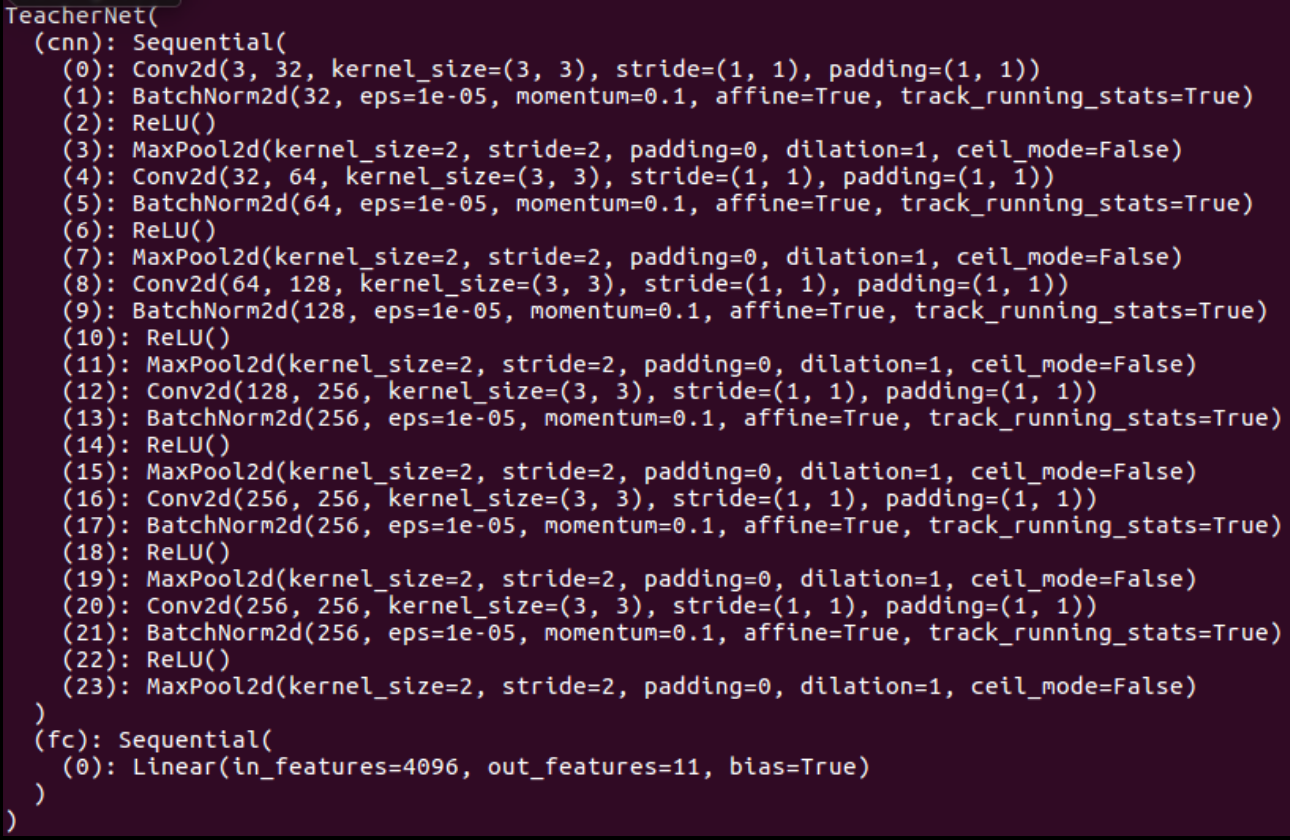
學號：r07921001 系級：電機所二 姓名：李尚倫

1. 請從Network Pruning/Quantization/Knowledge Distillation/Low Rank Approximation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%)

這題中用到的大Model的結構如下圖，是一個自己接的簡單的CNN model (以下稱Big model)：6 layer CNN + 1 layer FC model train from scratch 100 ep using Adam



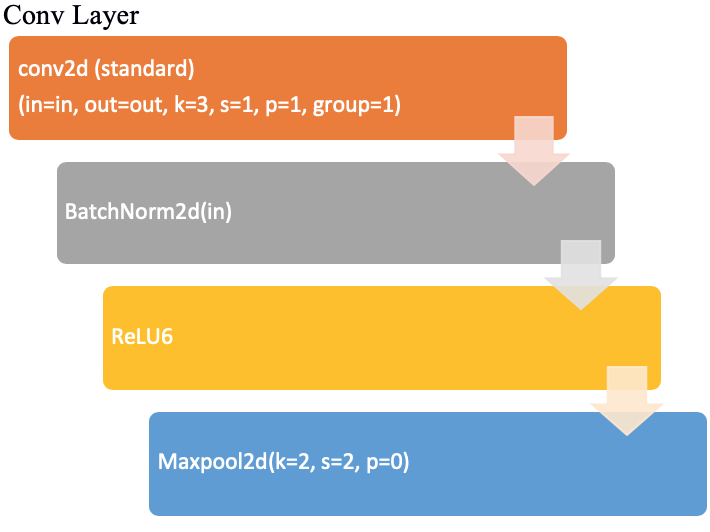
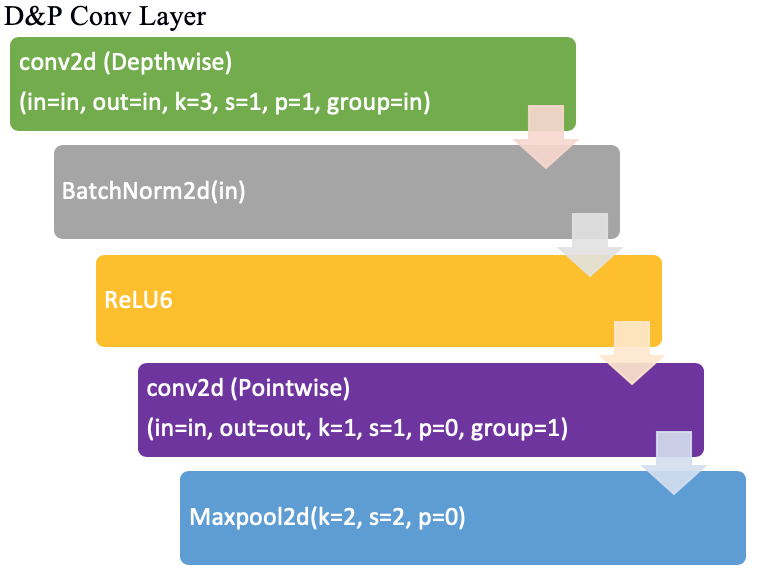
而我們使用以下兩種方法來壓縮此Model至同一個數量級，結果如下表：

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Validation Accuracy | File size (B) | Model’s parameters |
| Big model | 0.7988 | 6,477,626 | 1,615,627 |
| Knowledge Distillation with Low Rank Approximation | 0.7901 | 1,047,430 | 256,779 |
| Directly Quantization | 0.7913 | 1,623,152 | 1,615,627 |

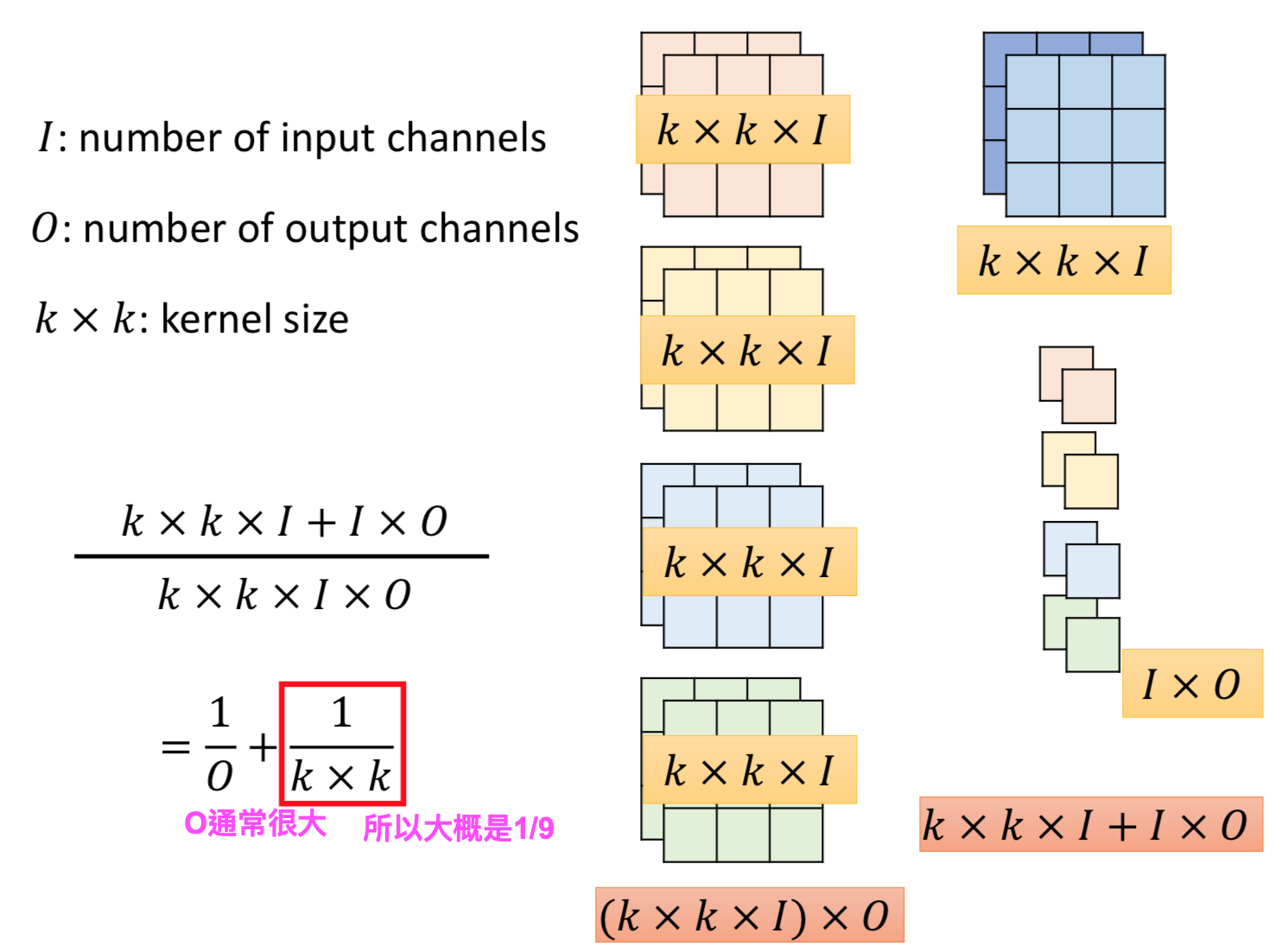
Knowledge Distillation with Low Rank Approximation的方法是使用Low Rank Approximation 的方式去設計一個輕量的Student model用Knowledge Distillation的方式來學Teacher model (即為Big model)，Low rank 的Student model的設計如下：

Student\_model：(inputs=256x256x3, output=11)

一般的Convolution Layer weight數量很大，僅在第一層使用(通常第一層都需要是一般的conv)，而D&P的DW(Depthwise Convolution Layer)則是各channel的feature map各過自己的filter處理後，再用PW(Pointwise Convolution Layer)把所有channel的feature map的單個pixel資訊合在一起。如下圖是Student\_model的單個layer展開：

如此D&P的做法可以大量減少參數，做出一個輕量化的model，圖示如下圖，且accuracy比單純只是參數少的CNN來的更高(詳見第四題討論)：



而輕量化的student model通常會搭配teacher model一起訓練會比train from scratch的效果好(詳見第二題討論)，因此搭配Knowledge Distillation的方法，讓student model在訓練時不只和原本data的label算loss也和teacher model的output算loss：

上式為設計的Loss，用Adam optimizer lr=0.001並training 100ep後的結果，參照上面的table顯示，結果和Big model接近，無論檔案大小或參數上的壓縮效果都不錯。

而Directly Quantization則是直接壓縮Big model的大小，把原本的訓練好的Float32的參數，32bit改降存成8bit，使用的是numpy.unit8，轉換的方式是根據上下界做normalize後map到0~255的區間，公式如下：



儲存上下界與轉換完的unit8 weight，並存成.pkl檔，檔案的大小可以縮小四倍左右，並保有不錯的accuracy，藉此可以觀察出weight的resolution其實並沒有很高，只需要0~255搭配上下界就可以很趨近了，但相較於前者的方式，這個方式僅降低檔案大小，並沒有減少模型實際的參數量，因此計算速度不會增快。

由上面的比較可知，兩者各有優缺點，所以最後kaggle上的結果我們使用兩者混合的方式，可以既減少參數增加predict的速度，也可以大量減少檔案大小。先用Low Rank Approximation的方式重新設計了一個Student model比原本助教提供的更多層，並調整model結構，然後再用Knowledge Distillation以助教提供的ResNet18 ImageNet pretrained & fine-tune 當作Teacher net來訓練，以0.002的learning rate搭配Adam optimizer，訓練200個epoch，得到還不錯的accuracy，然後再下調learning rate到0.001訓練3個epoch，accuracy仍有上升，繼續下調learning rate到0.0005，訓練2個epoch，就到極限了(後續還有再降低learning rate或訓練多一點epoch並沒有獲得更好的效果)，得到第一段的compressed model (.bin model)。此階段的model仍有918038 Bytes還不到門檻，再使用Quantization把參數從32bit的資料型態用前面提到的方式採存成8bit，一口氣把大小降到236937 Bytes，為第二階段compressed model (.pkl model)，在Kaggle的accuracy 有0.8494超過strong baseline，並有在條件的大小內。

Student\_model\_deeper：(inputs=256x256x3, output=11)

實驗的Table如下，最後submit為student\_model\_deeper\_203ep：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model / epoch | .bin  acc  on val | .pkl  acc  on val | .pkl  acc  on kaggle | model parameters | .pkl  file size (B) |
| student\_custom\_small  (助教提供的小model) | 0.8137 | 0.8087 | 0.8344 | 256779 | 268471 |
| student\_model/202ep | 0.8207 | 0.8155 | 0.8332 | 256779 | 268471 |
| student\_model\_deeper/200ep | 0.8283 | 0.8277 | 0.8446 | 224011 | 236937 |
| **student\_model\_deeper/203ep** | **0.8388** | **0.8362** | **0.8494** | **224011** | **236937** |
| student\_model\_deeper/205ep | 0.8429 | 0.8376 | 0.8470 | 224011 | 236937 |

以下三題只需要選擇兩者即可，分數取最高的兩個。

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:   
使用第一題的Student\_model, with 256,779 parameters.

a. Teacher net (ResNet18) from scratch: 80.12%

b. Teacher net (ResNet18) ImageNet pretrained & fine-tune: 88.51%

c. Your student net from scratch: 79.94%

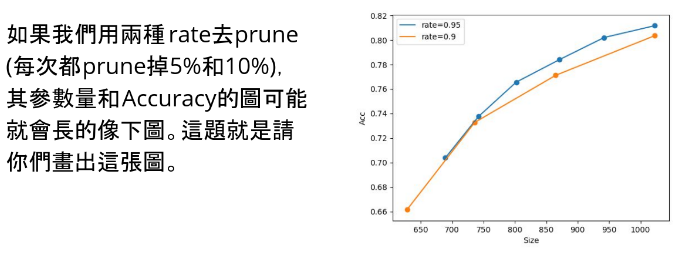
d. Your student net KD from (a.): 79.80%

e. Your student net KD from (b.): 82.13%

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Role  (model)  training method | Validation accuracy | Model  size (B) | Parameters | Filename |
| a. Teacher net (ResNet18)  from scratch | 0.8012 | 44,788,712 | 11,182,155 | teacher\_resnet18\_from\_scratch.bin |
| b. Teacher net (ResNet18)  ImageNet pretrained & fine-tune | 0.8851 | 44,788,712 | 11,182,155 | teacher\_resnet18.bin |
| student\_custom\_small | 0.8137 | 1047430 | 256779 | student\_custom\_small.bin |
| c. Student net  (Student\_model)  from scratch | 0.7994 | 1047574 | 256779 | student\_model\_from\_scratch.bin |
| d. Student net  (Student\_model)  KD from a | 0.7980 | 1047574 | 256779 | student\_model\_from\_scrateacher.bin |
| **e. Student net**  **(Student\_model)**  **KD from b** | **0.8213** | **1047574** | **256779** | **student\_model\_200ep.bin** |

使用第一題描述的KD方式分別以兩種不同的teacher model weight的去訓練第一題的student model，並和train from scratch 做比較，其中都train 200個epoch，都使用Adam optimizer with lr=0.001。由結果可以看出，很明顯的雖然對student model train from scratch的效果還可以，但還是有teacher model做KD的準確性更高，而且還必須是好的teacher model效果才會比較好。原因就如同老師上課所講的，第一，當data不是很乾淨的時候，對一般的model來說他是個noise，只會干擾學習。透過去學習其他大model預測的logits會比較好，能朝已知對的方向收斂。第二，可以給予label和label之間的關連，teacher model有每個分類的score，而不是像ground truth的one hot encoding，這可以引導小model去學習，讓學習的效果更好。例如在數字辨識數字8可能就和6,9,0有關係。因此有teacher model會比train from scratch好，而teacher model本身的分類效果(accuracy)也必須要好，才能夠正確引導student model往好的方向收斂，所以KD from b會比KD from a好。

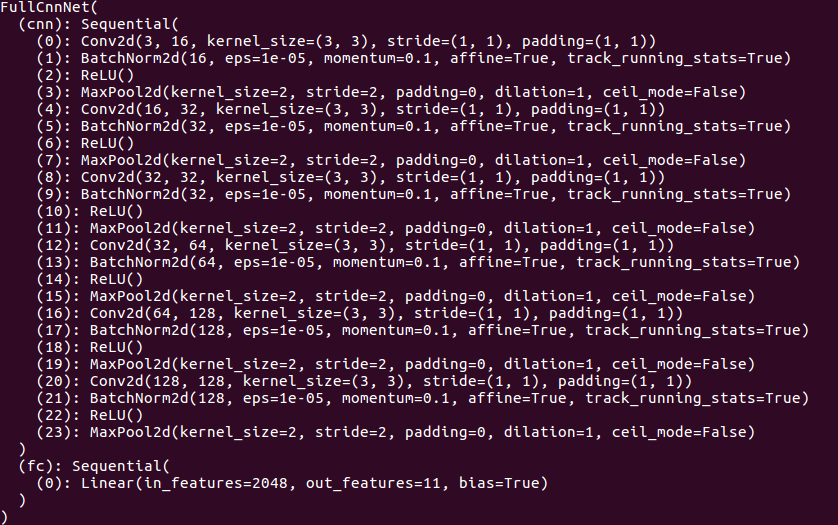
3. [Network Pruning] 請使用兩種以上的 pruning rate 畫出 X 軸為參數量，Y 軸為 validation accuracy 的折線圖。你的圖上應該會有兩條以上的折線。(2%)

(Skip) (refer to hw slide)

被prune掉的neuron越多，雖然model的大小變小了運算速度也會比較快，但accuracy也會有顯著的下降，因為被prune過的model並不能完全代表原本的model，且刪掉neuron的依據是按重要性，刪越多(rate越大)越有可能刪到重要的，因此當一直迭代下去越剪枝越多(rate\*rate\*rate...)，pruned model和原model的失真就越大。

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%)

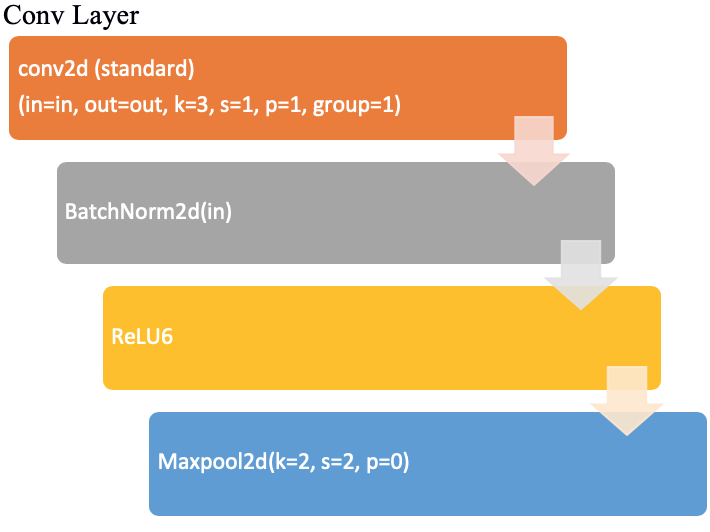
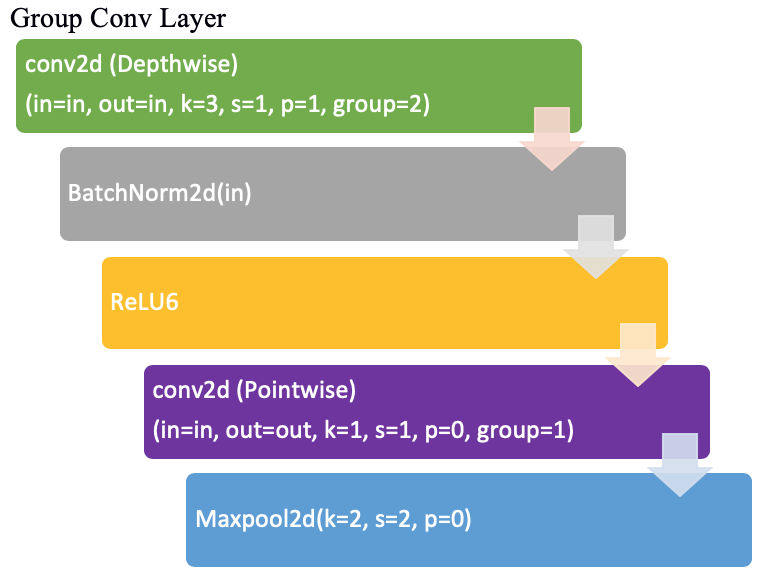
a. 原始 CNN model (用一般的 Convolution Layer)



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

(參見第一題的Student model)

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)

而下方table 則是上面三個model train from scratch 100ep with Adam lr=0.001的結果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Validation acc | Model size (B) | Paramters | Model structure |
| Standard Convolution Layer | 0.7615 | 1120786 | 277611 | 見4.a. |
| Depthwise & Pointwise Convolution Layer | 0.7834 | 1047430 | 224011 | 見4.b. |
| Group Convolution Layer | 0.7429 | 958468 | 236571 | 見4.c. |

一般的convolution layer都是在講group=1的時候，即一個filter的大小就是kernal\_size\* kernal\_size\*input channel數，參數量多。而group=input channel的時候，一個filter的大小是kernal\_size\* kernal\_size\*1，如果搭配output channel數 = input channel數的話，即是第一題提到的Depthwise的convolution ，搭配Pointwise使用的話，要達到相同效果的convolution相較一般法參數量可以減少很多。group=2的意思則介於兩者之間，一個filter的大小是kernal\_size\* kernal\_size\* input channel數/2，參數量介於兩者之間，和原本的一般法之間比較沒有直接的等效關係。

而上邊表格紀錄了，所設計的三種不樣的convolution model的performance，其中D&P的方法表現得最好，因為在一樣參數量(model size)的情況下，他所等效(代表)的model實際上是要用更多參數量才能在一般convolution model上實作的，而至於group convolution model的表現原本預期是應該介於一般與D&P之間的，估計是因為model的接法上並沒有考慮太多的設計，沒有像D&P的方法在Depthwise後搭配一個剛好的Pointwise，做出一個可以合理解釋的model，所以performance沒有特別好。

圖源：

Conv Layer

D&P Conv Layer

Group Conv Layer