

# 課堂活動與討論: CNN小專案 (2020/04/28)

曾宏鈞 06160485  
徐友笙 05360365  
蔡毓丞 06370136  
盧君彥 05360153

## 1. 上傳CNN實驗結果(每個同學都要上傳)

input圖片大小:64x64,1

```
Epoch 1/5
937/937 [=====] - 23s 25ms/step - loss: 0.1251 - acc: 0.9617 - val_loss: 0.0565 - val_ac
0.9811
Epoch 2/5
937/937 [=====] - 22s 23ms/step - loss: 0.0521 - acc: 0.9840 - val_loss: 0.0520 - val_ac
0.9832
Epoch 3/5
937/937 [=====] - 22s 23ms/step - loss: 0.0442 - acc: 0.9864 - val_loss: 0.0324 - val_ac
0.9897
Epoch 4/5
937/937 [=====] - 22s 24ms/step - loss: 0.0373 - acc: 0.9887 - val_loss: 0.0238 - val_ac
0.9919
Epoch 5/5
937/937 [=====] - 22s 24ms/step - loss: 0.0364 - acc: 0.9890 - val_loss: 0.0256 - val_ac
0.9913
10000/10000 [=====] - 1s 92us/step
```

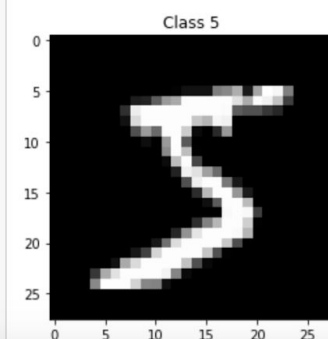
Test accuracy: 0.9914



All Conv with 5 feature

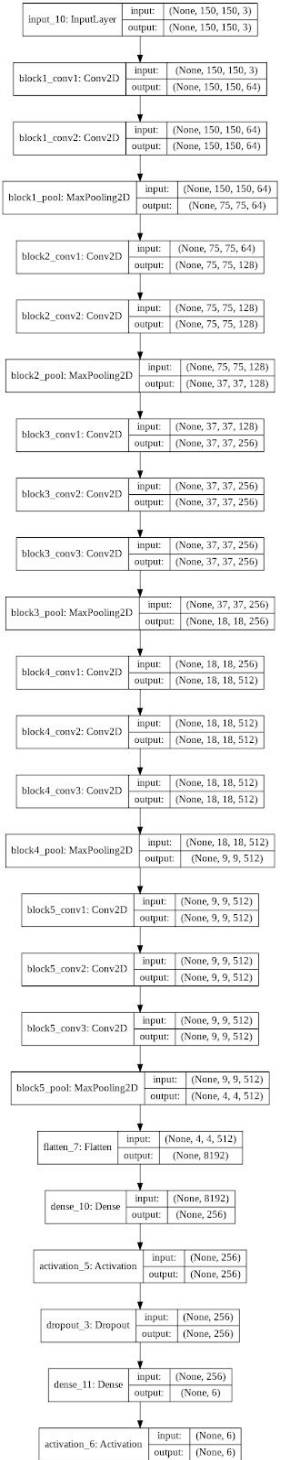
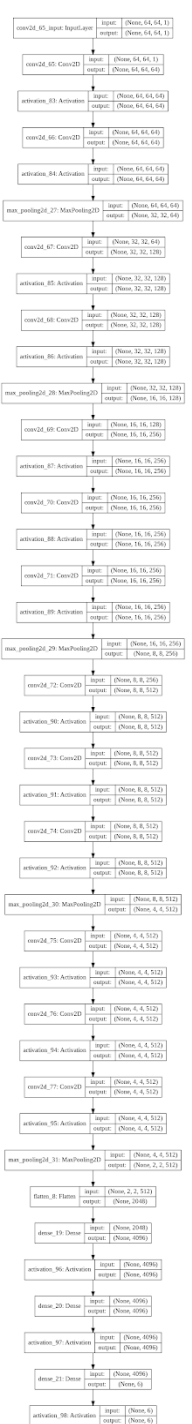
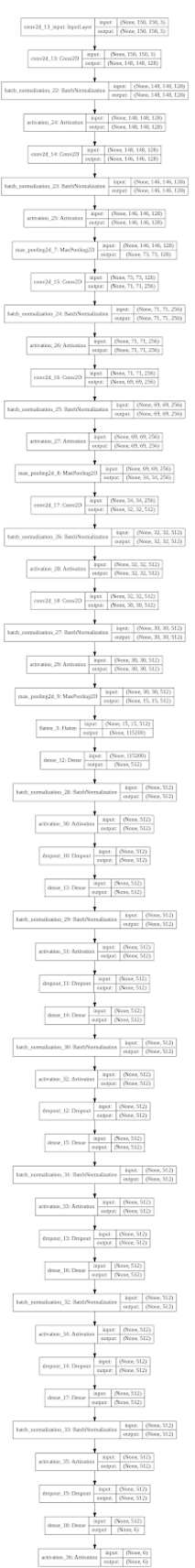
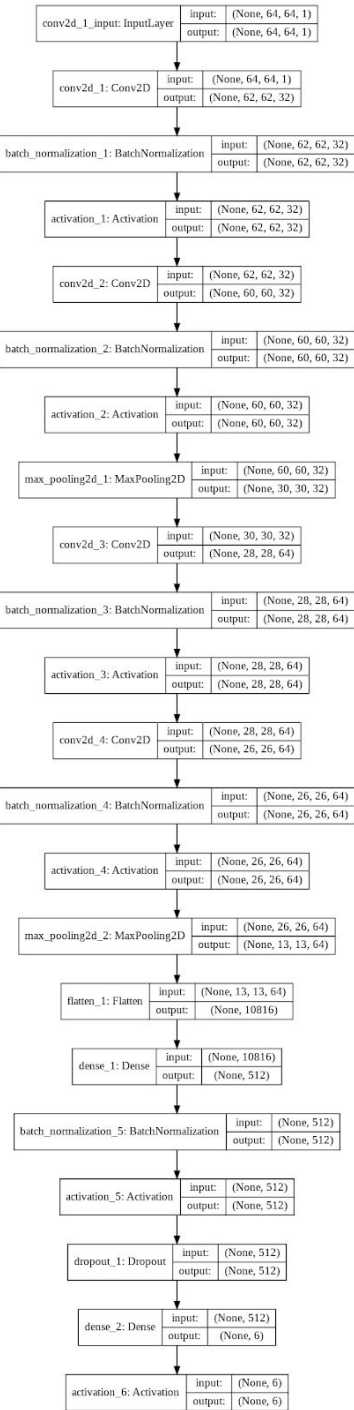
```
Epoch 1/5
937/937 [=====] - 23s 25ms/step - loss: 0.3190 - acc: 0.8985 - val_loss: 0.0934 - val_ac
0.9702
Epoch 2/5
937/937 [=====] - 21s 22ms/step - loss: 0.1268 - acc: 0.9617 - val_loss: 0.0710 - val_ac
0.9775
Epoch 3/5
937/937 [=====] - 21s 22ms/step - loss: 0.0992 - acc: 0.9693 - val_loss: 0.0491 - val_ac
0.9843
Epoch 4/5
937/937 [=====] - 20s 22ms/step - loss: 0.0867 - acc: 0.9723 - val_loss: 0.0469 - val_ac
0.9850
Epoch 5/5
937/937 [=====] - 20s 22ms/step - loss: 0.0771 - acc: 0.9755 - val_loss: 0.0499 - val_ac
0.9839
10000/10000 [=====] - 1s 100us/step
```

Test accuracy: 0.9841

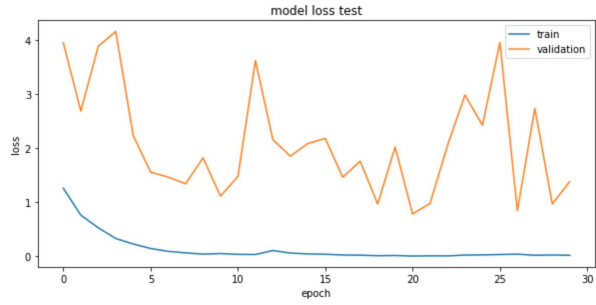
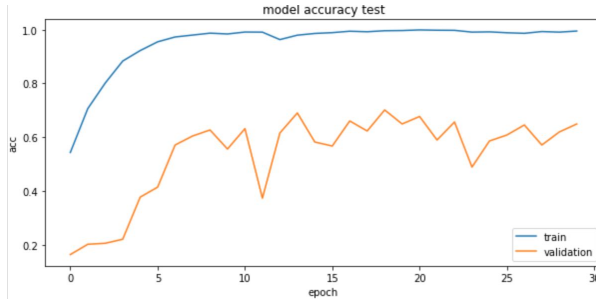
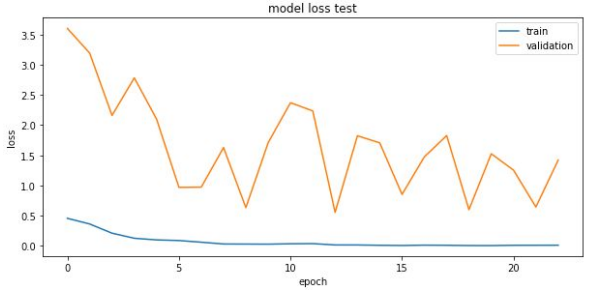
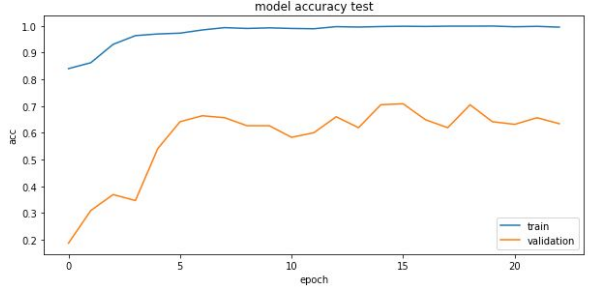
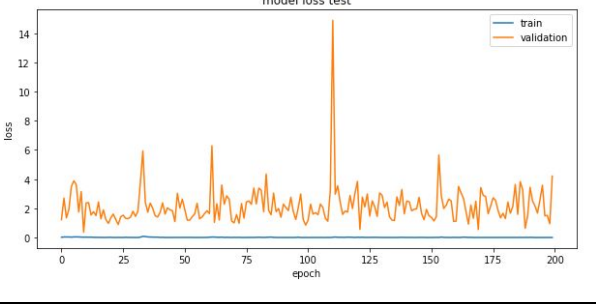
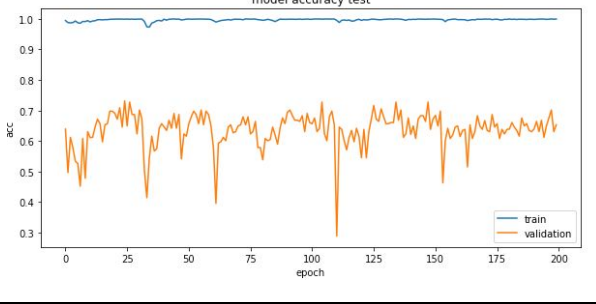


## 2. 上傳CNN小專案實驗結果：

## 2.1 網路結構

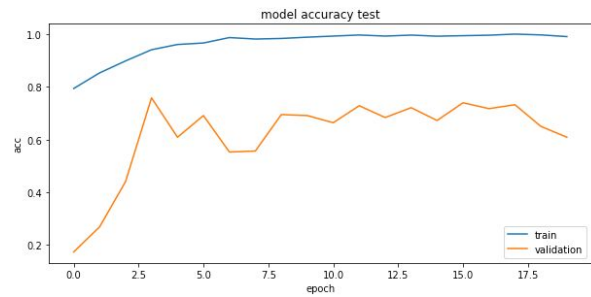
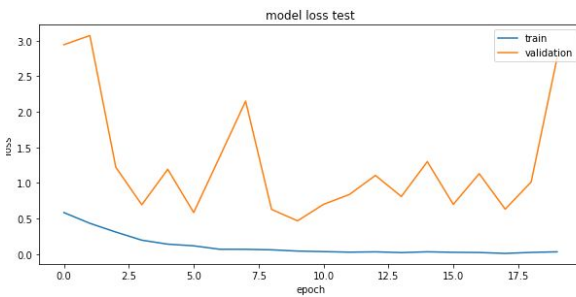


## 2.2 測試準確度

Model-1	Epoch:30 Input shape:(64,64,1) Dropout 0.2
	
Model-1	Epoch:30 Input shape:(150,150,1)
	
Model-1	Epoch:200 Input shape:(150,150,1)
	

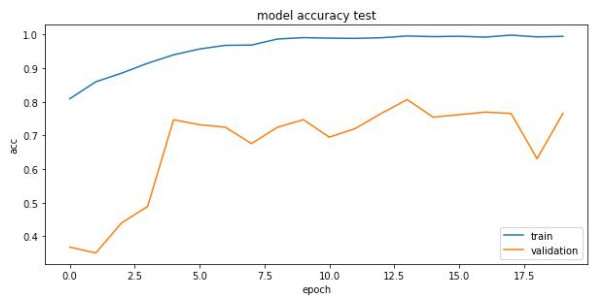
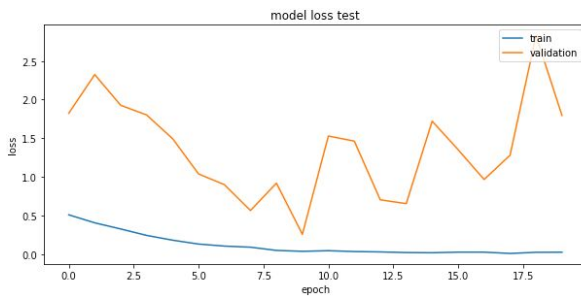
Model-1

Epoch:20 (early stoping)  
Input shape:(150,150,3)



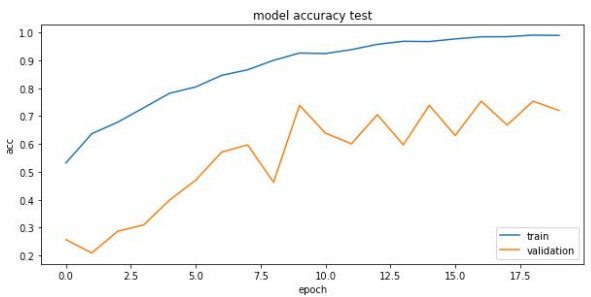
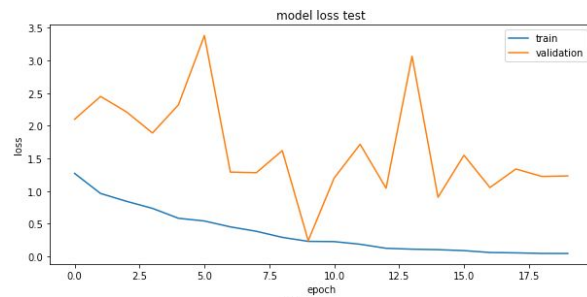
Model-1

Epoch:20 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256  
kernel\_size:(3,3)



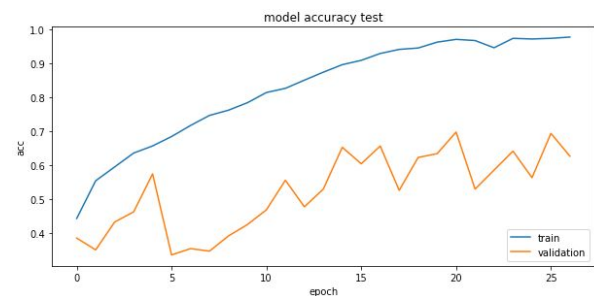
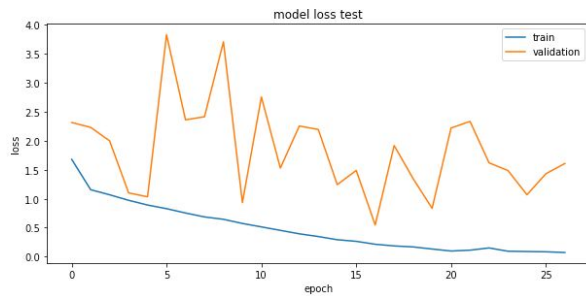
Model-1

Epoch:20 (early stoping)  
Input shape:(150,150,3)  
Conv filters:128、 256  
kernel\_size:(5,5)



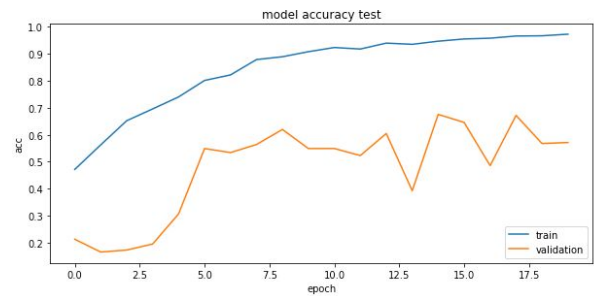
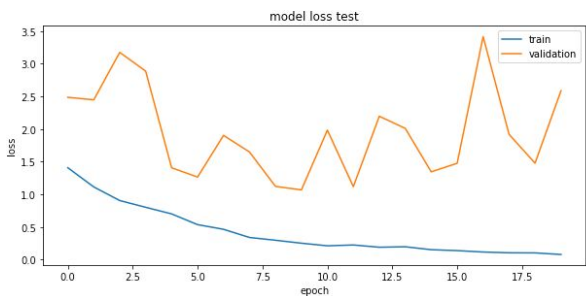
Model-1

Epoch:27 (early stoping)  
Input shape:(150,150,3)  
Conv filters:128、 256  
kernel\_size:(9,9)



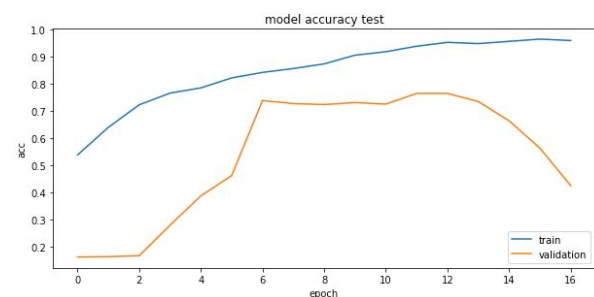
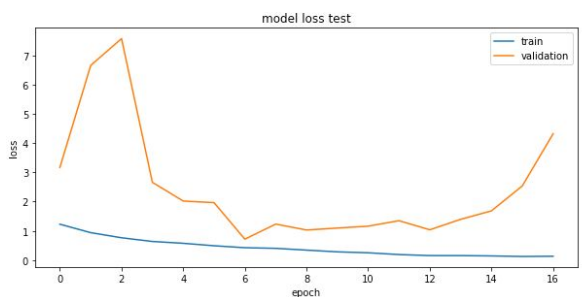
Model-1

Epoch:20 (early stoping)  
Input shape:(150,150,3)  
Conv filters:128、 256  
kernel\_size:(3,3)  
stride:(2,2)



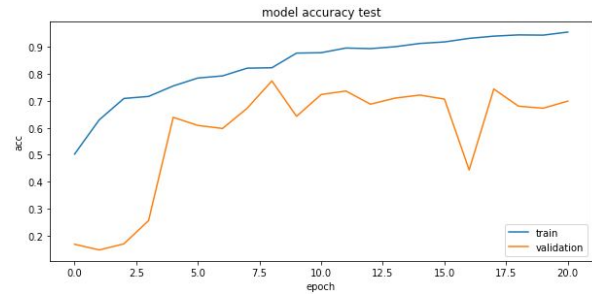
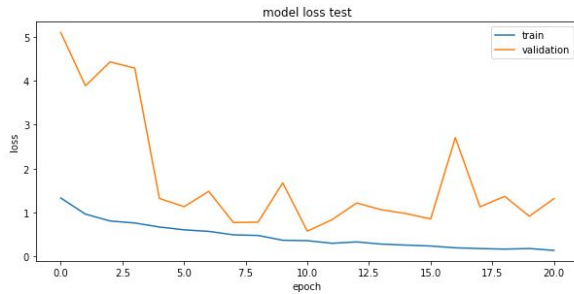
Model-1

Epoch:14 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512



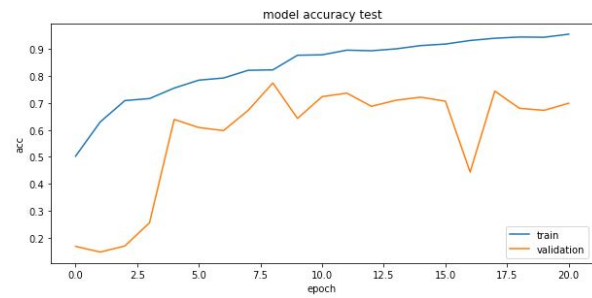
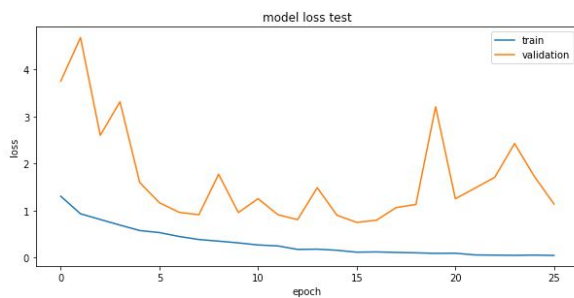
Model-1

Epoch:14 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
dense:512、 512



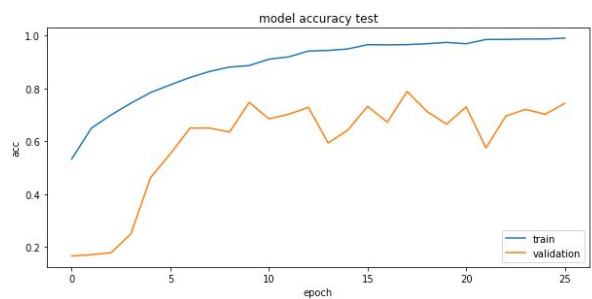
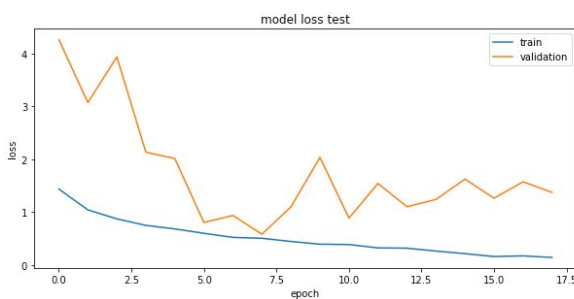
Model-1

Epoch:26 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
dense:1024



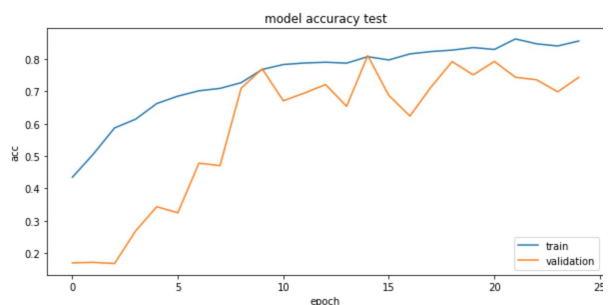
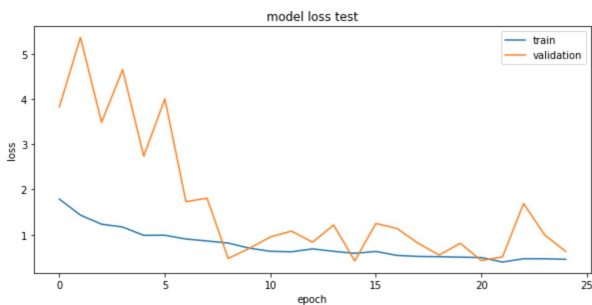
Model-1

Epoch:18 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
dense:2048



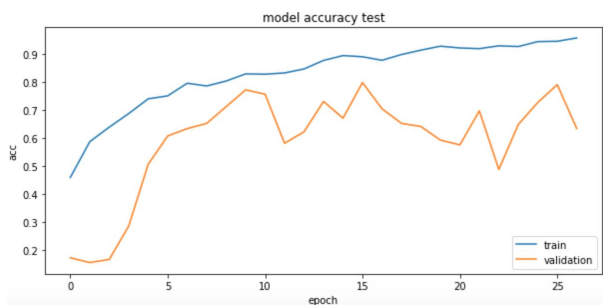
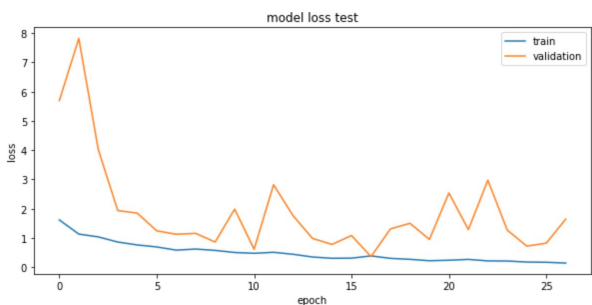
Model-1

Epoch:25 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
Dense:2048  
dropout:0.8



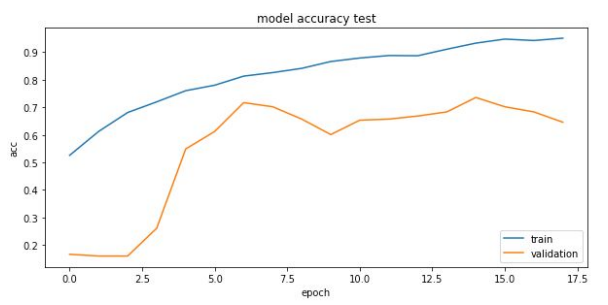
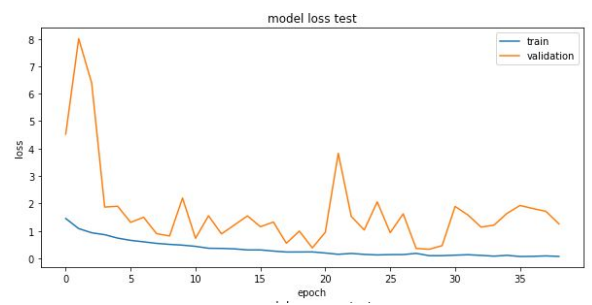
Model-1

Epoch:27 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
Dense:2048  
dropout:0.6



Model-1

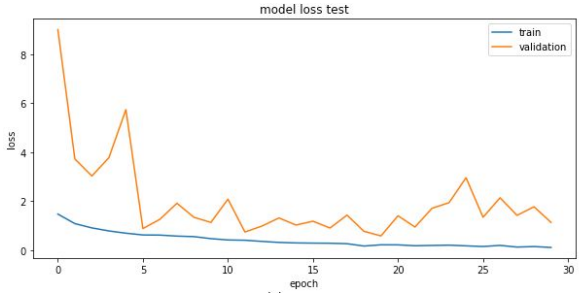
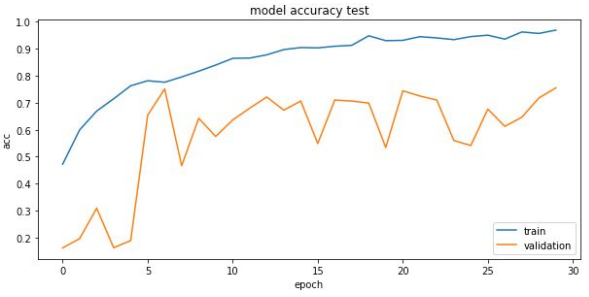
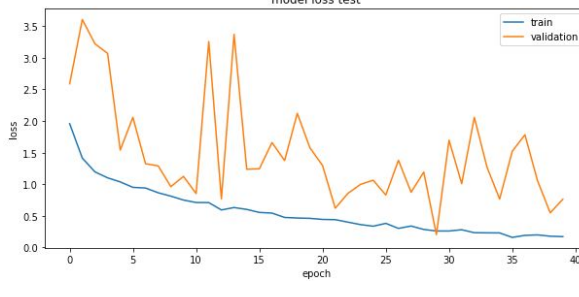
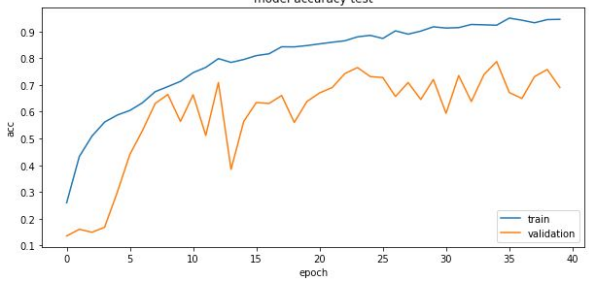
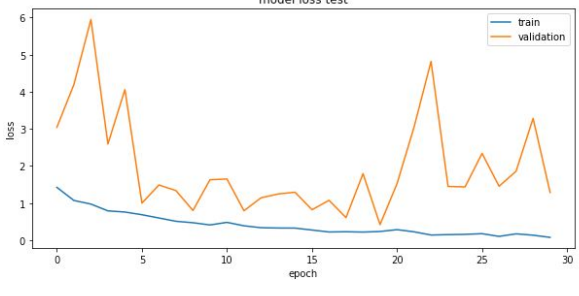
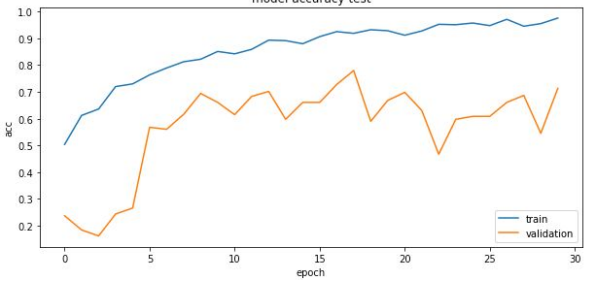
Epoch:37 (early stoping)  
Input shape:(150,150,3)  
Conv filters 128、 256、 512  
dense:512、 512、 512

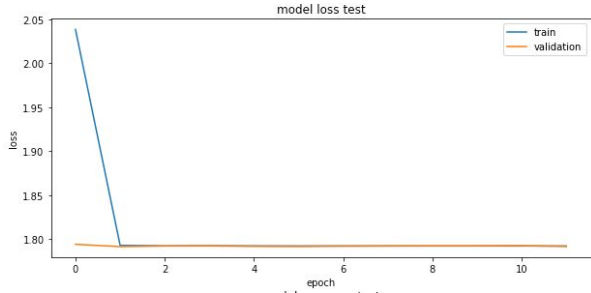
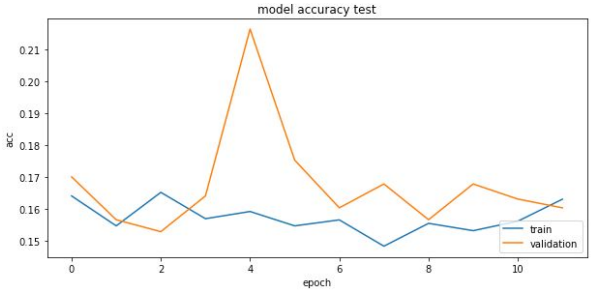
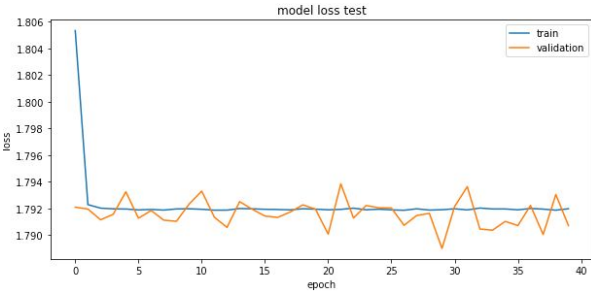
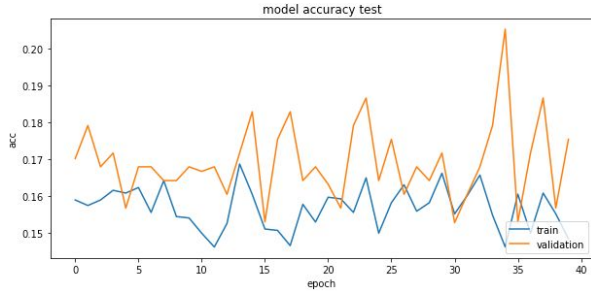
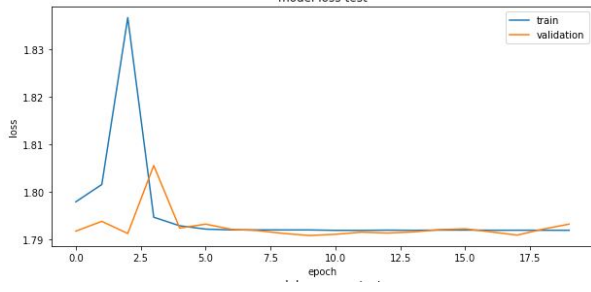
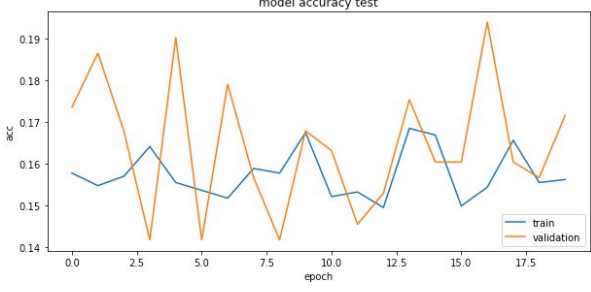


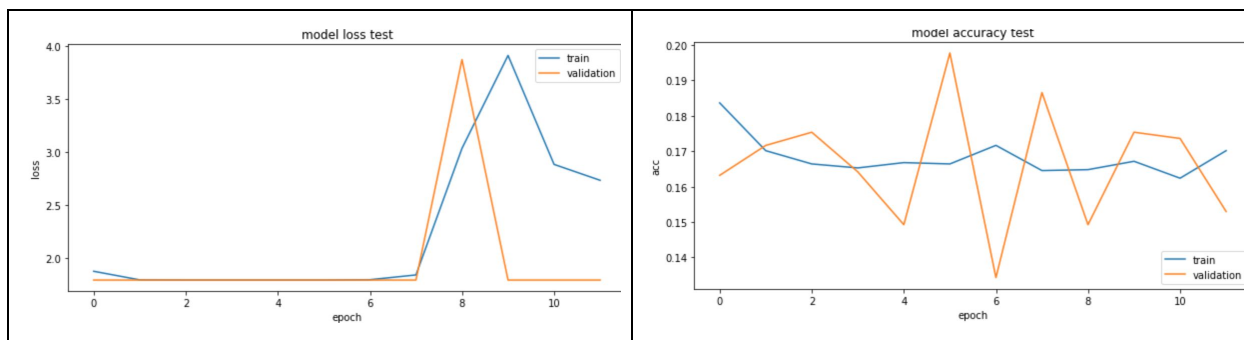
Model-1

Epoch:30 (early stoping)  
Input shape:(150,150,3)



	Conv filters 128、256、512 dense:1024、1024、1024
	
Model-1	Epoch:30 (early stoping) Input shape:(150,150,3) Conv filters 128、256、512 dense:512、512、512、512、512、512
	
Model-1	Epoch:30 (early stoping) Input shape:(150,150,3) Conv filters 128、256、512 dense:2048、2048
	
Model-1	Epoch:30 (early stoping) Input shape:(150,150,3) Conv filters 128、256、512 dense:2048、2048

VGG16	Epoch:14(early stoping) Input shape:(64,64,1)
	
VGG16	Epoch:40(early stoping) Input shape:(150,150,3)
	
VGG16	Epoch:40(early stoping) Input shape:(150,150,3) Batchnormalize
	
VGG16	Epoch:12(early stoping)



### 3. 心得與結論

由於上次在MLP實驗的時候，一直有over-fitting的問題，且訓練時間非常長，平均都要20-30分鐘之久。這次學習到了Convolution層，就是使用影像處理的sliding window去做的捲基層。使計算的方法比較快，由上圖的實驗也可知訓練速度快了不少，大約差1/10。上次作業發現的另一個問題是一開始前處理就把圖片縮到64x64，因此這次一開始先實驗150x150跟64x64的解析度差別，實驗結果顯示若圖片解析度較大，一開始的準確度會較佳，但對整體訓練準確度影響不大，反而圖片若是用rgb三個通道的話，訓練會比灰階圖片佳，使用rgb三個通道訓練的話，上升了10%。

觀察在不同的kernel\_size大小下，以3x3的kernel validation的準確率較佳，就算增加stride(window的偏移)，效果也不會比較好。增加filters的數目會使準確度上升。在實驗中可看出128+256比64+128 validation的準確度較好，甚至都是512會使over-fitting問題比較不嚴重，但訓練準確度會無法達到接近1.0，只有接近0.9。單純加深Conv層不會比較好，我們的模型中看起來較佳的模型配置為filter三層(128、256、512)，而dense層為2048個神經元的時候。

在實驗的過程中，我也嘗試去從頭實作一些常見的圖像辨識模型，如VGG16，更試著使用transfer learning的概念，可以看得出他的設計是filter(128、256、512)搭配兩層4096的dense層，雖然在我們的資料集中，分類效率不太佳，可能需要更多時間去理解為什麼模型在此資料集會訓練不佳。