機器學習概論作業

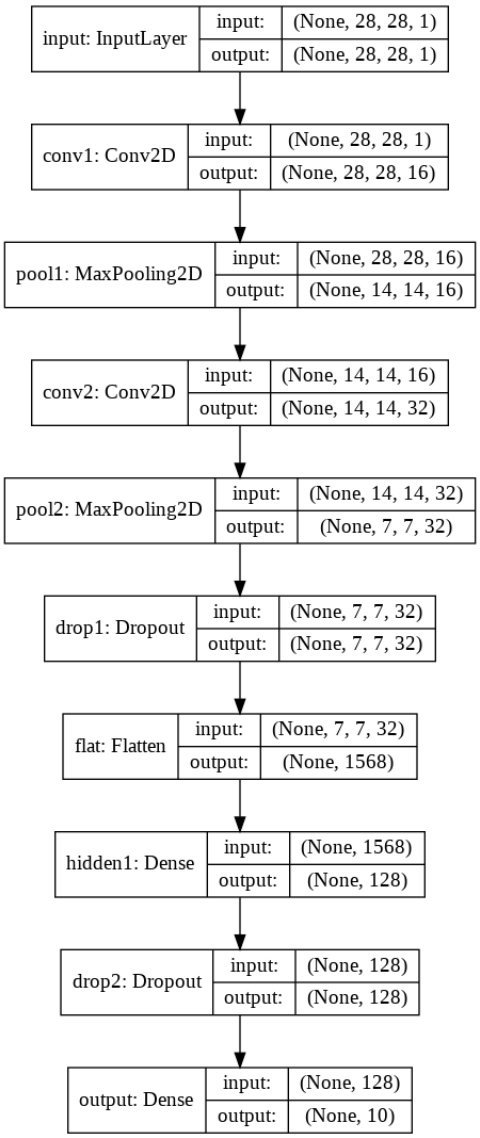
範圍： Functional API

銘傳大學電腦與通訊工程系

班　　級	電通三乙
姓　　名	李柏賢
學　　號	07050862
作業成果	應繳作業共 3 題，前 2 題每題 30 分，第 3 題 40 分。 我共完成 3 題，應得 100 分
授課教師	陳慶逸

■ 請確實填寫自己寫完成題數，填寫不實者(如上傳與作業明顯無關的答案，或是計算題數有誤者)，本次作業先扣 50 分。

一、　使用 Functional API 建立手寫數字辨識的 CNN 模型:



程式碼:

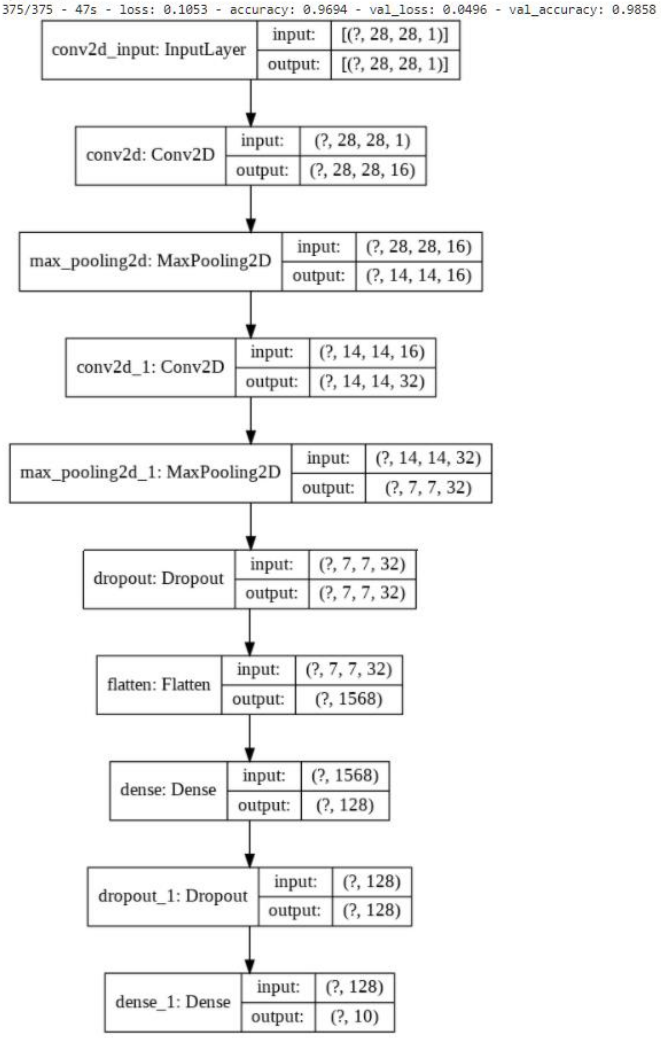
```
import numpy as np
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Dropout
from keras.utils import to_categorical
```

```
# 指定亂數種子
seed = 7
np.random.seed(seed)
# 載入資料集
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()
# 將圖片轉換成 4D 張量
X_train = X_train.reshape(X_train.shape[0], 28, 28, 1).astype("f
loat32")
X_test = X_test.reshape(X_test.shape[0], 28, 28, 1).astype("floa
t32")
# 因為是固定範圍，所以執行正規化，從 0-255 至 0-1
X_train = X_train / 255
X_test = X_test / 255
# One-hot 編碼
Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)
# 定義模型
model = Sequential()
model.add(Conv2D(16, kernel_size=(5, 5), padding="same",input_sh
ape=(28, 28, 1), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, kernel_size=(5, 5), padding="same",activati
on="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(128, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(10, activation="softmax"))
model.summary() # 顯示模型摘要資訊
# 編譯模型
model.compile(loss="categorical_crossentropy", optimizer="adam",
metrics=["accuracy"])
# 訓練模型
history = model.fit(X_train, Y_train, validation_split=0.2,
epochs=3, batch_size=128, verbose=2)
from keras.utils import plot_model
```

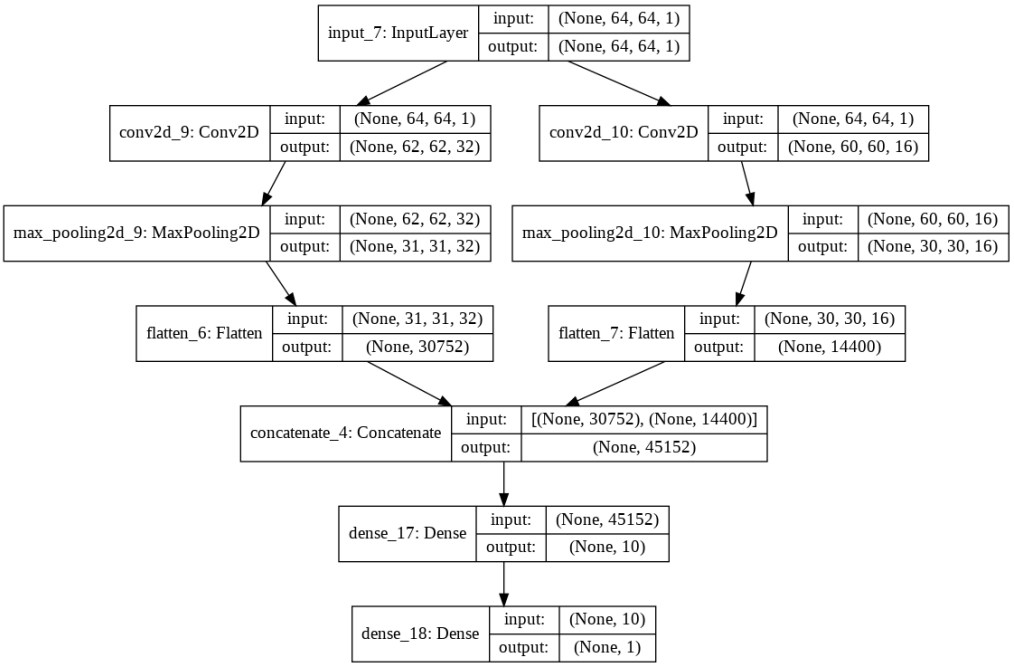
```
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

輸出結果擷圖:

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 16)        416
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 14, 14, 32)        12832
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 32)          0
_________________________________________________________________
dropout (Dropout)            (None, 7, 7, 32)          0
_________________________________________________________________
flatten (Flatten)            (None, 1568)              0
_________________________________________________________________
dense (Dense)                (None, 128)               200832
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 215,370
Trainable params: 215,370
Non-trainable params: 0
_________________________________________________________________
Epoch 1/3
375/375 - 48s - loss: 0.4185 - accuracy: 0.8670 - val_loss: 0.0876 - val_accuracy: 0.9740
Epoch 2/3
375/375 - 47s - loss: 0.1382 - accuracy: 0.9582 - val_loss: 0.0582 - val_accuracy: 0.9829
Epoch 3/3
375/375 - 47s - loss: 0.1053 - accuracy: 0.9694 - val_loss: 0.0496 - val_accuracy: 0.9858
```



二、　建立兩組卷積層和池化層來共享一個輸入層:



程式碼:

```
# Shared Input Layer
from keras.utils import plot_model
from keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate
# input layer
visible = Input(shape=(64,64,1))
# first feature extractor
conv1 = Conv2D(32, kernel_size=3, activation='relu')(visible)
pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
flat1 = Flatten()(pool1)
# second feature extractor
conv2 = Conv2D(16, kernel_size=5, activation='relu')(visible)
pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
flat2 = Flatten()(pool2)
```

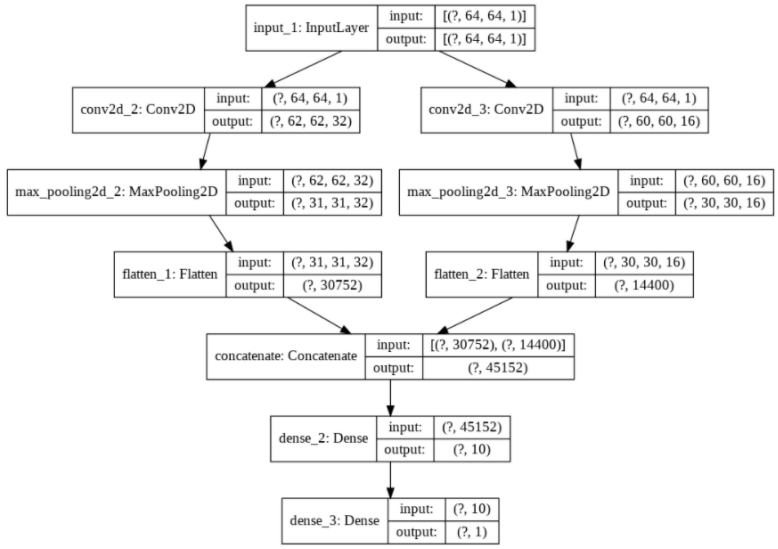
```
# merge feature extractors
merge = concatenate([flat1, flat2])
# interpretation layer
hidden1 = Dense(10, activation='relu')(merge)
# prediction output
output = Dense(1, activation='sigmoid')(hidden1)
model = Model(inputs=visible, outputs=output)
# summarize layers
print(model.summary())
# plot graph
plot_model(model, to_file='shared_input_layer.png')
#繪出模型圖片
from keras.utils import plot_model
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

輸出結果擷圖:

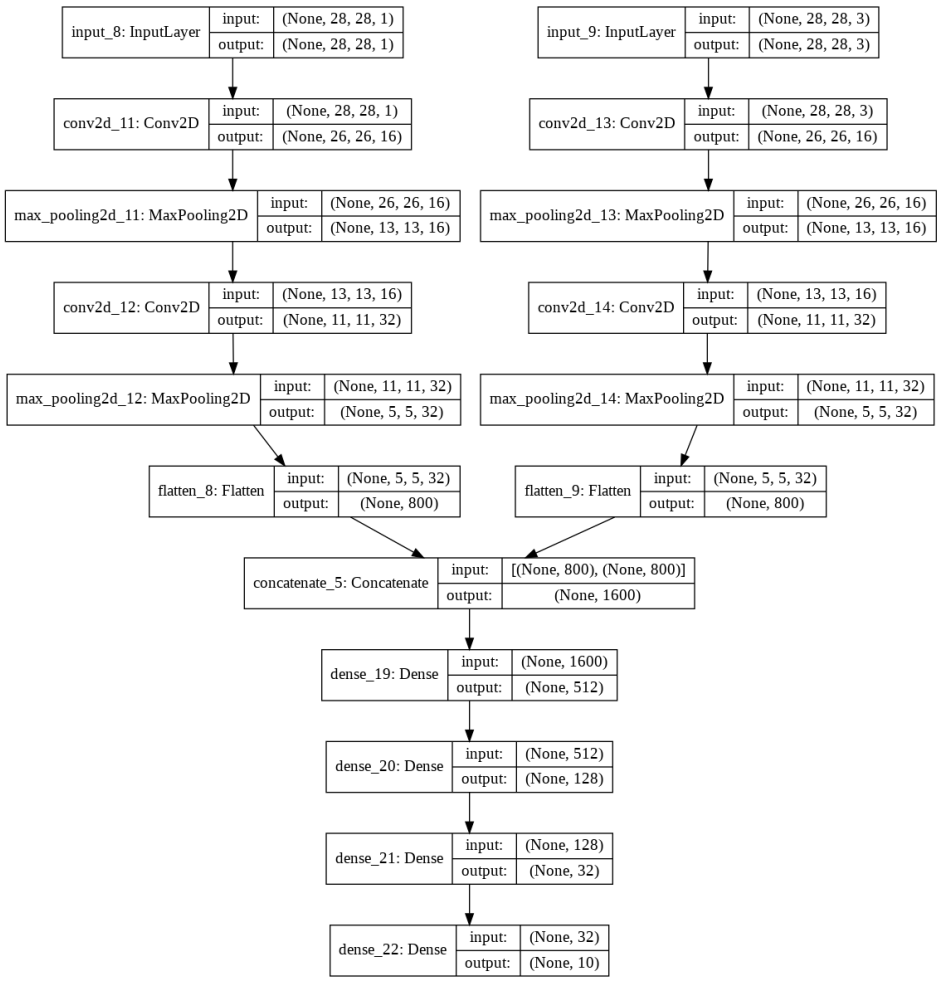
```
Model: "functional_1"
```

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 64, 64, 1)]	0	
conv2d_2 (Conv2D)	(None, 62, 62, 32)	320	input_1[0][0]
conv2d_3 (Conv2D)	(None, 60, 60, 16)	416	input_1[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 31, 31, 32)	0	conv2d_2[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 30, 30, 16)	0	conv2d_3[0][0]
flatten_1 (Flatten)	(None, 30752)	0	max_pooling2d_2[0][0]
flatten_2 (Flatten)	(None, 14400)	0	max_pooling2d_3[0][0]
concatenate (Concatenate)	(None, 45152)	0	flatten_1[0][0] flatten_2[0][0]
dense_2 (Dense)	(None, 10)	451530	concatenate[0][0]
dense_3 (Dense)	(None, 1)	11	dense_2[0][0]

```
Total params: 452,277
Trainable params: 452,277
Non-trainable params: 0
None
```



三、　多輸入多輸出模型: 模型有兩個輸入層，一個是灰階圖片(28x28x1)，另一個是彩色圖片(28x28x3)。



程式碼:

```
# Multiple Inputs
from keras.utils import plot_model
from keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate
# first input model
```

```
visible1 = Input(shape=(28,28,1))
conv11 = Conv2D(16, kernel_size=3, activation='relu')(visible1)
pool11 = MaxPooling2D(pool_size=(2, 2))(conv11)
conv12 = Conv2D(32, kernel_size=3, activation='relu')(pool11)
pool12 = MaxPooling2D(pool_size=(2, 2))(conv12)
flat1 = Flatten()(pool12)
# second input model
visible2 = Input(shape=(28,28,3))
conv21 = Conv2D(16, kernel_size=3, activation='relu')(visible2)
pool21 = MaxPooling2D(pool_size=(2, 2))(conv21)
conv22 = Conv2D(32, kernel_size=3, activation='relu')(pool21)
pool22 = MaxPooling2D(pool_size=(2, 2))(conv22)
flat2 = Flatten()(pool22)
# merge input models
merge = concatenate([flat1, flat2])
# interpretation model
hidden1 = Dense(512, activation='relu')(merge)
hidden2 = Dense(128, activation='relu')(hidden1)
hidden3 = Dense(32, activation='relu')(hidden2)
output = Dense(10, activation='sigmoid')(hidden3)
model = Model(inputs=[visible1, visible2], outputs=output)
# summarize layers
print(model.summary())
# plot graph
plot_model(model, to_file='multiple_inputs.png')
#繪出模型圖片
from keras.utils import plot_model
plot_model(model, to_file="Ch16_1.png", show_shapes=True)
```

輸出結果擷圖:

```
Model: "functional_3"
```

Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	[(None, 28, 28, 1)]	0	
input_3 (InputLayer)	[(None, 28, 28, 3)]	0	
conv2d_4 (Conv2D)	(None, 26, 26, 16)	160	input_2[0][0]
conv2d_6 (Conv2D)	(None, 26, 26, 16)	448	input_3[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 13, 13, 16)	0	conv2d_4[0][0]
max_pooling2d_6 (MaxPooling2D)	(None, 13, 13, 16)	0	conv2d_6[0][0]
conv2d_5 (Conv2D)	(None, 11, 11, 32)	4640	max_pooling2d_4[0][0]
conv2d_7 (Conv2D)	(None, 11, 11, 32)	4640	max_pooling2d_6[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 5, 5, 32)	0	conv2d_5[0][0]
max_pooling2d_7 (MaxPooling2D)	(None, 5, 5, 32)	0	conv2d_7[0][0]
flatten_3 (Flatten)	(None, 800)	0	max_pooling2d_5[0][0]
flatten_4 (Flatten)	(None, 800)	0	max_pooling2d_7[0][0]
concatenate_1 (Concatenate)	(None, 1600)	0	flatten_3[0][0] flatten_4[0][0]
dense_4 (Dense)	(None, 512)	819712	concatenate_1[0][0]
dense_5 (Dense)	(None, 128)	65664	dense_4[0][0]
dense_6 (Dense)	(None, 32)	4128	dense_5[0][0]
dense_7 (Dense)	(None, 10)	330	dense_6[0][0]

```
Total params: 899,722
Trainable params: 899,722
Non-trainable params: 0
```

None