學號：B06901053　系級：電機三　姓名：謝承延

1. 請從 Network Pruning/Quantization/Knowledge Distillation/Low Rank Approximation/Design Architecture 選擇兩個方法(並詳述)，將同一個大 model 壓縮至同等數量級，並討論其 accuracy 的變化。 (2%)

大 Model:同 HW3 使用的 Model

Accuracy: Iteration:87

Train Accuracy	Train Loss	Validation Accuracy	Validation loss	Kaggle Accuracy
0.922664	0.003689	0.761808	0.014849	0.82725

因為是 HW3 的 Model，詳細的訓練曲線就不附上了

架構及參數說明:

檔案大小:49.45MB

總參數量:12956235

Batch size= 64

Learning rate = 0.001

Optimizer:Adam

照片 Resize 為 128X128

1	`nn.Conv2d(3, 64, 3, 1, 1),`
	`nn.BatchNorm2d(64),`
	`nn.ReLU(),`
	`nn.Conv2d(64, 64, 3, 1, 1),`
	`nn.BatchNorm2d(64),`
	`nn.ReLU(),`
	`nn.MaxPool2d(2, 2, 0),`

2	`nn.Conv2d(64, 128, 3, 1, 1),`
	`nn.BatchNorm2d(128),`
	`nn.ReLU(),`
	`nn.Conv2d(128, 128, 3, 1, 1),`
	`nn.BatchNorm2d(128),`
	`nn.ReLU(),`
	`nn.MaxPool2d(2, 2, 0),`

3	`nn.Conv2d(128, 256, 3, 1, 1),`
	`nn.BatchNorm2d(256),`
	`nn.ReLU(),`
	`nn.Conv2d(256, 256, 3, 1, 1),`
	`nn.BatchNorm2d(256),`
	`nn.ReLU(),`
	`nn.MaxPool2d(2, 2, 0),  #`

4	nn.Conv2d(256, 512, 3, 1, 1), nn.BatchNorm2d(512), nn.ReLU(), nn.Conv2d(512, 512, 3, 1, 1), nn.BatchNorm2d(512), nn.ReLU(), nn.MaxPool2d(2, 2, 0), #
5	nn.Conv2d(512, 512, 3, 1, 1), nn.BatchNorm2d(512), nn.ReLU(), nn.Conv2d(512, 256, 3, 1, 1), nn.BatchNorm2d(256), nn.ReLU(), nn.MaxPool2d(2, 2, 0), #
6	self.fc = nn.Sequential(nn.Linear(256*4*4, 1024), nn.Dropout(p=0.5), nn.ReLU(), nn.Linear(1024, 512), nn.Dropout(p=0.5), nn.ReLU(), nn.Linear(512, 11),)

實做 1:Design Architecture

架構及參數說明:

照片 Resize 為 128X128

原本的大 MODEL convolution 的部分有 5 部分，我將第 2 3 4 5 部分做 Depthwise and Pointwise convolution，原本的 BatchNorm 跟 ReLu 也都不做更動

為使每一部分的 input 跟 output 跟原本大 MODEL 完全相同，1X1 的 filter 數量會跟原本的 output channel 數量相同。

Fully Connected Network 的部分則沒有做更動

檔案大小:21.67MB

總參數量:5671785

Batch size= 64

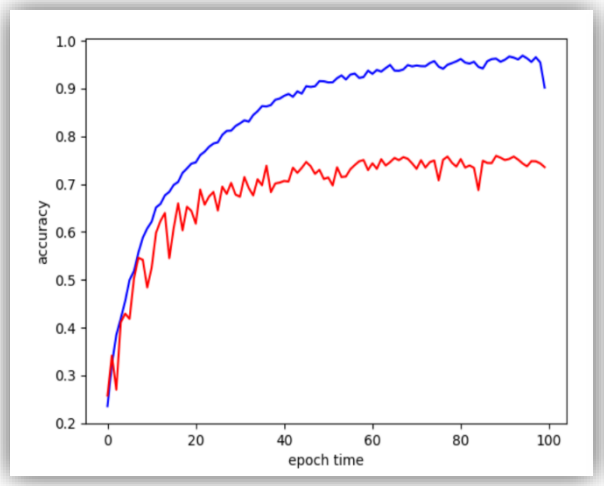
Learning rate = 0.001

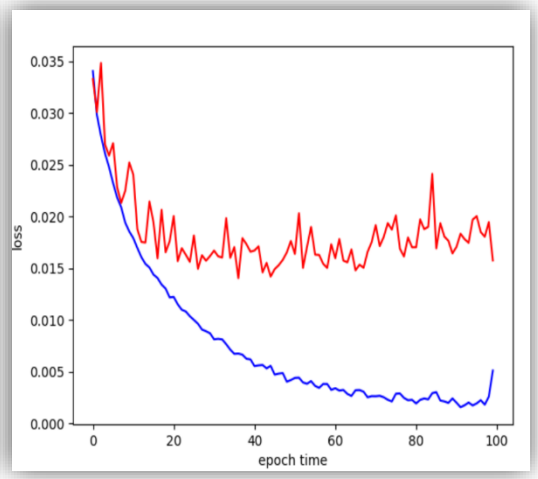
Optimizer:Adam

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration89)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.9624974660	0.0020080063	0.7591837	0.017666

Accuracy& LOSS:





訓練曲線的部分跟原本大 MODEL 差不多，train 到 100 次 accuracy 就差不多 saturate 了，參數量跟檔案大小變成了 1/2 倍，而 validation accuracy 只下降 1%不到，validation loss 也只差了一點。

實做 2: Knowledge Distillation

架構及參數說明:

照片 Resize 為 128X128

結構跟 Design Architecture 完全相同

訓練方式使用 KD(對於 Teacher NET 的預測做學習)，teacher Net 選擇上述的大 Model。

檔案大小:21.67MB

總參數量:5671785

Batch size= 64

Learning rate = 0.001

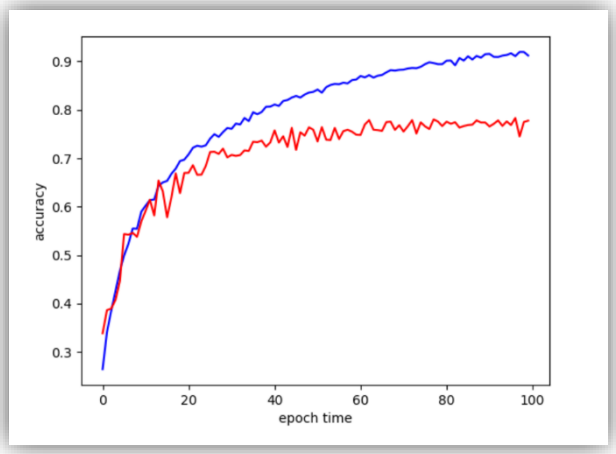
Optimizer:Adam

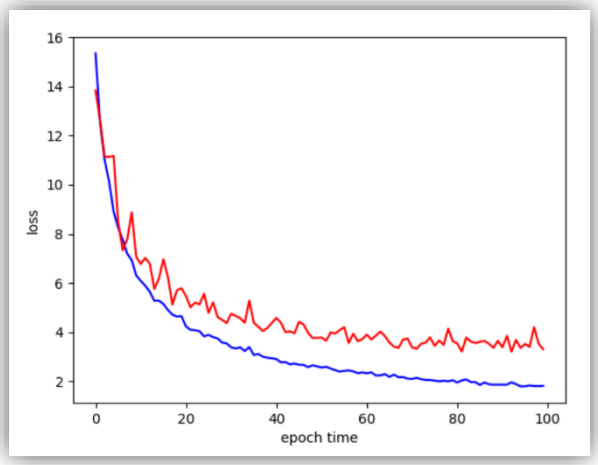
Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration97)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss

0.9107034	1.82830	0.783090	3.39739384

Accuracy& LOSS:





可以看到 LOSS 的部分比正常的 train 還要來的高許多，可能的原因是 KD 是根據大 MODEL 的預測數值做訓練，而大 MODEL 的預測數值本身就不可能會像用 label train 那樣，每個 class 的值只有 1 跟 0 的分別(如 class1 就是 10000000000)，小 MODEL 根據預測出來的數值結果去學習，預測出來的數值就自然會差得更遠，造成 loss 變高許多。

但是令人意外的是，小 model 利用 KD 做訓練得出來的 validation accuracy 竟然比原本還要高出約 2%，我認為可能的原因是:KD 的訓練除了學習某張圖片是甚麼 label 以外，他也能學習到別的 label 的資訊，因此可以彌補有些 label 本身 data 量比較少的限制，例如 train data 中的 class2 明顯就比其他 class 少，KD 的這種性質彌補了這樣的缺點，造成 validation accuracy 上升。

2. [Knowledge Distillation] 請嘗試比較以下 validation accuracy (兩個 Teacher Net 由助教提供)以及 student 的總參數量以及架構，並嘗試解釋為甚麼有這樣的結果。你的 Student Net 的參數量必須要小於 Teacher Net 的參數量。(2%)

x. Teacher net architecture and # of parameters:
torchvision’s ResNet18, with 11,182,155 parameters.

y. Student net architecture and # of parameters:
Student Net 我使用的是跟 sample code 相同的 Mobile Net v1，參數量為 256779

Type	Filter Size	Input size
Conv2D	3x3x3x16	256x256x3
MaxPool(2,2)		256x256x16
Conv2D(DW)	3x3x16 dw	128x128x16
Conv2D	1x1x16x32	128x128x16
MaxPool(2,2)		128x128x32
Conv2D(DW)	3x3x32 dw	64x64x32
Conv2D	1x1x32x64	64x64x32
MaxPool(2,2)		64x64x64
Conv2D(DW)	3x3x64 dw	32x32x64
Conv2D	1x1x64x128	32x32x64
MaxPool(2,2)		32x32x128
Conv2D(DW)	3x3x128 dw	16x16x128
Conv2D	1x1x128x256	16x16x128
Conv2D(DW)	3x3x256 dw	16x16x256
Conv2D	1x1x256x256	16x16x256
Conv2D(DW)	3x3x256 dw	16x16x256
Conv2D	1x1x256x256	16x16x256
Conv2D(DW)	3x3x256 dw	16x16x256
Conv2D	1x1x256x256	16x16x256
AvgPool2D(1,1)		16x16x256
FC	256x11	1x1x256

a. Teacher net (ResNet18) from scratch: 80.09%

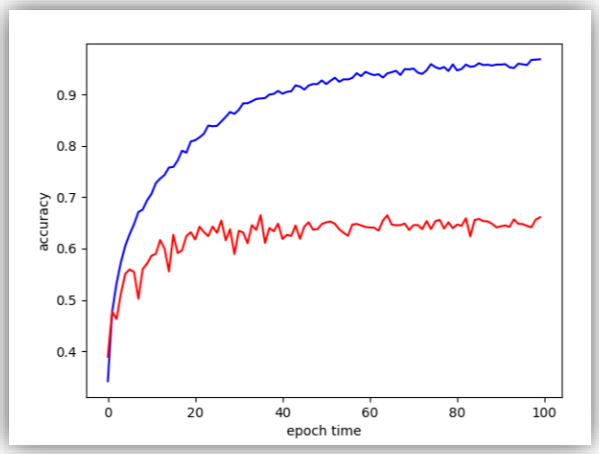
b. Teacher net (ResNet18) ImageNet pretrained & fine-tune: 88.41%

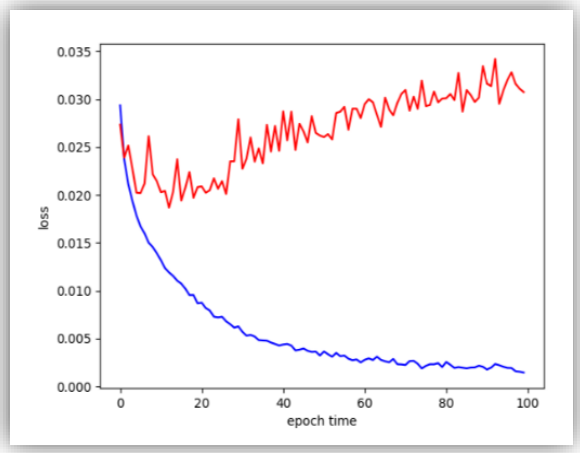
c. Your student net from scratch:
利用上述的 Student Net 做 train
Batch size= 32
Learning rate = 0.001
Optimizer:Adam

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration100)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.968275	0.001433	0.660933	0.030744

Accuracy and Loss:





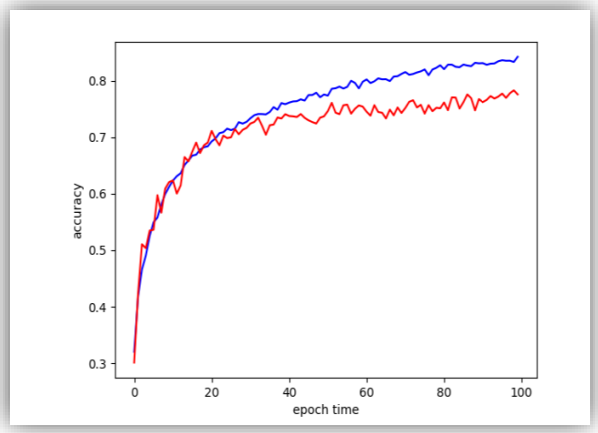
可以看到直接用 Student Net 訓練的結果 validation accuracy 是比較低的，呈現 overfitting 的狀況，因為 Mobile Net 本身 model 的參數量很少，因此容易 overfitting，也可以看到因為 model 本身架構較小，很快 validation accuracy 就飽和了。

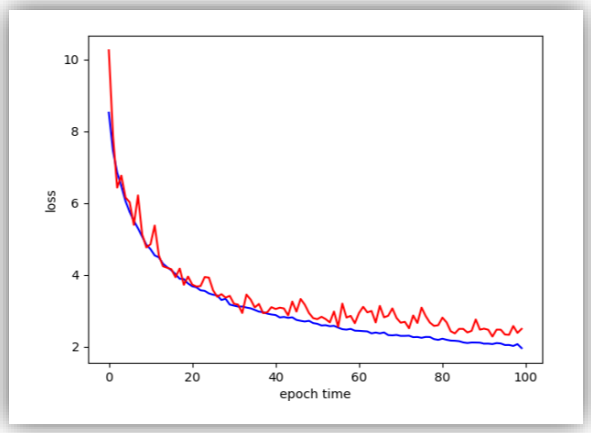
d. Your student net KD from (a.):
以 Resnet18 from scratch 作為 Teacher Net，Student Net 則同為 Mobile Net
Batch size = 32
Learning rate = 0.001
Optimizer:Adam

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration99)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.83286	2.072909	0.782799	2.381792

Accuracy and Loss:





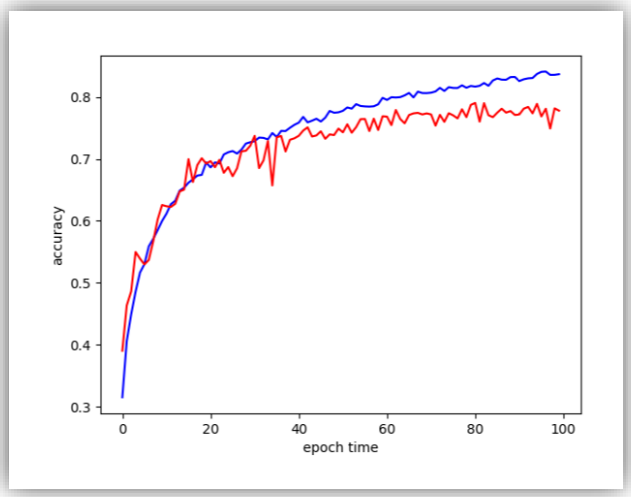
因為是利用 Knowledge Distillation 的方法，可以看到 loss 會比直接做訓練來的高，然而卻解決了 overfitting 的問題，且 validation accuracy 也高出許多，這可能原因是 KD 這個方法讓一個 sample 除了提供該 label 的學習，也提供其他 class 的資訊，使 validation accuracy 能夠提升。

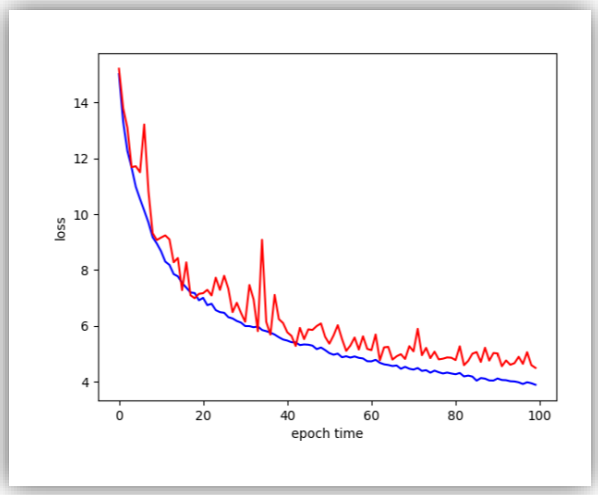
e. Your student net KD from (b.):
以 Pretrained Resnet18 作為 Teacher Net，Student Net 則同為 Mobile Net
Batch size = 32
Learning rate = 0.001
Optimizer:Adam

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration81)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.816643	4.263776	0.790379	4.77001

Accuracy and Loss:





Validation Accuracy 比起(d)的狀況來說高了岔約 1%，然而 train loss 跟 val loss 都高了大約 2 左右，這一點其實蠻讓我意外的，畢竟 pretrained ResNet18 可以取得大約 88%的正確率，照理來說他的預測結果 loss 會較小，然而以它為 TeacherNet 時 MobileNet 的預測結果 loss 卻較大，這有可能代表 MobileNet 是比較無法去模仿它的預測行為的，因為如果他能夠正確模仿的話，照理而言 loss 應該會比(d)的結果來的小。
當然如果再將 epoch 開到更大的話，也許結果會不同，但為了公平比較(c)(d)(e)在此就以 validation accuracy 大約進入 saturate 的狀態來討論。

3. [Network Pruning] 請使用兩種以上的 pruning rate 畫出 X 軸為參數量，Y 軸為 validation accuracy 的折線圖。你的圖上應該會有兩條以上的折線。(2%)(X)

4. [Low Rank Approx / Model Architecture] 請嘗試比較以下 validation accuracy，並且模型大小須接近 1 MB。 (2%)

a. 原始 CNN model (用一般的 Convolution Layer) 的 accuracy

架構及參數說明:

照片 Resize 為 128X128

相關參數及結構如下

檔案大小:947.44KB

總參數量:241099

Batch size= 256

Learning rate = 0.001

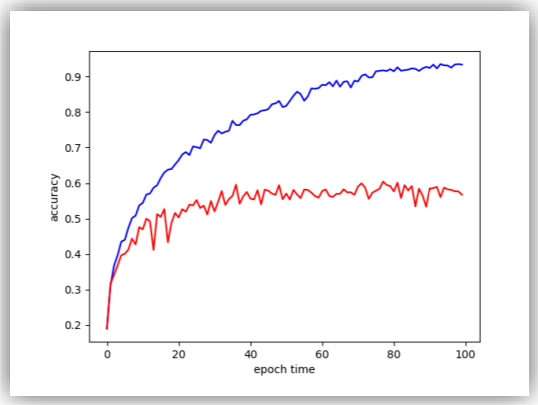
Optimizer:Adam

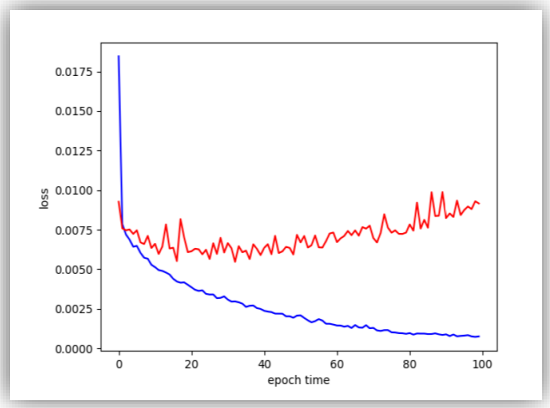
Type	Filter Size	Input size
Conv2D	3x3x3x16	128x128x3
MaxPool(2,2)		128x128x16
Conv2D	3x3x16x32	64x64x16
MaxPool(2,2)		64x64x32
Conv2D	3x3x32x64	32x32x32
Conv2D	3x3x64x64	32x32x64
MaxPool2D(1,1)		32x32x64
FC	(16x16x64)x11	16x16x64

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration78)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.917697	0.000967	0.604373	0.007238

Accuracy& LOSS:





Validation accuracy 大約落在 60%，因為 model 的參數很少所以 accuracy 也蠻快就達到飽和了，可以看到 validation loss 在 epoch 30 後開始有不減反增的趨勢，整體而言呈現 overfitting 的狀況。

b. 將 CNN model 的 Convolution Layer 換成參數量接近的 Depthwise & Pointwise 後的 accuracy

架構及參數說明:

照片 Resize 為 128X128

結構如下

檔案大小:744.75KB

總參數量:188971

Batch size= 256

Learning rate = 0.001

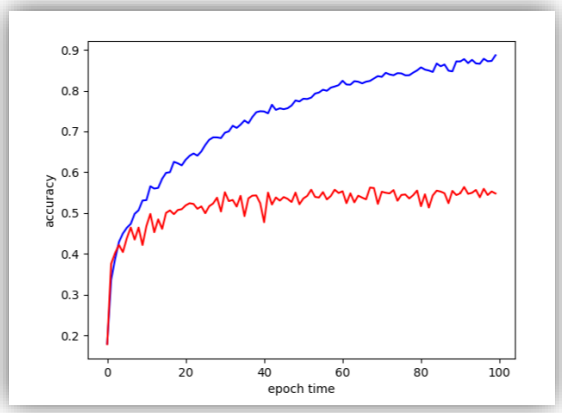
Optimizer:Adam

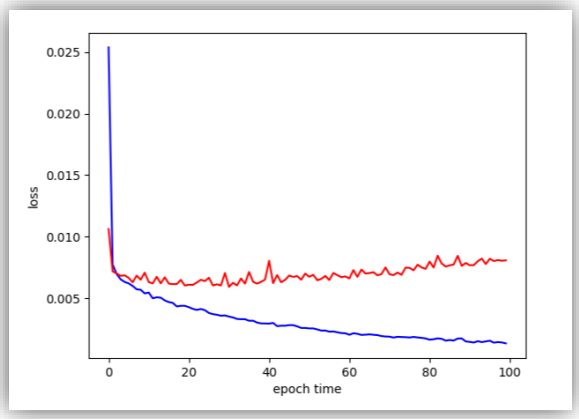
Type	Filter Size	Input size
Conv2D	3x3x3x16	128x128x3
MaxPool(2,2)		128x128x16
Conv2D(DW)	3x3x16 dw	64x64x16
Conv2D	1x1x16x32	64x64x16
MaxPool(2,2)		64x64x32
Conv2D(DW)	3x3x32 dw	32x32x32
Conv2D	1x1x32x64	32x32x64
Conv2D(DW)	3x3x64 dw	32x32x64
Conv2D	1x1x64x64	32x32x64
MaxPool(2,2)		32x32x64
FC	(16x16x64)x11	16x16x64

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration92)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.877458	0.001389	0.563848	0.00768

Accuracy& LOSS:





Validation accuracy 比起正常 CNN model 少了 4%左右，loss 也都有變差一些，參數量的改變大約為原本的 75%

c. 將 CNN model 的 Convolution Layer 換成參數量接近的 Group Convolution Layer （Group 數量自訂，但不要設為 1 或 in_filters）

架構及參數說明:

照片 Resize 為 128X128

結構如下

檔案大小:788.05KB

總參數量:200059

Batch size= 256

Learning rate = 0.001

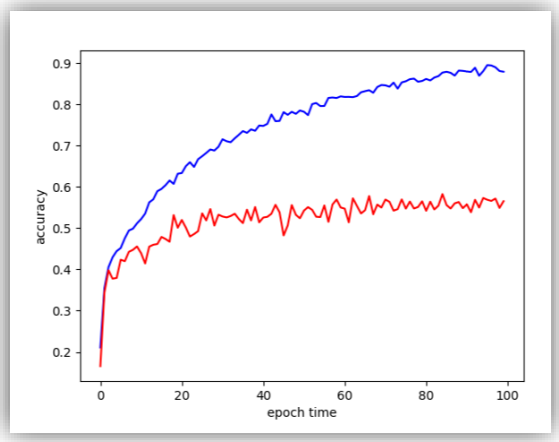
Optimizer:Adam Groups=4

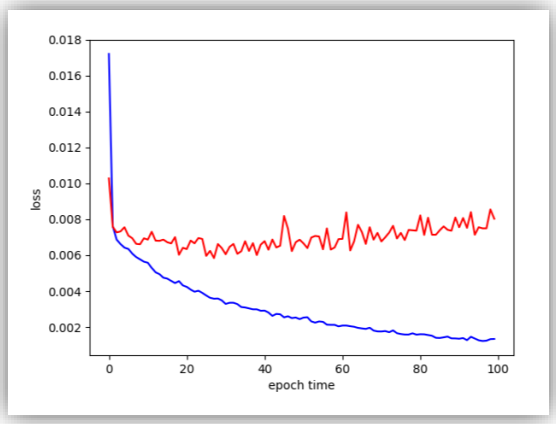
Type	Filter Size	Input size
Conv2D	3x3x3x16	128x128x3
MaxPool(2,2)		128x128x16
Conv2D(groups=4)	3x3x4x16 dw	64x64x16
Conv2D	1x1x16x32	64x64x16
MaxPool(2,2)		64x64x32
Conv2D(groups=4)	3x3x4x32 dw	32x32x32
Conv2D	1x1x32x64	32x32x64
Conv2D(groups=4)	3x3x4x64 dw	32x32x64
Conv2D	1x1x64x64	32x32x64
MaxPool(2,2)		32x32x64
FC	(16x16x64)x11	16x16x64

Accuracy: Iteration:100(以下數據取 validation accuracy 最高者 發生於 iteration 85)

Train Accuracy	Train Loss	Validation Accuracy	Validation loss
0.87685	0.001402	0.582507	0.007138

Accuracy& LOSS:





Validation accuracy 比正常 CNN 大約少了 2%，loss 則相當。

Findings:
可以發現使用 CNN 的時候 group 的多寡造成了參數量跟準確率之間的 trade off，一般的 CNN 取用的是 groups=1，DepthWise & Pointwise Convolution 則是 groups=input channel，Group Convolution 則是藉在其中，而參數量跟準確率排名分別為
Accuracy: CNN > Group Convolution > DW+PW
參數量: CNN < Group Convolution < DW+PW

在我的實驗中，參數量越多的 model 達到越高的 validation accuracy，且因為總大小 1MB 的 model 其實算是架構很小的，所以參數量的提升可以明顯反映到 accuracy 上，因此可以清楚觀察到參數數量跟 accuracy 的 trade off。

因為這個 CNN 的架構大部分參數量仍來自於 Fully connected network，若在許多知名 CNN 架構下，因為 convolution layer 很多層，所以來自於 convolution layer 的參數量自然就多很多，在這樣的狀況下，若要用 groups 數量調整 model 檔案大小，衡量參數量 vs Accuracy 的 trade off 就更加重要了。