**Pattern Recognition**

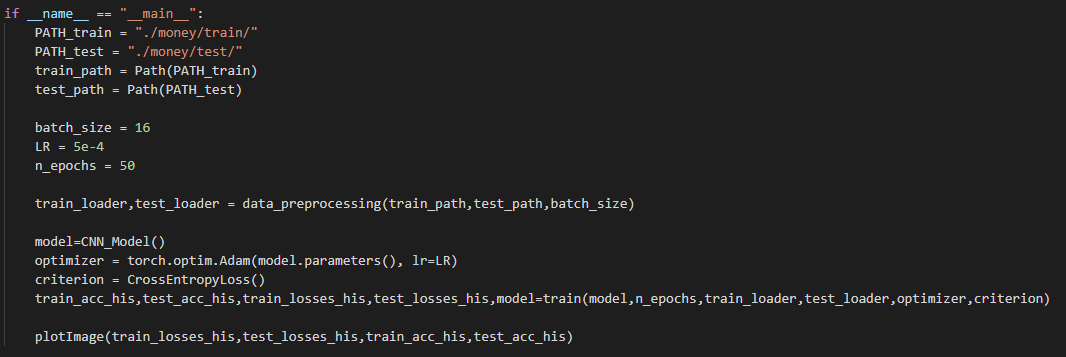
7108056103 彭顯詠

1. **作業說明(HW3)**

Image classification with convolutional neural networks, you should use at least two techniques of the following to improve the accuracy, (1) data argumentation (2) regularization -weight decay (3) dropout (4) transfer learning.

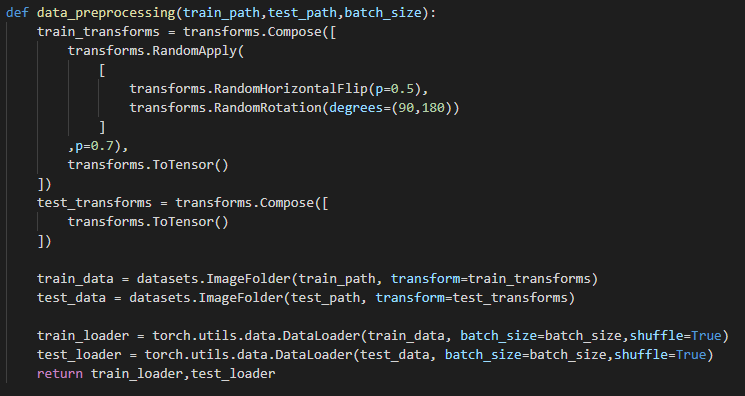
1. **Main function**

在CNN搭建的部分使用了我較熟悉且就常使用的pytorch其中的torchvision可用來對影像相關的深度學習做使用，先將資料切分為50% training data和50% testing data，最後結果以batch\_size=16，learning rate=5e-4，訓練50個epoch獲得較好成果。



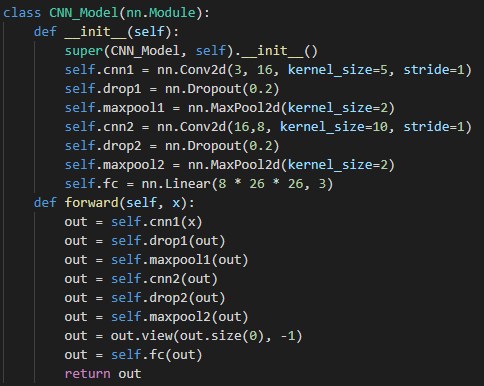
1. **Data Augmentation**

資料增量的部分使用了pytorch的transform套件實作，我們使用RandomApply 機率為0.7，增量部分則以RandomHorizontalFlip與RandomRotation做增量，testing data則維持不變。

****

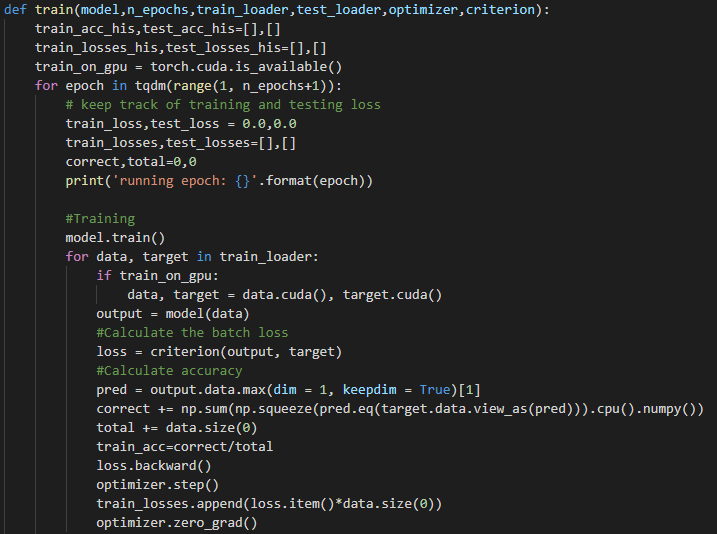
1. **CNN model**

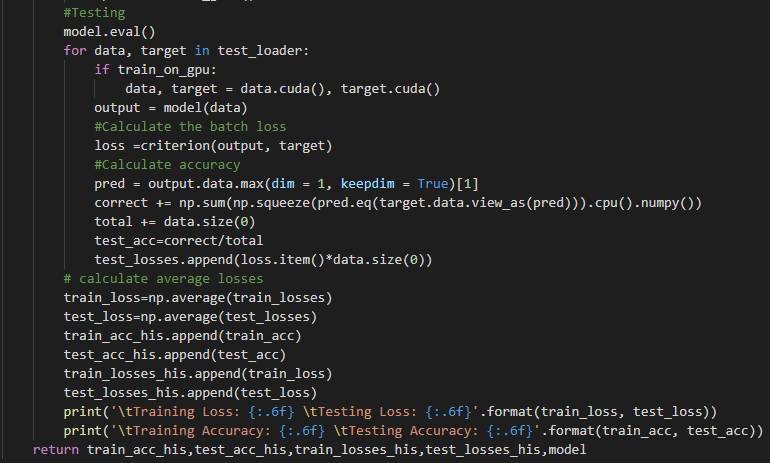
Model部分我使用了兩層CNN網路，中間都將20%部分dropout以避免overfitting的情況發生，採用max pooling，kernel size設為2，最後以一層linear網路將結果分成三類，取其最高者作為預測的類別。

****

1. **Training階段**

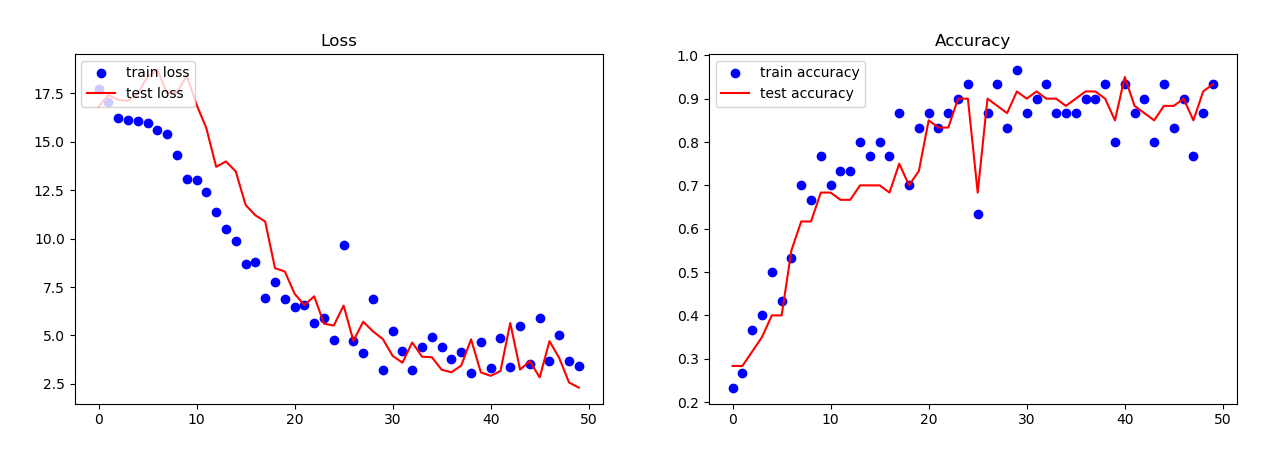
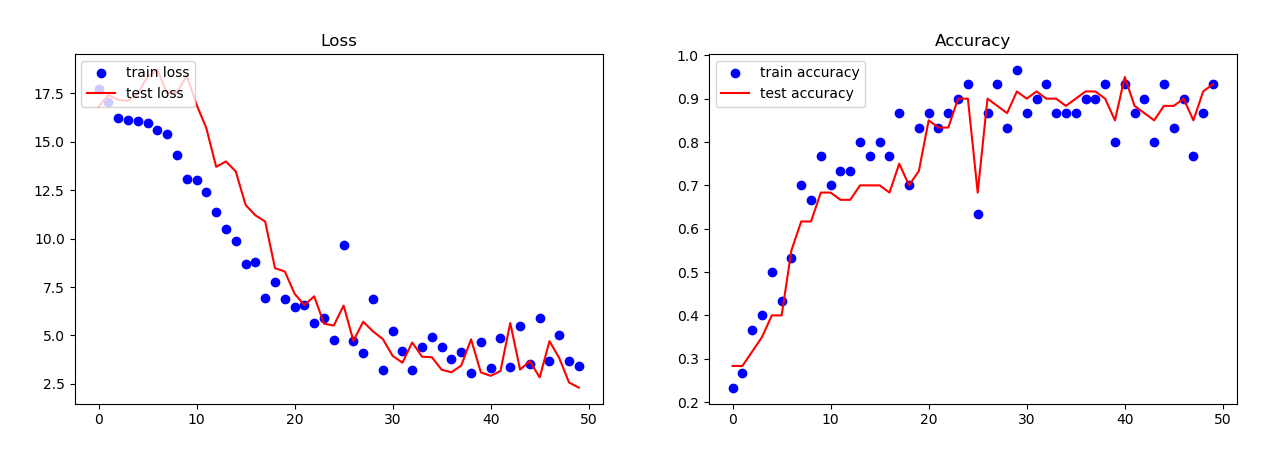
訓練過程中會將資料丟入model中做training，並在每個epoch訓練結束後，將資料做evaluation並且分別印出此時的loss及accuracy以檢查此時是否已經overfitting。

****

****

1. **結果**

本次的鈔票分類任務，training loss與testing loss在epoch=50時分別達到3.4以及2.3，且在testing accuracy達到0.933，顯見大部分資料皆能分類正確，並且在loss趨勢研判，loss已經無法再下降。



1. **結論**

本次作業在實作上來說並不困難，然而相較於keras來說，pytorch使用上中文的資源較少，實際使用上來說並不會比keras複雜，雖然code長度較長，以及調整參數時須注意model丟入的參數格式必須相連接，卻也比較容易理解model中的各項細節處理，可以微調的部分也更多。