ADLxMLDS2017 - HW1

電信碩一 宋易霖 r06942076

1 Problem Define

根據觀察到的一串 phone sequence 的特徵來預測該 phone sequence。

2 Dataset

fbank (69個特徵), mfcc (39個特徵)。分別共有3696筆資料，其中最長的sequence爲777。

3 Model Description

3.1 RNN

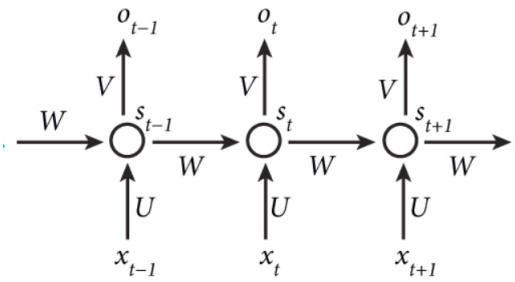

Figure 1: rnn model (reference: *http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/*)

RNN的結構如Figure 1所示，其中 $x_1, x_2, ..., x_n$(n 爲sequence長度)是每個phone的特徵, 經過以下運算得到每個hidden state

$$s_t = tanh(Ux_t + b_U + Ws_{t-1} + b_W)$$

得到hidden state後，接上fully connected layer和softmax來預測最有可能的phone是甚麼。

$$predict_t = softmax(Vs_t + b_V)$$

3.2 CNN + RNN

CNN主要用來對X取出一些區域性的特徵, 再將這些特徵當作新的X餵給RNN。而kernel的寬度儘量不會跨太多的frame，因爲後面還會再用RNN取出時間性的特徵，在這邊就專心對每個frame各自的特徵取出區域特徵即可。

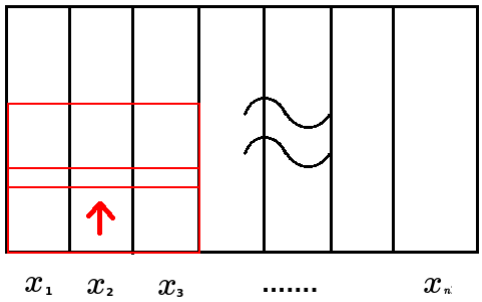

Figure 2: CNN block

4 How to Improve Your Performance

4.1 LSTM

因爲傳統的RNN在sequence很長的時候容易遺忘較前面的資訊，因此利用LSTM能夠有較好的表現。其中每個變數的求法如下：
記住前面部份的比例f_t:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

新進的資訊 $i_t * \tilde{C_T}$：

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$
$$\tilde{C_t} = tanh(W_C[h_{t-1}, x_t]) + b_C)$$

計算要傳給下個frame和計算hidden state的資訊：

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t}$$

hidden state h_t:

$$o_t = \sigma(W_o[h_{t-1}, x_t]) + b_o$$
$$h_t = o_t * tanh(C_t)$$

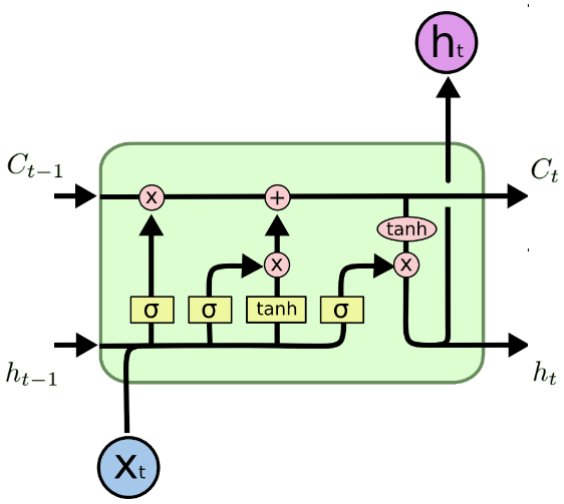

Figure 3: LSTM block (reference: *http://colah.github.io/posts/2015-08-Understanding-LSTMs/*)

4.2 [-1, 1] Normalization

在rnn和lstm的模型中，會將前一層輸出和現在的輸入接起來在繼續運算(參考4.1的式子)，而前一層的輸出因爲經過了tanh函數數值介於−1和1之間，因此我把輸入都標準化到−1和1之間，輸出和輸入數值之間就會比較相近，訓練結果也較好。詳細標準化作法如下

for i in features of X: do

$$middle_i = \frac{max(X_i) + min(X_i)}{2}$$

$$range_i = \frac{max(X_i) - min(X_i)}{2}$$

$$X_{i,normalized} = \frac{X - middle_i}{range_i}$$

4.3 Smoothing

在預測出來的phone sequence中(尚未去除重複以前)，連續同樣的phone中間不太可能出現一個不一樣的，所以在去除掉重複的phone以前，會先做一次smoothing。方法 是用一個sliding window掃過處理前phone sequence中每個phone，在這個window裏面出現最多的phone替換掉現在window中間那個phone，如果有相同出現次數的phone 的話，以靠左邊的爲主。參考Figure 4

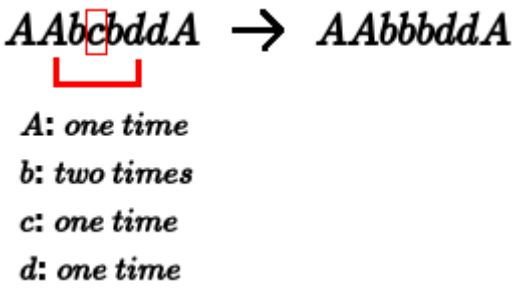

Figure 4: smoothing process

5 Experimental Results and Settings

model type	edit distance		
	training set	validation set	test set
RNN (both, w_s = 5, [-1, 1])	11.82	13.22	10.98
RNN + CNN (both, w_s = 5, [-1, 1])	20.19	20.53	17.10

Table 1: RNN及RNN + CNN的edit distance比較, model type欄位的格式爲： model (使用的特徵: [fbank, mfcc, both]其中一個, window size, 標準化方法：[standardization(std), [-1, 1]]其中一個)

以上的實驗中，RNN的hidden layer爲3層，每層有64個node和dropout=0.5，並且是bidirectional的。 CNN則是有兩層，兩層的kernel size分別是 (3, 3), (1, 3)。 依照3.2的想法，kernel不要跨很多frame，這邊是第1層跨3個，第2層就只有取一個frame而已了(因爲

前一層已經取了前後各一frame的資訊，這邊如果kernel size大於 一的話，就會取到原本前後各二個frame的資訊了)
從Table 1 可以看出，加上了CNN後表現差了不少，可能是因爲資料本身並沒有區域的特性，因此做了CNN後反而讓進入RNN的輸入變得更混亂了。

model type	**edit distance**		
	training set	**validation set**	**test set**
a. RNN (both, w_s = 5, [-1, 1])	11.82	13.22	10.98
b. LSTM (both, w_s = 5, std)	8.19	10.60	8.76
c. LSTM (both, w_s = 5, [-1, 1])	7.12	10.23	8.03
d. LSTM (fbank, w_s = 5, [-1, 1])	7.33	9.95	8.10
e. LSTM (both, no smoothing, [-1, 1])	8.99	12.4	9.32
f. LSTM (both, w_s = 7, [-1, 1])	6.66	9.56	7.68
g. LSTM (both, w_s = 9, [-1, 1])	6.71	9.52	7.76
h. LSTM (both, w_s = 11, [-1, 1])	6.80	9.29	9.09

Table 2: 其他model edit distance比較, model type欄位的格式爲： model (使用的特徵: [fbank, mfcc, both]其中一個, window size, 標準化方法：[standardization(std), [-1, 1]]其中一個)

LSTM的設定都和RNN一樣: hidden layer三層， 每層64個node和dropout=0.5，並且爲bidirectional。
1. 從a, c兩個模型可以看出LSTM比起RNN的表現好上不少, 在validation set和test set上edit distance都少了3左右。
2. 從b, c兩個模型可以發現用4.2提到的normalization比用standardization在validation set和test set的edit distance少了0.4、0.7。
3. 從c, e兩個模型可以發現加上smoothing可以使得模型在validation set和train set上edit distance少了2.2、 1.3。
4. 在c, d兩個模型比較feature的使用：單用fbank和用fbank+mfcc差不了太多，在validation set上雖然好了0.3，但是在test set上稍微輸了一點，最後基於多用一些特徵準沒錯的狀況下還是兩個都用了。
5. 在f ∼ h中試著增加window size，因爲這個步驟和訓練無關，不會有overfitting的問題，所以edit distance的表現應該全部的set一起看，發現在window size爲7的狀況下可以讓整體的edit distance下降最多。

6 Reference

Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs
Understanding LSTM Networks