**Simple Experiment**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Linear Model | My CNN Model | TensorFlow CNN Model |
| Epochs | 50 | 50 | 50 |
| Training time | 5.4s | 97m9.5s | 18.3s |
| Accuracy | Train:0.66 Valid:0.65 | Train:0.74 Valid:0.71 | Train:0.77 Valid:0.79 |
| Number of parameters | 32833 | 7221 | 64625 |
| Training loss curve |  |  |  |

**Advanced part design:**

在advanced part的model架構上，基本上是沿用basic part的部分，一樣是兩層convolutional + pooling，但在dense layer上選擇了三層relu和最後一層sigmoid。而在細節上，參數對應basic有做調整，在convolutional layer轉換到dense layer時，中間有做Dropout 40%的units，避免units過多導致overfitting，而在dense layer之間，都有做batch normalization。最後在compile model時， optimizer根據網路上最常用的是adam和SGD兩種，在經由多次嘗試後，最終仍選擇adam作為model的optimizer，會這麼選擇可能有一部分原因是因為自己的英文名字就是Adam吧XD。而loss function則因為最後一層只有一個unit且為sigmoid，所以選擇的是binary crossentropy。在資料處理上，我有先運用data augmentation將資料集擴充，但因為本身的dataset就沒有很大且圖片的size也偏小，為了保證data仍能辨識，能做的變化其實不多，而就在做完data augmentation後，初期的training反而loss升高，但到後期就逐漸升高，能train的epochs也變多，相比沒有做augmentation前，並不會那麼容易就overfitting，整個model的泛性有所提高。

**Summary:**

此次CNN的作業相比前幾次打code的量又變得更少了，基本上都是建立於上一次作業的neural network，再在前面增加convolutional network而已，而且在實作上，因為沒有要讓我們實作backpropgation，僅有實作forward和initial等較為簡單的事情，並且template中給出的資訊再搭配上課講義，在打code上面的進度是比較順利的，只是因為圖片維度在經過filters之後會有所改變，這部分的維度對應上卻也花了不少時間，但若真的要講到最花時間的事情，應該是CNN是如何運作的原理吧，因為其想法有點過於抽象，要說服自己那就是一堆數字轉換來轉換去，最後得出一個結果，這個過程是不太容易的。

而後advanced的部分，因為可以使用TensorFlow，在架構model的部分上更是快速，在google如何使用TensorFlow去建立CNN後，按照網上的教學大概幾十分鐘內就寫好了，但為了要使用GPU加速，架設環境上卻是煞費苦工，前前後後大概花了好幾個小時才將環境架好，各種package的衝突和版本不合，安裝了又卸載，反反復復真的一度想要放棄，後來想到多花點時間，省下的卻是training那漫長的等待過程，就咬著牙繼續架設，所幸最後還是有成功，看到training的時間被壓縮了幾十倍，頓時覺得自己也是一個ML的工程師了。

此次的作業不僅讓我接觸到時下最熱門的CNN model，也就是電腦視覺的部分，也讓我開始使用python community的package後，感覺好像已經成為了這個圈子的一份子，覺得倍感成就，接下來所剩差不多一個月的時間要與組員們一起努力奮鬥final project，也期待可以在這門課上完美的畫下一個句點。