ADLxMLDS A1

孫奧 R05922147

1 Model description

1.1 RNN (1%)

RNN 是用 keras 來 implement 的，其中用到的 RNN 是 Bi-GRU。之所以用到 GRU 是參考論文 ***Improving speech recognition by revising gated recurrent units***，此外，論文中建議將 reset-gate 去掉，不過由於時間關係很遺憾。

由於我的 best_model 是 bagging 得到的，所以對於 RNN 的參數我沒有特別去挑戰，簡單的架構大概是從 input 進來，經過 mask-layer 以後，後面接上了 Bi-GRU，hidden unit 數量 256，dropout=0.2，有趣的是，論文中建議 recurrent state 的轉換用 ReLU，但是我做實驗的過程中就直接爆了(keras 中設 recurrent_activation)

此外在實作上，每一次 valid-data 只選取了總共 3696 中的 37 個(1%)，雖然看起來太少了，但是在絕對 frame 數量上以及不少了，而實際應用上，valid 的結果和 kaggle 的結果非常一致，並且在 valid 的中，並不是只觀察 val_acc 而是寫了 callback 來觀察 edit_distance 和 PER(edit_distance 除以正確字符串的長度)，

1.2 RNN+CNN (1%)

RNN 沿用了 1.1 中的架構。前面接上的 CNN,CNN 使用的是 Keras 的 Conv1D，filter 數量是 1024，kernel_size 是 7，沒有特別的去調 CNN，因為對於時間尺度上的卷積以及 MFCC 中的 feature 加在一起的物理意義不是特別明顯（好像是有時間上的加上一些導數二階導數等等），所以只是實作了模型，沒有去調參數。

2 How to improve your performance

2.1 Write down the method that makes you outstanding

是用 bagging 和一個 trim 的 trick。

2.2 Describe the model or technique (0.5%)

第一个，是用 bagging 的方式将训练的很多模型丢在一起，用投票的方式来决定分类。這大大的提升了模型的表現，不過在用這個方法的時候距離截止時間已經很近了，如果可以還有時間，會使用更多的方法去增加模型的 diversity：用 mfcc 的模型同時用 fbank 的

模型: 輸出 592*48 的預測的類別幾率矩陣, 便可以將很不同的模型也都 ensemble 在一起。

另一個 trick：将 aabaa 这样的转换成 aaaa,详细可参见 hw1_data_utils.py 中的 super_trim 的 function,並且在嘗試多次以後，應用了很多次 super_trim,（之所以用很多次是因為原本函數只考慮到不同的前後三組，考慮的不夠仔細，用多次以後可以減少不嚴謹，從結果來看），這個想法是同學教我的，沒想到有非常大、明顯的提升。這個也讓我稍稍的點失望：這更像是一門 kaggle-trick 的作業了，好像距離深度學習神經網絡架構、前沿知識、基礎理論差的蠻多的

2.3 Why do you use it (0.5%)

Bagging 的原因很簡單，因為單個的模型我沒有辦法再承擔更深的模型、更多的參數了，所以就利用手頭有的模型，而 bagging 則是最簡單又有效的一種。

至於 trim 的 trick，那個是同學告知想法，然後自己做測試的出來的。

3 Experimental results and settings

3.1 Compare and analyze the results between RNN and CNN (0.5%)

如前面所說，在 CNN 的部分沒有特別去調（本身用到的 MFCC 可能不是特別適合 CNN），所以在表現上，在 MFCC 數據集上 RNN 是完爆 CNN 的。

此外，由於 keras 使用了 Conve1D 以後，masking 無法使用，只使用 sample_weight 會使得 acc 有錯所以很難根據 acc 來評價，不過從 edit_distance 的角度來說，的確是差了很遠。

3.2 Compare and analyze the results with other models (0.5%)

有用到 MLP，即完全沒有時間訊息，想看一下沒有時間訊息情況下