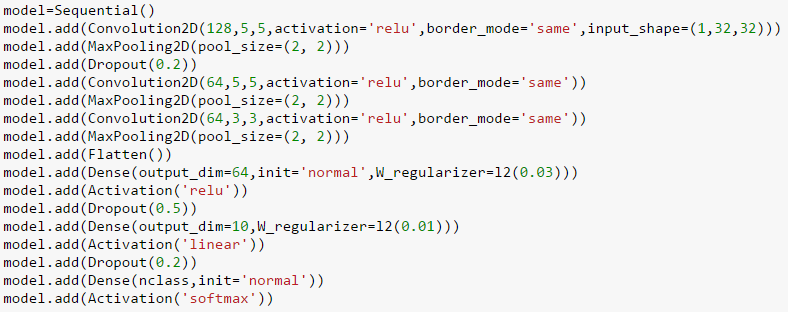
HW3

1. **Supervised learning**:

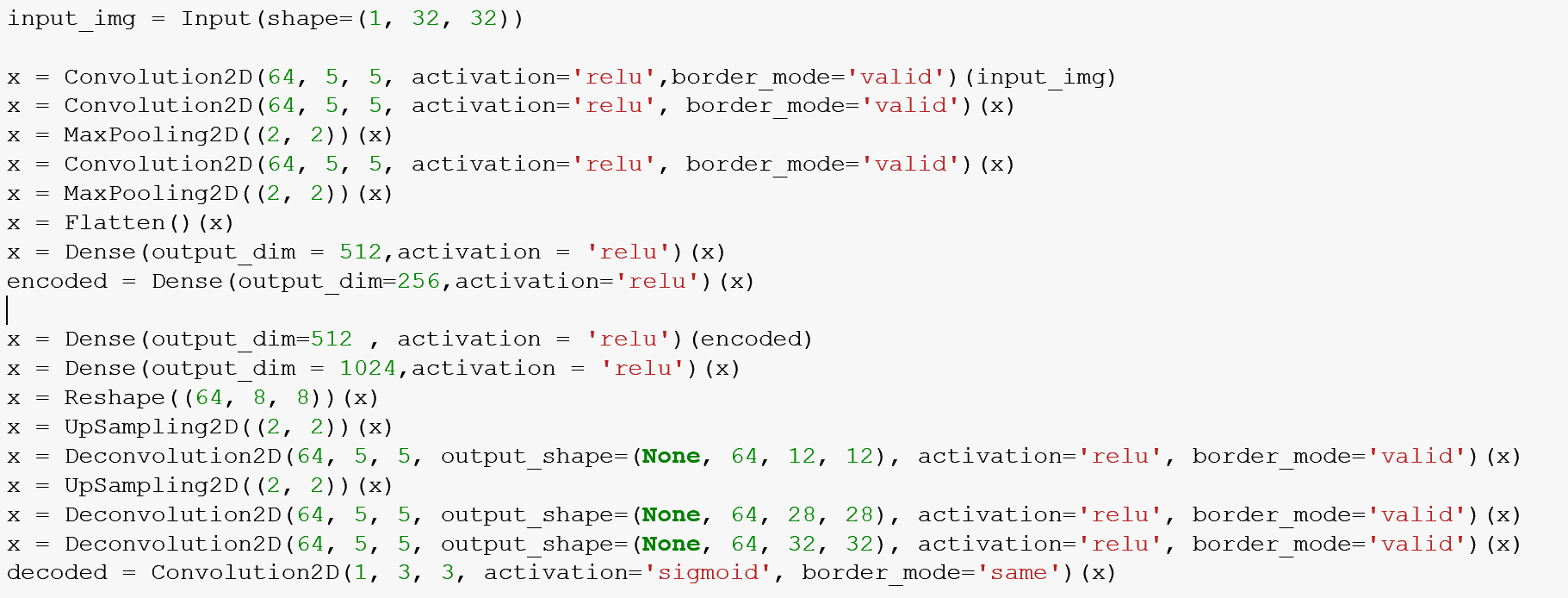
　一開始是直接將labeled data (RGB image)拿去train，test data set predict的結果，accuracy在public set為0.38而在private set為0.39，但是若將image先做灰階後再丟去train，結果在public set為0.46而在private set為0.47，提升了不少，CNN的架構由下圖程式碼表示 ( 使用灰階image )，loss function 採用category crossentropy，而optimizer使用adam。

1. **Semi-supervised learning (1)**

第一個方法使用的是self-training，先train labeled data得到model，再來拿這model去predict unlabeled data 得到的每筆unlabeled data屬於各個class的機率並計算entropy，將entropy小於0.8的unlabeled data進行label後與原本labeled data一起再丟回model去train，得到新的model後再拿剩下的unlabeled data (entropy>0.8) 去predict，若predict出來的結果entropy小於0.8的unlabeled data不到2000筆，則此model為最終model，若大於2000筆則繼續重覆此流程。

1. **Semi-supervised learning (2)**

第二個方法使用的是cluster by autoencoder，下圖是convolutional autoencoder的結構，loss function 採用category crossentropy，而optimizer使用adam。



將labeled data與unlabeled data共50000筆data，先做完灰階化 (維度變成(1,32,32) )，再都丟進此mode進行training，但由於實在train得太慢所以loss值到達0.5772時我就將其暫停，而此時predict的結果大概由下面兩張圖所示，第一張為input的image，而第二張為predict的image，可見還相差許多，但predict 的image中物件的輪廓已相當明顯。

C:\Users\gary\AppData\Local\Microsoft\Windows\INetCache\Content.Word\input.pngC:\Users\gary\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output.png

再來，將所有data做encode時，得到長度為256的code，發現每筆data的code中大約有1/2為0，這實在非常奇妙，所以我再將所有data的code用PCA做降維成長度為128的code，並將所有data依照這128去做kmeans clustering，而n\_cluster設為10，執行後發現，5000筆labeled data有絕大部分都被分在同一群，之後又再試試Spectral Clustering，發現class 5 的label data 在三個群組裡，相較於其他class的label data，是出現最多次的，由此可見autoencoder 的training還不夠完美，使得encode 的code之識別能力還不夠強。由於最後無法完美的clustering，所以第二個方法沒有做完，所以沒有上傳method2的code。

1. **Compare and analyze your results**

在self-training方法中，直接使用RGB image去做的話，得到的model於test data set predict的結果，accuracy在public set為0.5而在private set也為0.5，但若以灰階的image做的話，public set為0.46而private set也為0.46，反而沒有improve原本supervised的model，可能是entropy的threshold設太小了，使沒有足夠的unlabeled data來improve model。而github上傳的為使用RGB的code。