學號：B06705001 系級：資管二 姓名：楊力行

1. 請比較你本次作業的架構，參數量、結果和原 HW3 作業架構、參數量、結果做比較。(1%)

HW3 是 20.4m 的參數量

模型架構是套用了 VGG19 的卷積和池化層，但全連接層數不同

2 層 64 個 3*3 大小的 filter, padding 為 same 的卷積層後接一層 maxpool 層

2 層 128 個 3*3 大小的 filter, padding 為 same 的卷積層後接一層 maxpool 層

4 層 256 個 3*3 大小的 filter, padding 為 same 的卷積層後接一層 maxpool 層

4 層 512 個 3*3 大小的 filter, padding 為 same 的卷積層後接一層 maxpool 層

4 層 512 個 3*3 大小的 filter, padding 為 same 的卷積層後接一層 maxpool 層

flatten 層

1 層 512 個 unit 的全連接層

1 層 7 個 unit 的 sofmax 輸出層

激活函數除最後一層皆為 relu, loss 為 categorical_crossentropy, optimizer 為 sgd lr=0.01,momentum=0.9,decay=5e-4

acc 為 0.68431 0.69044

本次作業是 48000 的參數量

1 個 convb2dlock

model.add(Convolution2D(int(32), (3, 3), strides=(2, 2), padding='same' ,activation='linear',input_shape=(42,42,1)))

model.add(LeakyReLU(alpha=0.01))

model.add(BatchNormalization())

1 個 dwclock

model.add (DepthwiseConv2D((3, 3), strides=(2, 2), padding='same' ,activation='linear'))

model.add(LeakyReLU(alpha=0.1))

model.add(BatchNormalization())

output 數量=input 數量

我的架構是(padding 都是 same，以及 leakyrelu 的 alpha 有部分是 0.01，有的是 0.1，是因為我當初在修改參數時漏改了部分)

1 個 conv2dblock stides=2, 32 個 3*3 大小的 filter

1 個 dwblock, strides=1

1 個 conv2dblock stides=1, 32 個 1*1 大小的 filter

1 個 dwblock , strides=2

1 個 conv2dblock stides=1, 32 個 1*1 大小的 filter

1 個 dwblock, strides=1

1 個 conv2dblock stides=1, 64 個 1*1 大小的 filter

1 個 dwblock, strides=2

1 個 conv2dblock stides=1, 96 個 1*1 大小的 filter

1 個 dwblock, strides=1

1 個 conv2dblock stides=1, 128 個 1*1 大小的 filter

1 個 dwblock, strides=2

1 個 conv2dblock stides=1, 128 個 1*1 大小的 filter

1 層 globalAveragePooling2D()

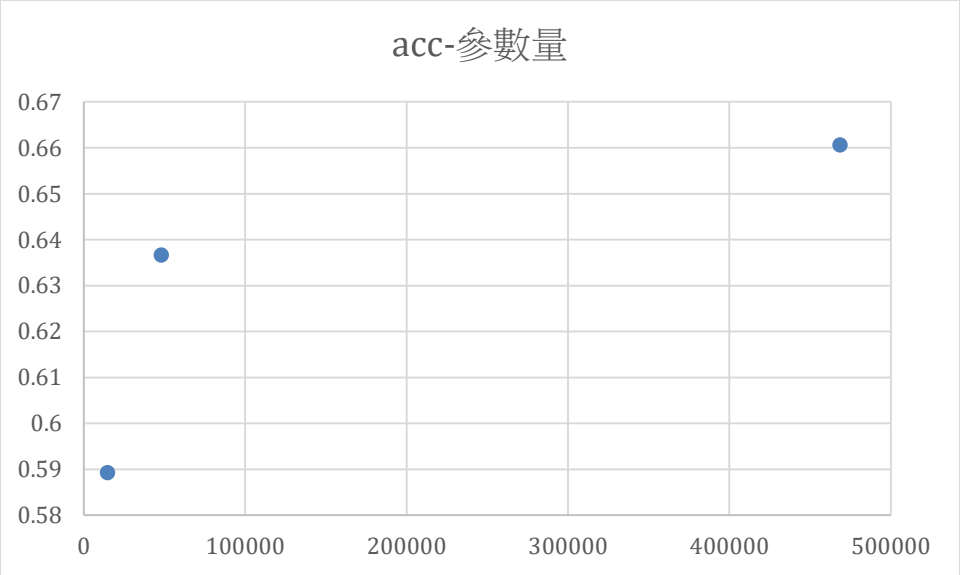
1 層 1 層 7 個 unit 的 sofmax 輸出層

loss='categorical_crossentropy',optimizer='adam'

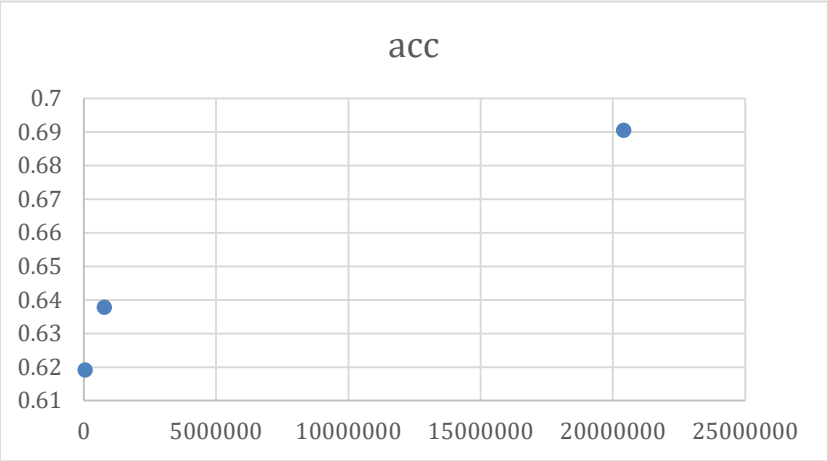
acc=0.62468 0.63666

本次的架構只用了 7 層的 conv2d，並且使用的並不是 3*3 大小的 filter 而是 1*1，另外用參數量遠低於 conv2d 的 7 層的 dw 層取出特徵，並且把 maxpool 層用 dw strides=2 來代替降維的作用，以及用 globalAveragePooling2D 來取代原本參數量超多的全連接層，透過這些才使得在參數量為原本的 1/500 時 acc 卻沒有下降的太多。

2. 請使用 MobileNet 的架構，畫出參數量-acc 的散布圖（橫軸為參數量，縱軸為 accuracy，且至少 3 個點，參數量選擇時儘量不要離的太近，結果選擇只要大致收斂，不用 train 到最好沒關係。）(1%)



3. 請使用一般 CNN 的架構，畫出參數量-acc 的散布圖（橫軸為參數量，縱軸為 accuracy，且至少 3 個點，參數量選擇時儘量不要離的太近，結果選擇只要大致收斂，不用 train 到最好沒關係。）(1%)



4. 請你比較題 2 和題 3 的結果，並請針對當參數量相當少的時候，如果兩者參數量相當，兩者的差異，以及你認為為什麼會造成這個原因。(2%)

可以看出，當 acc 差不多時，cnn 所用的參數量是 mobilenet 的數倍之多，然而，當參數夠大時，mobilenet 的 acc 卻難以達到 cnn 的 acc，隨著參數量的增加，mobilenet 得到的 acc 收斂的較快，也較不易得到更高的 acc

48000 參數的 cnn　　acc 0.61911

48000 參數的 mobilenet acc 0.63666

小參數在參數量相近時，可以看出 mobilenet 所得到的 acc 較高，我想造成這個原因的是因為 mobilenet 是用 DepthwiseConv2D 這個參數量非常少的層去取出特徵，再用 conv2d 1*1 這個比 3*3 小了數倍的 conv 層去進行運算，也就是把原本的 conv2d 3*3 用這兩層疊加來代替，可以用少了數倍的參數量來做出接近原本的效果，因此在同等參數量時，可以疊加出較多的層數和各層的 filter 數，因此最終的表現較好。