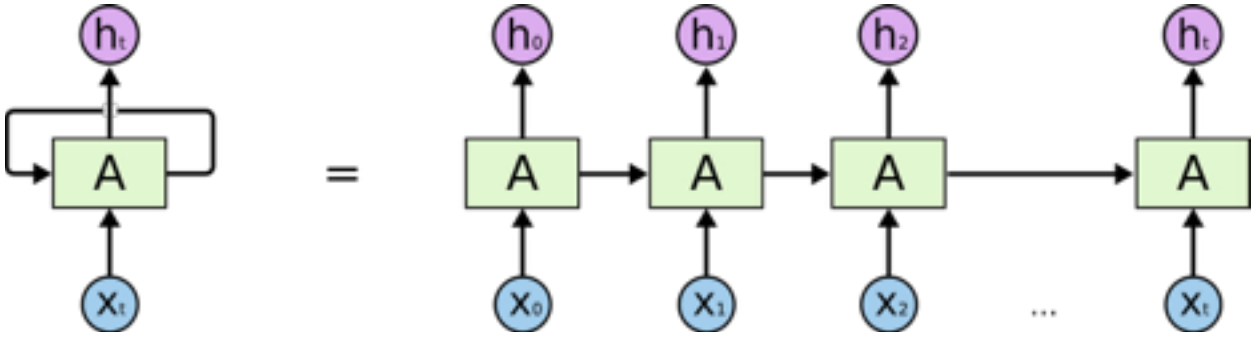
HW1 Report

資管碩一 R06725053 韓宏光

1. Model description

(1) RNN (1%)

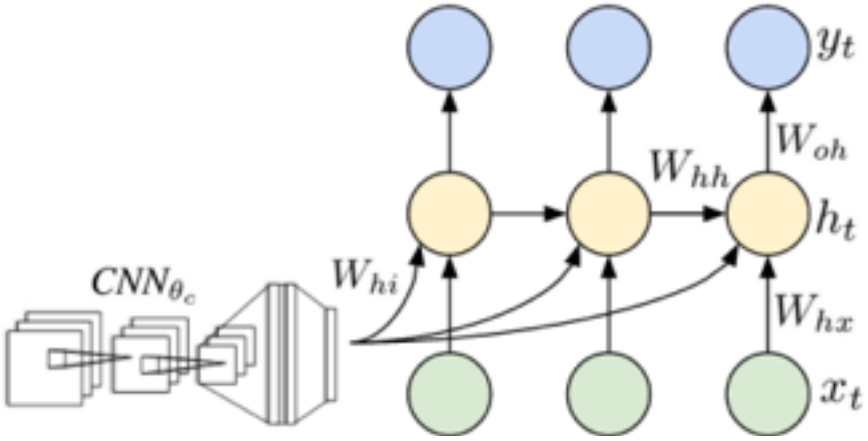
RNN(遞歸神經網絡)是時間遞歸神經網絡（recurrent neural network），由不同曾的神經元間互相連接構成為一個矩陣。相較傳統單純的類神經網路(neural network)，RNN透過將上一層的輸出便作下一層的輸入輸進模型，得到擁有前面輸入資料的資訊關聯。
RNN可以描述動態時間行為，因為和傳統類神經前饋網絡（feedforward neural network）接受較特定結構的輸入不同，RNN將狀態在自身網絡中循環傳遞，因此可以接受更廣泛的時間序列結構輸入。



然而最簡單的RNN無法處理因為隨著遞歸，權重指數級爆炸或消失的問題（Vanishing gradient problem），難以捕捉長期時間關聯；而LSTM使用了記憶閘、遺忘閘可以很好地解決這個問題。

(2) RNN+CNN (1%)

卷積神經網路（Convolutional Neural Network, CNN）是一種神經網絡，透過用filter一部分覆蓋範圍內的周圍單元，來找特定的特徵以讓模型重複利用。
CNN由一個或多個可供調整大小、橫跨範圍的filters所組成，同時也包括pooling layer來縮減特徵的複雜度。這一結構使得CNN能夠利用輸入數據的二維結構，在相比較不同類型的類神經網路，CNN需要考量的參數更少。
再透過CNN的filter取出關鍵特徵後就可以透過RNN進行進一步的訓練，不僅提升訓練的速度、也更可以降低模型的冗余複雜度。



2. How to improve your performance

(1) Describe the model or technique

我的model不像大家針對每個句子做padding後來當作sequence輸入進RNN，我是使用一個TImeWindow其長度為31，在同一個sentence中將每個frame前後共31個frame(若無相對應的frame則補0)結合成為一個sequence，舉例：Sentence1有3個frame：[A,B,C], Sentence2有2個frame[D,E]，在經過前處理(TimeWindow)後會變成Sequence1：[0,A,B], Sequence2:[A,B,C],Sequence3:[B,C,0],Sequence4:[0,D,E],Sequence5:[D,E,0]。

我也針對這些輸入的Sequence先是逐一進行正規劃(BatchNormalization)、以8個3*3大小的filters輸入進CNN，並以MaxPooling的方式提取特徵、再輸入進雙向Bidirectional LSTM(為多對一的LSTM結構)，最後經過兩層512大小的full connected layer後得到結果

(2) Why do you use it

我認為與其直接由padding句子來作為sequence去訓練，並不如直接框一個timeWindow去做預測，因為在這個timeWindow內可以更強烈地依據附近的frame來做判斷phone。

接下來運用CNN以擷取這段TimeWindow特定的特徵。

使用多對一架構的LSTM除了解決vanishing的問題外，也是因為LSTM在經過前後的frame訓練後，將可以更好的判斷當下的frame。

最後為了提升模型複雜度多加了兩層的full connected layer。

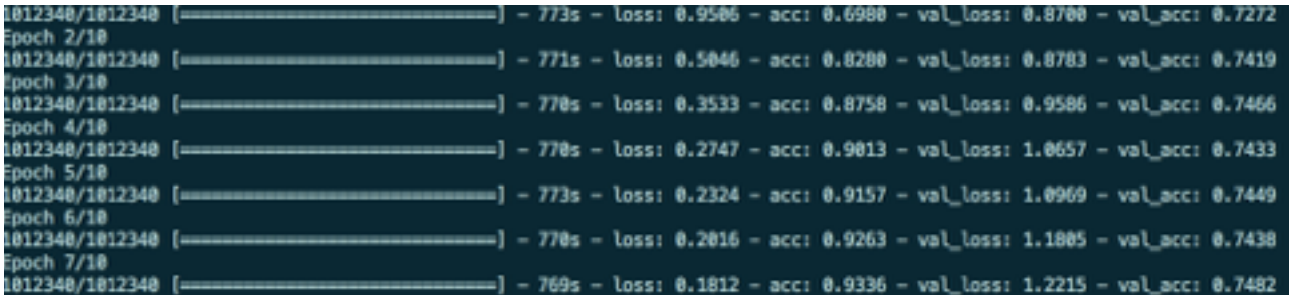
3. Experimental results and settings

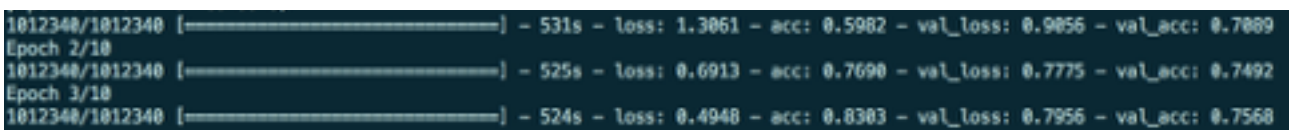
(1) Compare and analyze the results between RNN and CNN

第一張圖與第二張圖分別為RNN與CNN的訓練過程，可以觀察出CNN明顯在表現上勝過RNN一籌，在耗時上也明顯較短。他們兩個的共通點在於當訓練了短短數次後，val_loss便開始拉高而不再降低。

我想這除了有著overfitting的因素參雜其中，也跟我所選擇的TimeWindow大小以及模型複雜度有關。

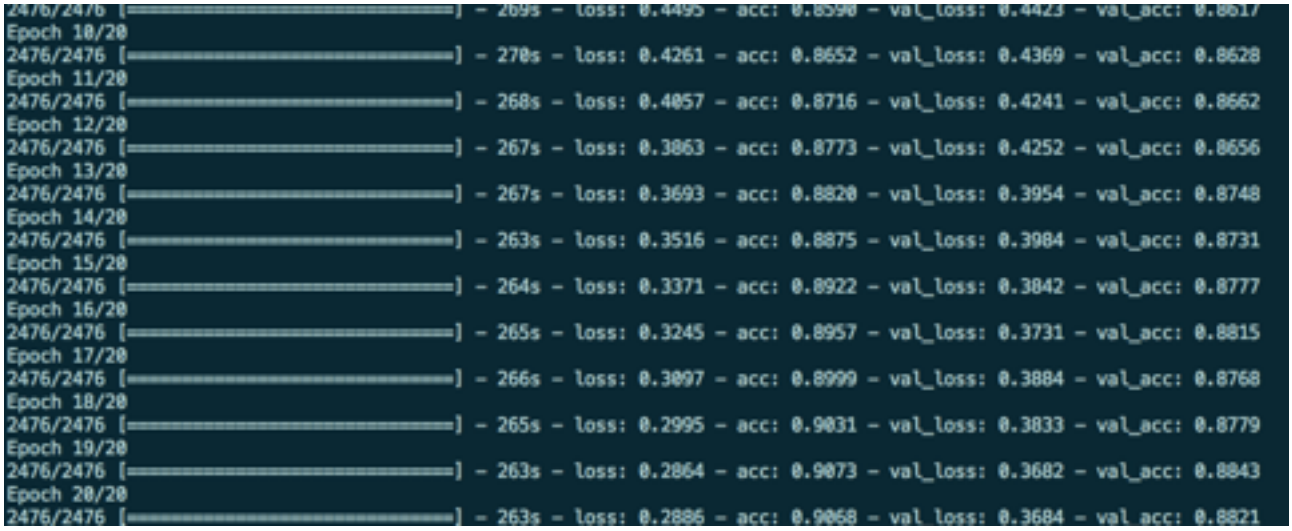
我認為CNN的表現優於RNN的原因主要是因為我選到的filter成功抓取到了frame之間的隱性關聯。





(2) Compare and analyze the results with other models (other models can be variant of basic RNN, like LSTM, or some novel ideas you use)

我曾經使用padding sentence來作為sequence輸入，並搭配建範圍為3的TimeWindow來進行CNN特徵擷取。乍看之下訓練效果不錯(88% validation accuracy及0.4的error)



但是在經過後處理後在kaggle上最好的表現也在17, 18上下徘徊。

以下為我的改善及推論過程：

I. 在最一開始我發現validation accuracy表現大概有86%但error很高(3點多)，雖然有加Mask layer，但後來發現應該是因為在經過CNN那一層後，由於CNN會帶出bias項目，因此原先設定Mask的遮罩值(0)變失效了，再增加了sample_weight後，error降到0.6
II. 後來認為應該是沒有亂數初始化(np.random.seed)以及正規化(BatchNormalization layer)，加上去之後儘管accuracy有了些許上升，但在kaggle上仍然成效不彰。
III. 由於我的accuracy不錯加上error很低，因此助教建議我使用39維度的label class，但我在使用後在kaggle上的表現反而變糟了，我覺得accuracy高應該只是我的模型成功預測後面一大串padding都為0而造成的假象。

我試著尋找過原因，但因為時限而無法得出確切的結論，若有時間，我接下來會再往增加模型複雜度與結合fbank與mfcc兩種維度的feature產出的結果去做分析跟比對。