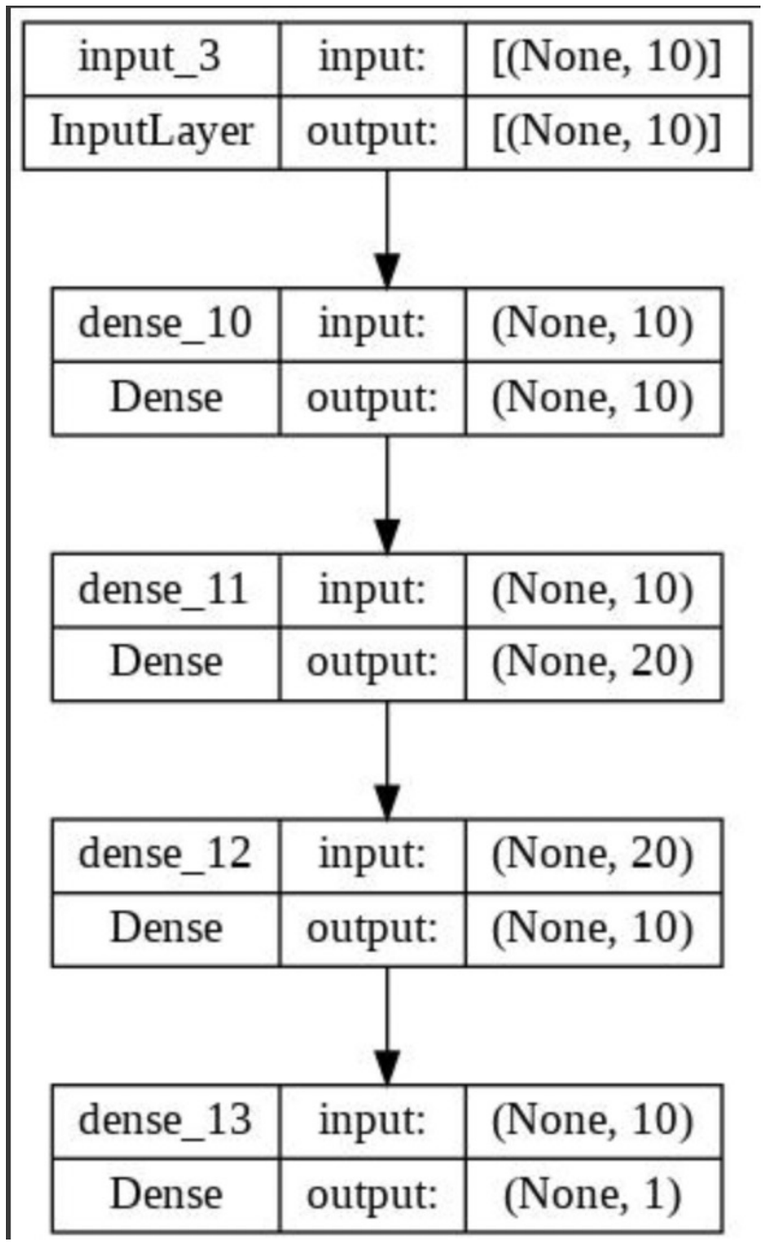
```
# And trained it via:
model.fit({'main_input': headline_data, 'aux_input': additional_data},
          {'main_output': headline_labels, 'aux_output': additional_labels},
          epochs=50, batch_size=32)
```

모델 생성 : Functional

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.utils import plot_model

visible = Input( shape=(10, ) )
hidden1 = Dense(10, activation='relu') (visible)
hidden2 = Dense(20, activation='relu') (hidden1)
hidden3 = Dense(10, activation='relu') (hidden2)
output  = Dense(1, activation='sigmoid') (hidden3)

model   = Model(inputs = [visible], outputs = [output])
```

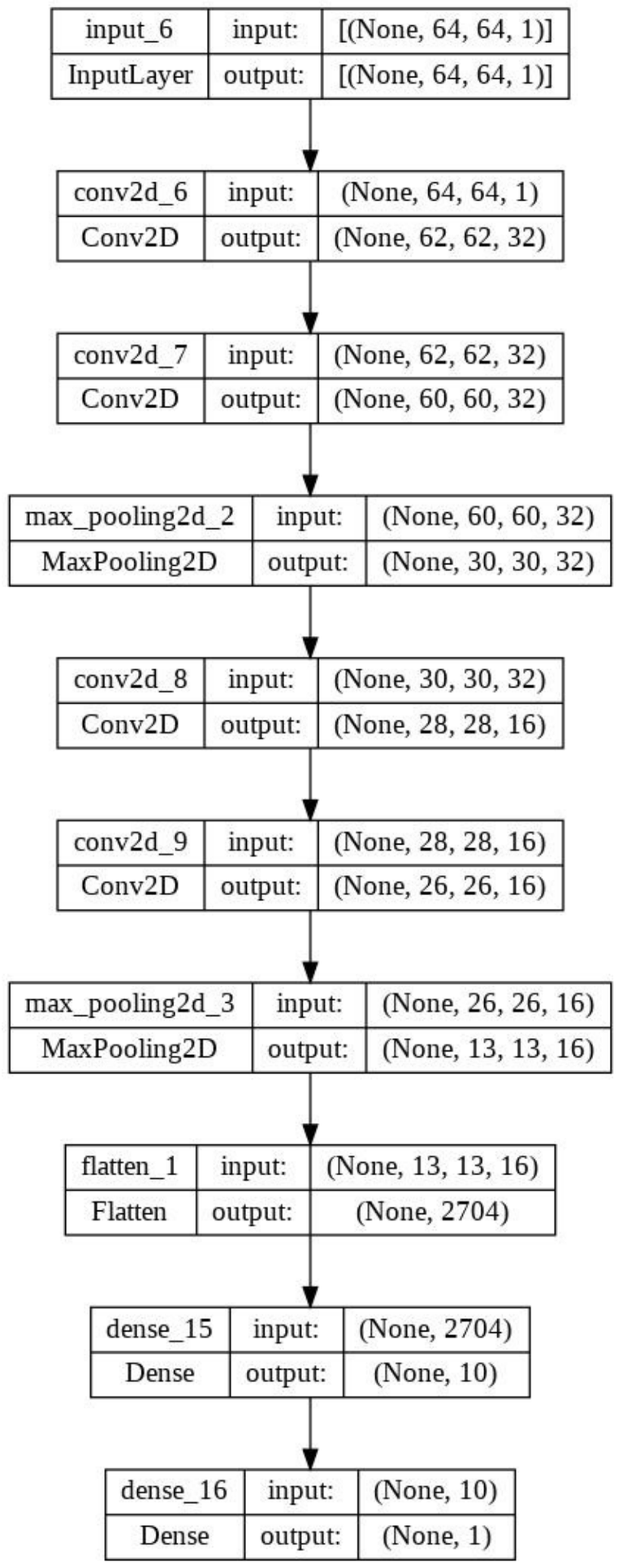
모델 출력

- `plot_model( model, to_file='model1.jpg', show_shapes=True)`



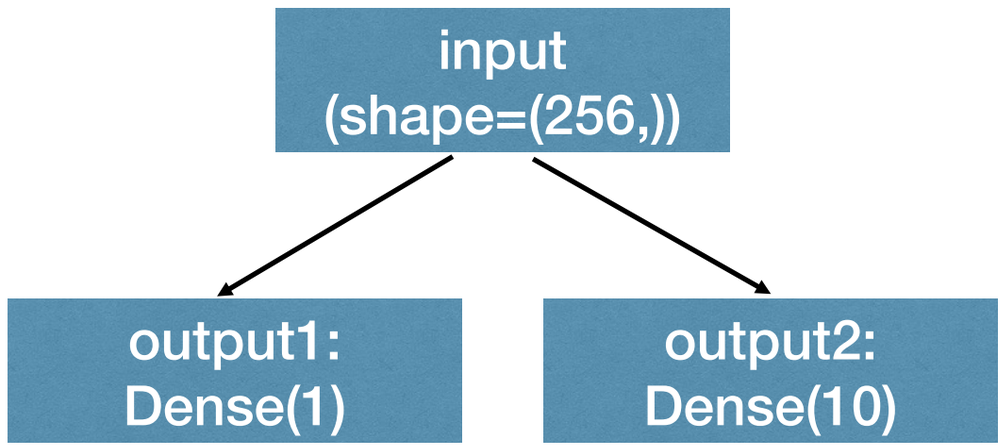
모델 생성 : Functional

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense, Flatten, Conv2D, MaxPooling2D
from tensorflow.keras.utils import plot_model

visible = Input( shape=(64,64,1) )
conv11 = Conv2D(32, kernel_size=(3,3), activation='relu') (visible)
conv12 = Conv2D(32, kernel_size=(3,3), activation='relu') (conv11)
pool1 = MaxPooling2D( pool_size=(2,2) ) (conv12)
conv21 = Conv2D(16, kernel_size=(3,3), activation='relu') (pool1)
conv22 = Conv2D(16, kernel_size=(3,3), activation='relu') (conv21)
pool2 = MaxPooling2D( pool_size=(2,2) ) (conv22)
flatten = Flatten() (pool2)
hidden1 = Dense (10, activation='relu') (flatten)
output = Dense(1, activation='sigmoid') (hidden1)

model   = Model(inputs = [visible], outputs = [output])
```


input_6 | input: | [(None, 64, 64, 1)]
InputLayer | output: | [(None, 64, 64, 1)]
conv2d_6 | input: | (None, 64, 64, 1)
Conv2D | output: | (None, 62, 62, 32)
conv2d_7 | input: | (None, 62, 62, 32)
Conv2D | output: | (None, 60, 60, 32)
max_pooling2d_2 | input: | (None, 60, 60, 32)
MaxPooling2D | output: | (None, 30, 30, 32)
conv2d_8 | input: | (None, 30, 30, 32)
Conv2D | output: | (None, 28, 28, 16)
conv2d_9 | input: | (None, 28, 28, 16)
Conv2D | output: | (None, 26, 26, 16)
max_pooling2d_3 | input: | (None, 26, 26, 16)
MaxPooling2D | output: | (None, 13, 13, 16)
flatten_1 | input: | (None, 13, 13, 16)
Flatten | output: | (None, 2704)
dense_15 | input: | (None, 2704)
Dense | output: | (None, 10)
dense_16 | input: | (None, 10)
Dense | output: | (None, 1)

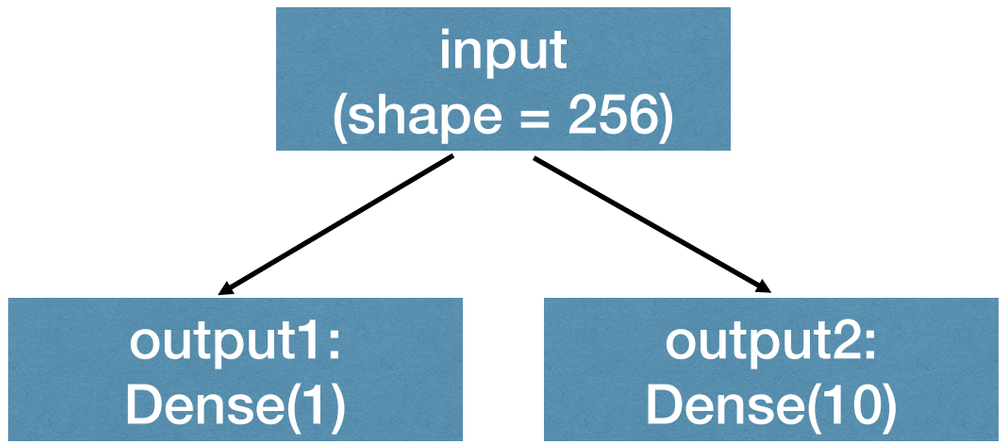
분기 모델

- 하나의 입력을 두 개의 출력으로 분기하는 경우



분기 모델

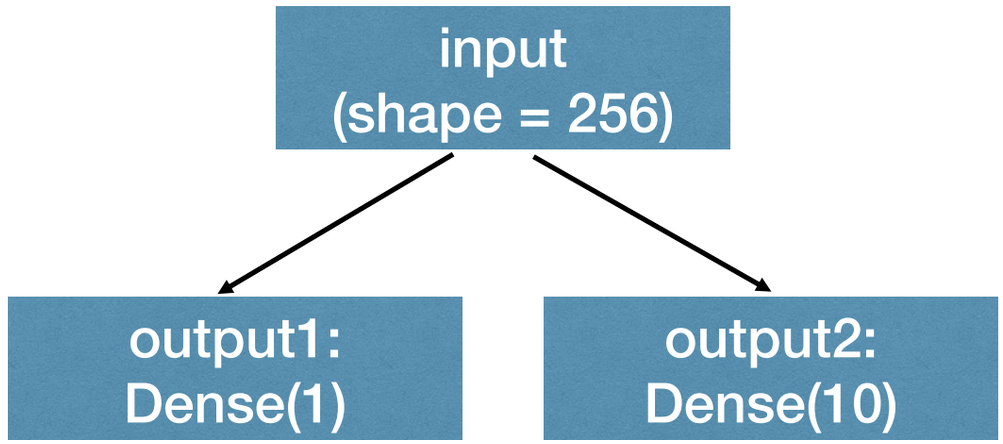
- 하나의 입력을 두 개의 출력으로 분기




```
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
from tensorflow.keras.utils import plot_model

input1  =
output1 =
output2 =
model1 = Model(inputs=[input1], outputs=[output1, output2])
model1.summary()
plot_model( model1, to_file='model1.jpg', show_shapes=True)
```

분기 모델

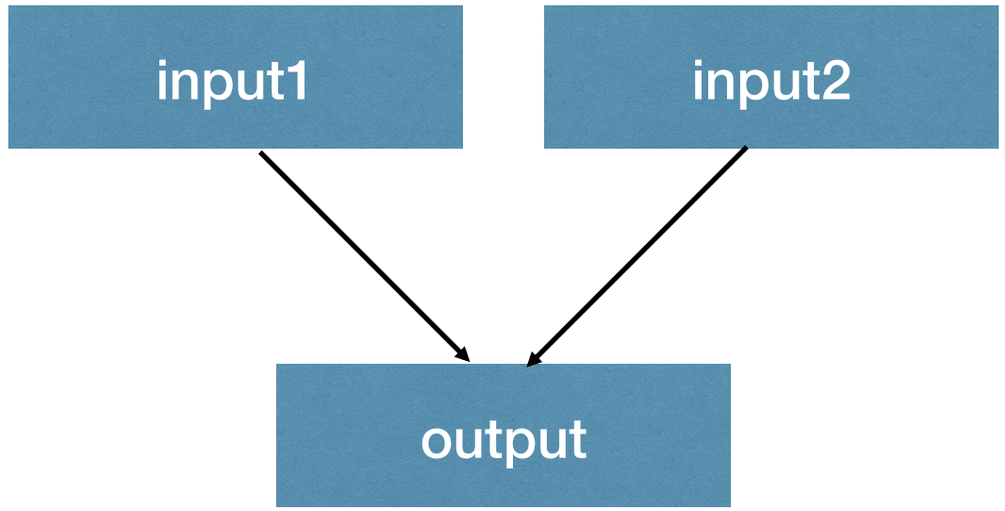
- 하나의 입력을 두 개의 출력으로 분기




```
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
from tensorflow.keras.utils import plot_model

input1 = Input( shape=(256,))
output1 = Dense(1, activation='sigmoid') (input1)
output2 = Dense(10, activation='softmax') (input1)
model1 = Model(inputs=[input1], outputs=[output1, output2])
model1.summary()
plot_model( model1, to_file='model1.jpg', show_shapes=True)
```

병합 모델

- 두개의 입력을 한 개의 출력으로 병합하는 경우



병합 모델

- 두개의 입력을 한 개의 출력으로 병합하는 경우

- 분기와 다르게 다양한 방식의 병합이 가능

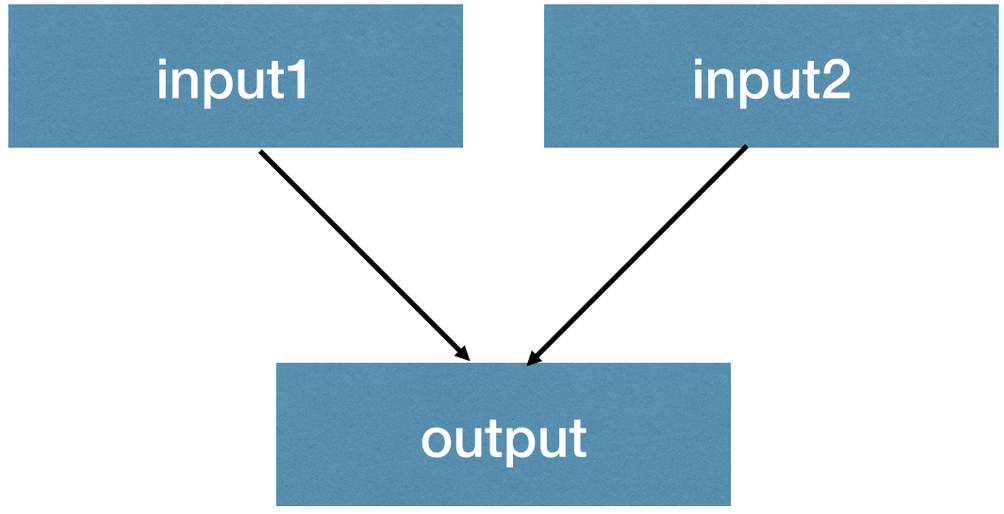
  입력들을 그대로 붙이는 경우

  입력들을 더하는 경우

  입력들을 빼는 경우

  입력들을 곱하는 경우

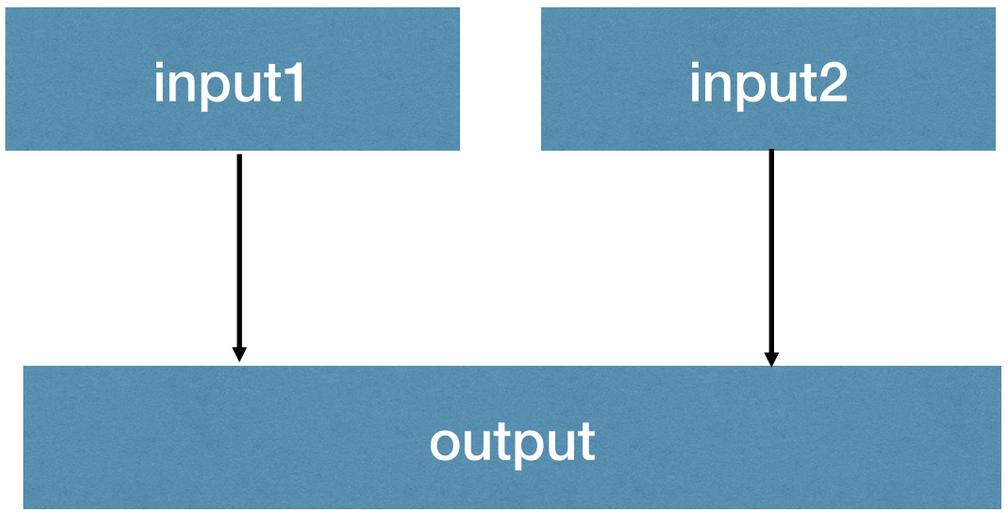
  입력들을 내적하는 경우 등



병합 모델 (붙이기)

- 두개의 입력을 한 개의 출력으로 병합하는 경우

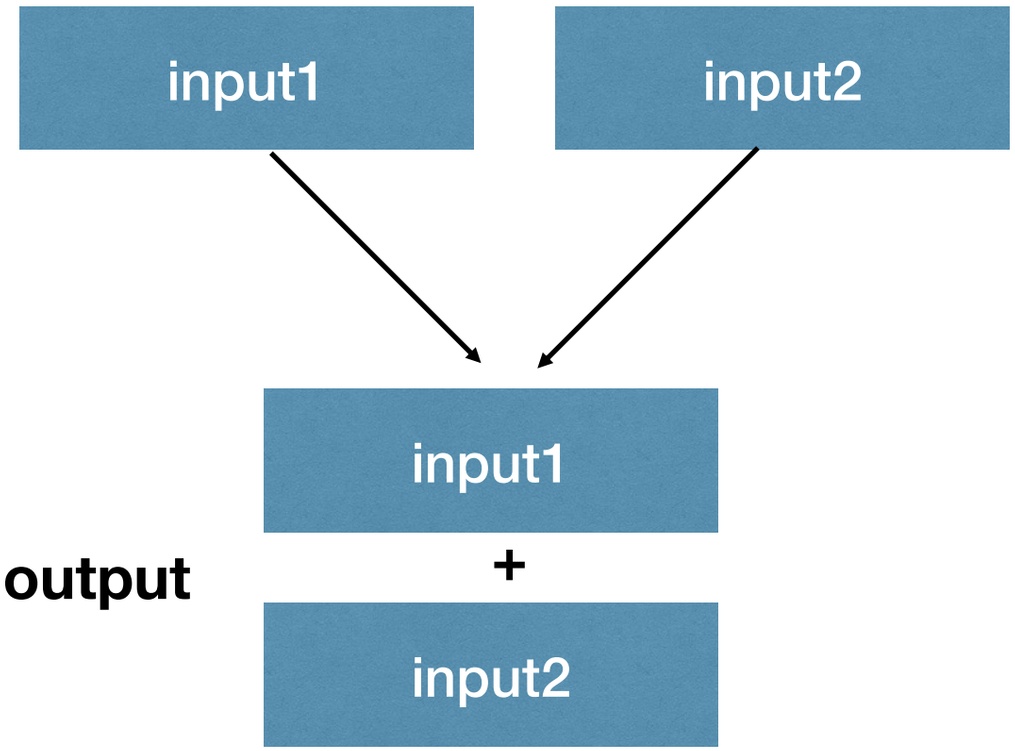
```
from tensorflow.keras.layers import Input, Concatenate
from tensorflow.keras.models import Model

input1 = Input(shape=(128,))
input2 = Input(shape=(256,))

output = Concatenate() ([input1, input2])

model2 = Model(inputs=[input1, input2], outputs=[output])

model2.summary()

plot_model( model2, to_file='model2.jpg', show_shapes=True)
```



병합 모델 (더하기)

- 두개의 입력을 한 개의 출력으로 병합하는 경우

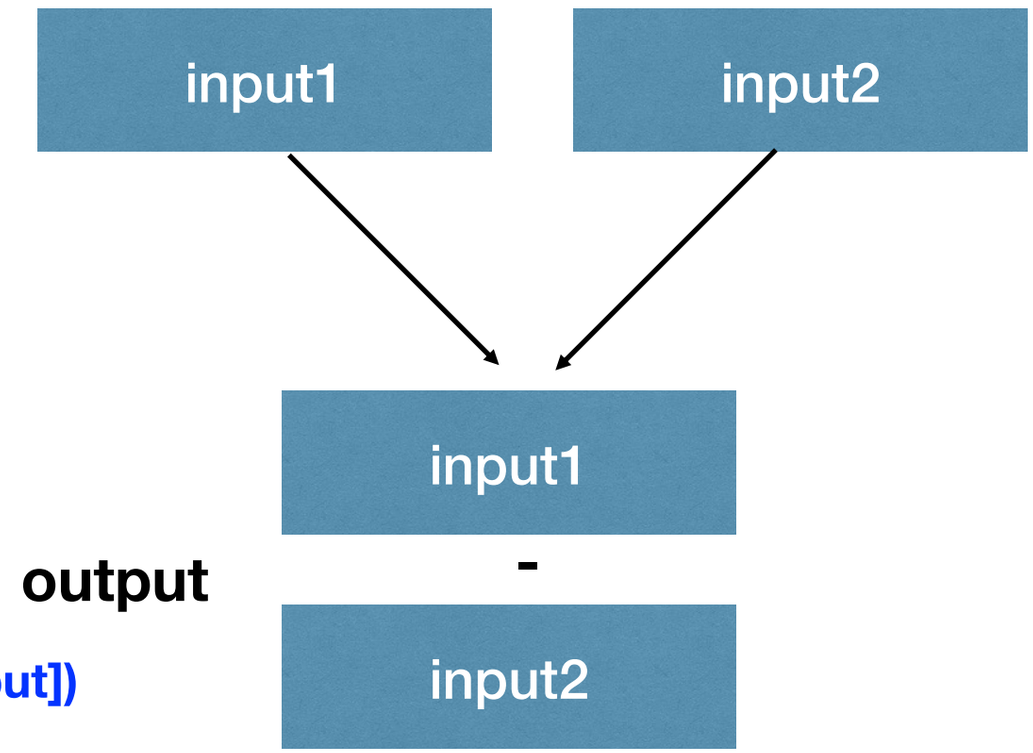
```
from tensorflow.keras.layers import Input, Add
from tensorflow.keras.models import Model

input1 = Input(shape=(128,))
input2 = Input(shape=(128,))

output = Add() ([input1, input2])

model2 = Model(inputs=[input1, input2], outputs=[output])

model2.summary()

plot_model( model2, to_file='model2.jpg', show_shapes=True)
```



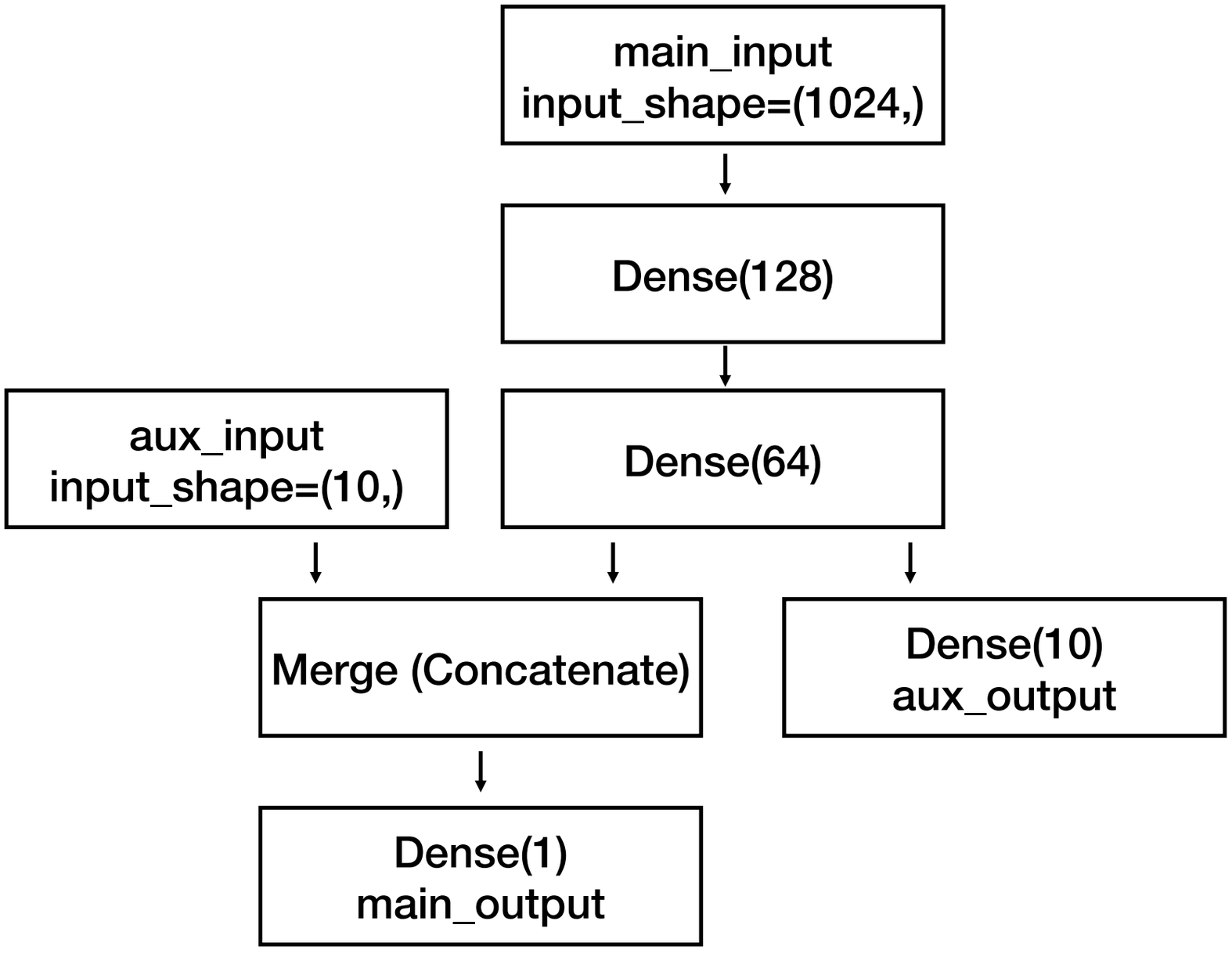
병합 모델 (빼기)

- 두개의 입력을 한 개의 출력으로 병합하는 경우

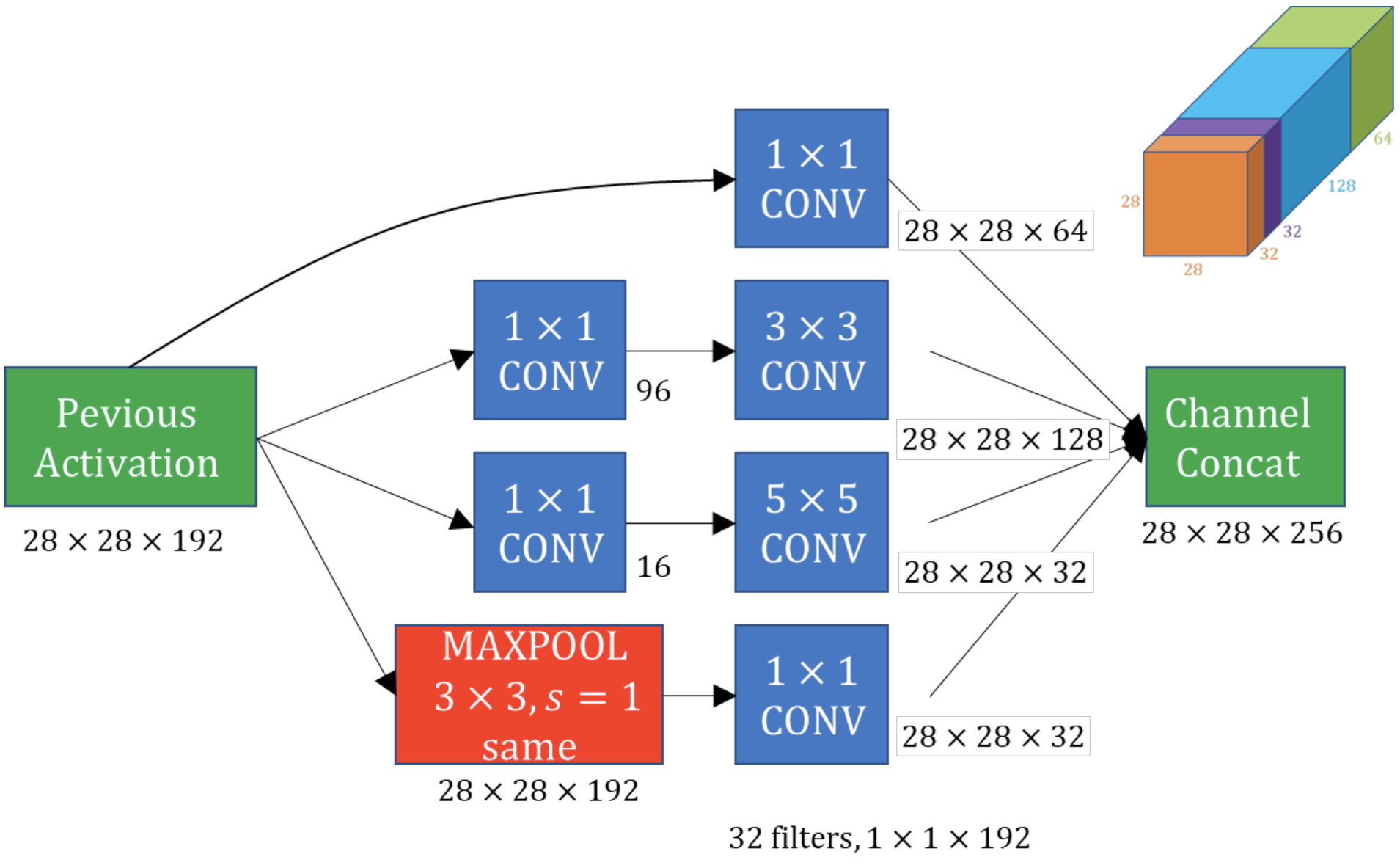
```
from tensorflow.keras.layers import Input, Subtract
from tensorflow.keras.models import Model

input1 = Input(shape=(128,))
input2 = Input(shape=(128,))

output = Subtract() ([input1, input2])

model2 = Model(inputs=[input1, input2], outputs=[output])

model2.summary()

plot_model( model2, to_file='model2.jpg', show_shapes=True)
```



실습


main_input
input_shape=(1024,)
Dense(128)
aux_input
input_shape=(10,)
Dense(64)
Merge (Concatenate)
Dense(10)
aux_output
Dense(1)
main_output

Inception Module



Residual Module

Input shape = (32, 32, 256)

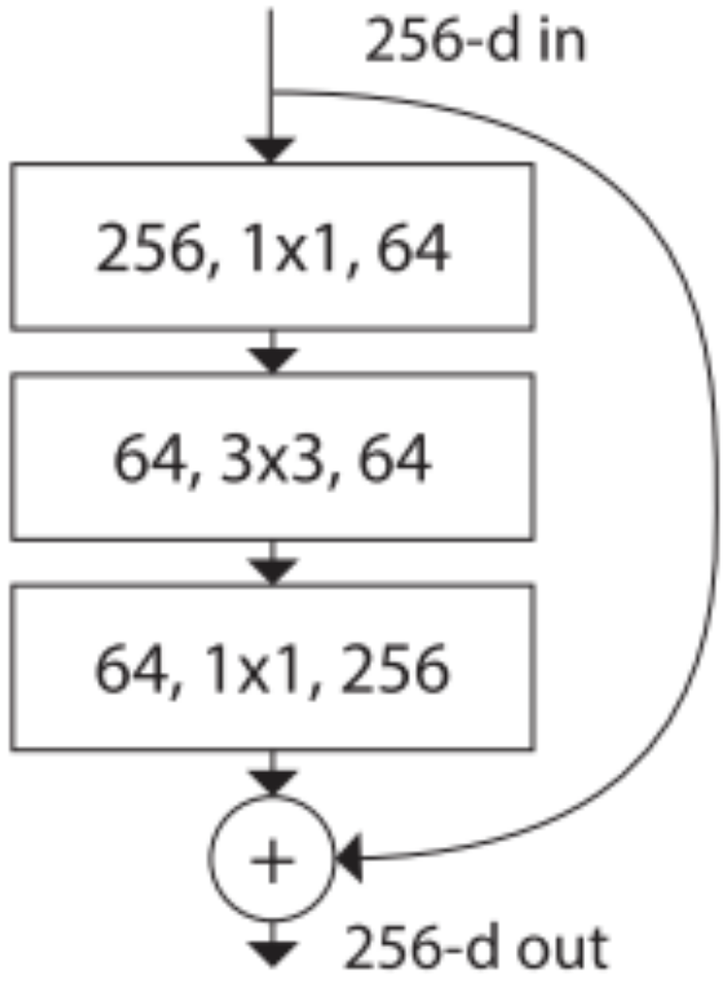
Input, Conv, output
256, 1x1, 64

Input, Conv, output
64, 3x3, 64

Input, Conv, output
64, 1x1, 256

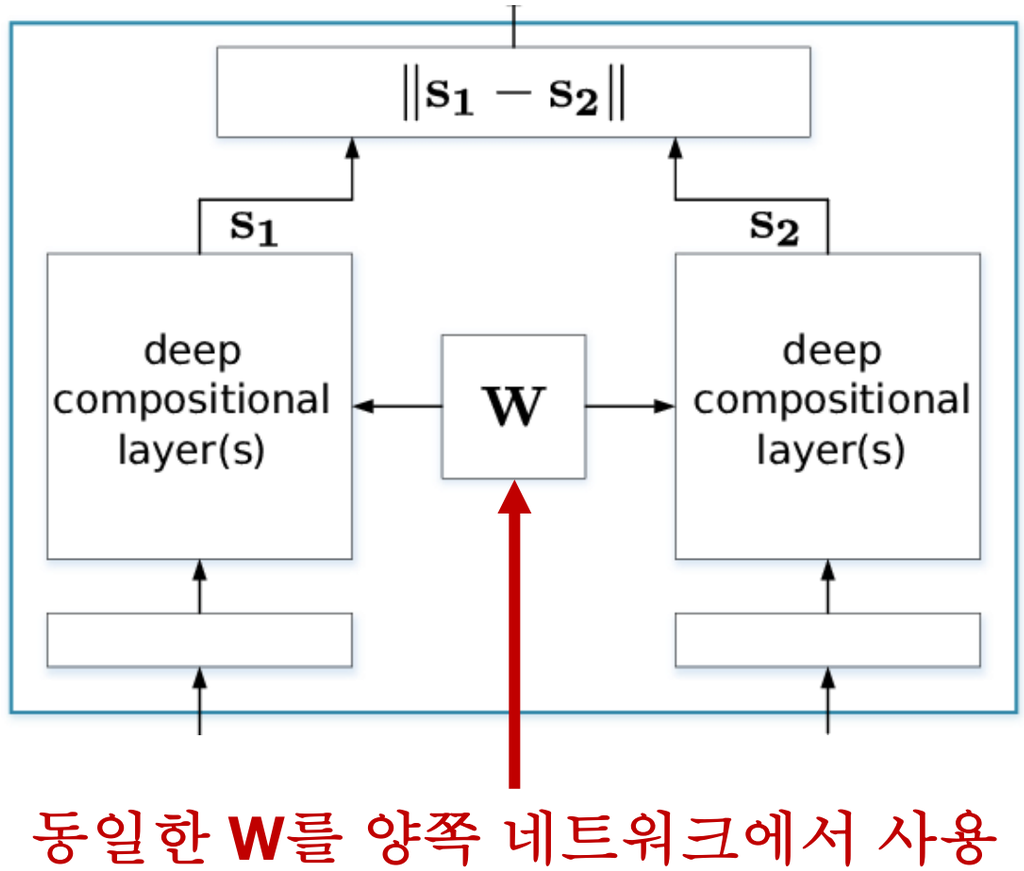
Add()

output shape = (32, 32, 256)



Siamese Network

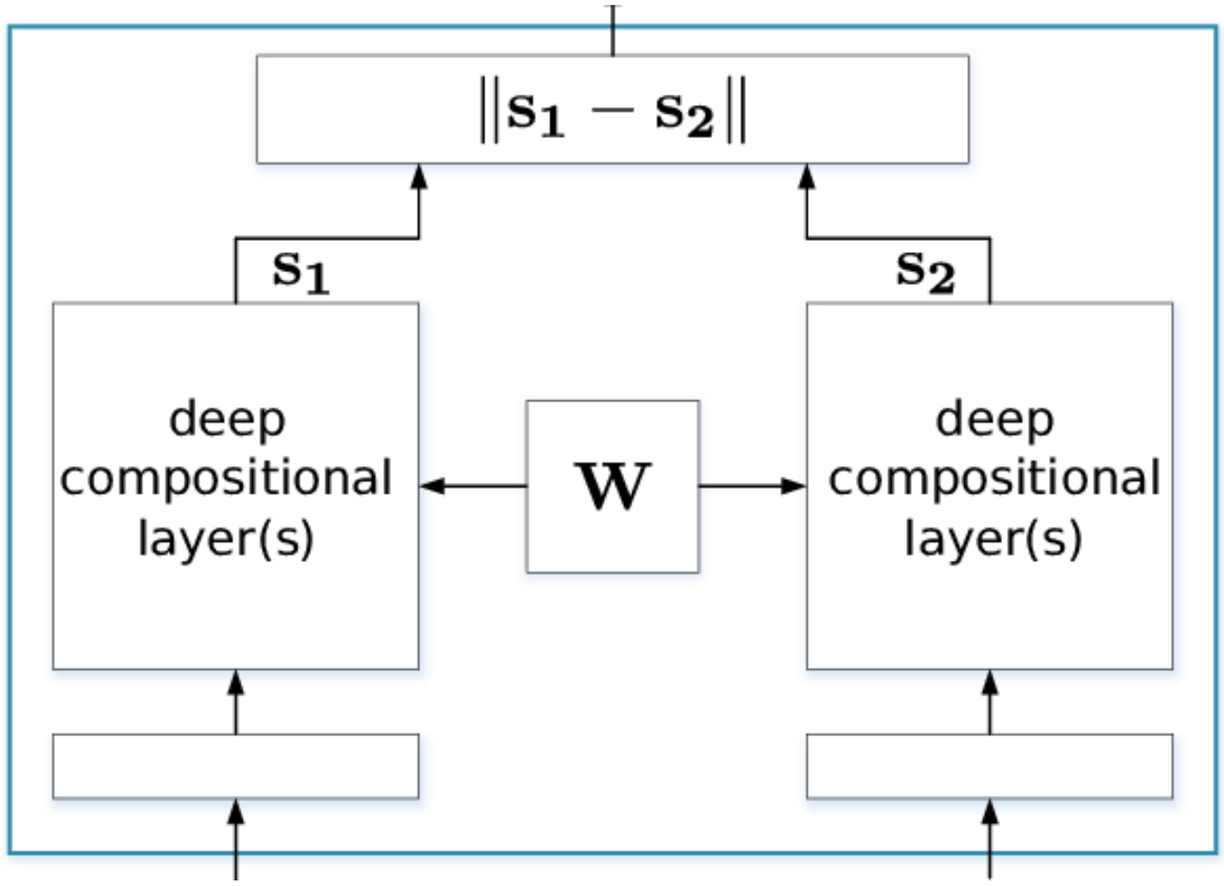
- weight가 동일한 두 개의 블록은 어떻게 구현이 가능할까?



Siamese Network

- weight가 동일한 두 개의 블록은 어떻게 구현이 가능할까?

```
a = Input(shape=(32, 32, 3))

b = Input(shape=(32, 32, 3))


conv = Conv2D(16, (3, 3))


s1 = conv(a)

s2 = conv(b)
```



Advanced

https://keras.io/guides/functional_api/

https://keras.io/guides/functional_api/#shared-layers

https://www.tensorflow.org/guide/keras/functional?hl=ko