一、supervise learning

在我的 CNN model 中，input 為一張 32*32 大小的圖片，共有 R、G、B 三種顏色，目標是讓 output 對應到每張圖片的 class。

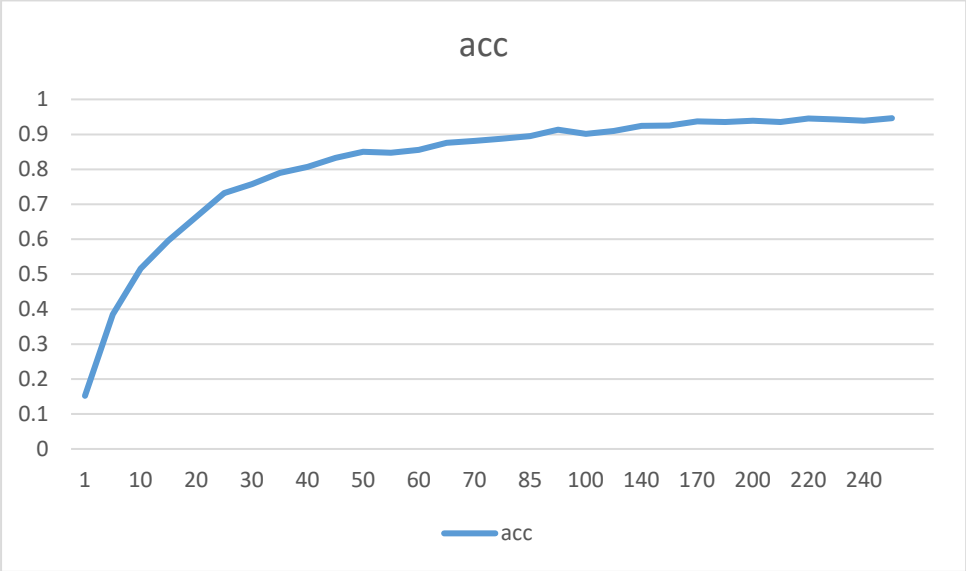
第 1 層 hidden layer 中有 64 個大小為 3*3 的 filter, 使用 relu 作為 activation function。

第 2 層、第 3 層與第 4 層中都有 32 個大小為 3*3 的 filter,兩層均使用 relu 作為 activation function, 且 dropout=0.25

而第 5 層與第 6 層的是 DNN，分別使用 relu 以及 softmax 作為 activation function，並且將維度轉成 512 再轉成 10，也就是我們要的 class 的總數量。

在更新參數使用 adam，計算 loss 的時候使用 cross entropy

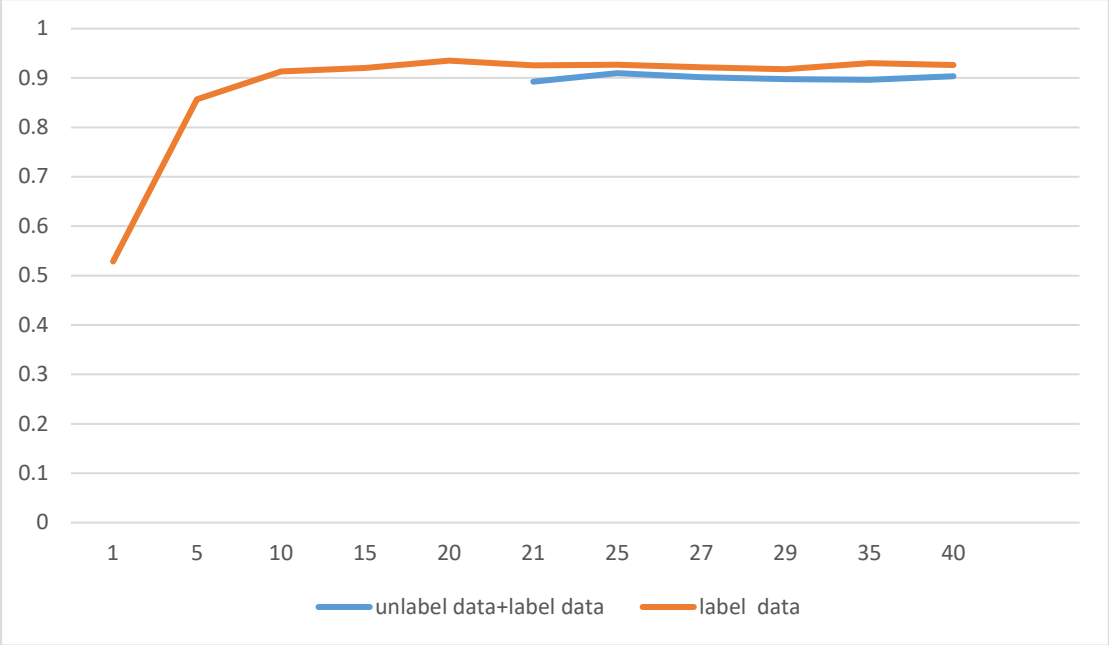
1. learning rate=0.05, epoch=350。



這個 model 在 train 到 250 個 epoch 時，並沒有跑出一個太好的分數，甚至比 200 epoch 時的分數還要低，合理懷疑應該是已經 overfitting.

二、semi-supervise learning--selftraining

在我的 Semi-supervise 當中，先利用 label data train 出一個 supervise 的 model，model 每次執行時的 epoch 為 10 次，在我的方法中，我先重複將 label data 丟到這個 model 20 次，再將 unlabel data 放進已經 train 好的 model 裡，此時會在 output 中找出任一欄位值大於 0.95 的 unlabel data，將他假設為該欄位所屬的 class 的 label data，再將這些假設的 label data 與原本就是 labeled 的 data 一起放到 model 裡 train，希望可以藉由更多的 label data 得到更好的 performance.



然而在 performance 方面，train 到一定的圈數後，acc 就很難再上升。

三、semi-supervise learning—autoencoder+kmeans clustering

autoencoder 最主要的目的在於將一個高維的向量，將它降維，取出這些 data 代表的特徵，在同樣的 label 中的 data 會有相同的特徵，藉由 kmeans clustering 可以觀察出他們在 latent space 中會有分群的關係，利用這些分群的關係，若我們也將 unlabel data 放入同樣的流程內，也可以從它們靠近哪一群去猜測他的 label。在我的時 model 中，autoencoder+kmeans clustering 的成績不是很好，可能是因為我 train 的時間沒有很久，所以沒能達到很好的 performance.

四、compare my result

在 supervise learning 當中我們只需要把 label data 整理成 keras 要吃的格式就可以順利 train 出 model，但也因為 label data 只有 5000 筆，所以達不到太好的 performance，更不要為了讓他達到更好的正確率而無上限的增加 training 的次數，這樣只會使 model over fitting, 除此之外改變 learning 似乎也不會對最終的正確率做出太大的影響。

在 semi-supervise 當中的 self-learning，不太好 train，首先我們要先決定到底把 unlabel data 看做是 label data 的 threshold 要設多少，太小會讓 model 往不對的方向更新，太大又沒辦法讓太多的 unlabel data 變成 label data，依據我在 training 的過程試過 threshold=0.80、0.85、0.93、0.98 的觀察，threshold 設越小，acc 會比較快地卡在一個比較小的值，而較大的 threshold 會使 model 的 acc 緩慢地逼近最後卡在一個比較大的值。再利用 unlabel data 生成的 label data 也要特別注意，不要可以太依賴特定一群 unlabel data 生成的 label data，就算他已經是超過 threshold 的 data，在我 training 的過程發現，如果一剛開始沒有利用 label data 先做好一個比較可靠的 model 或是在之後都只使用 unlabel data 生成的 label data 做 training 都部會有太好的分數，在 unlabel data 生成的 label data 裡面若是有錯誤的 label 會使 model 更新的方向錯誤，又會使下一批 unlabel data 生成的 label data 產生更多的錯誤，進入一個惡性循環。

在 semi-supervise 當中的 autoencoder+kmeans，遇到的問題也與 self-learning 差不多，如果要重複的使用預測過的 unlabel data 當作 label data 使用，我們必須確定他有一定的可信度，否則將無法得到一個最佳的結果。