ML2017FALL Final Project: Listen and Translate

NTU_r06946003_鬼氏企業

組內分工

R06946003	湯忠憲	RNN retrival model
R06922030	傅敏桓	CNN + RNN retrival model
B03902085	吳家謙	Seq2seq model
D05921027	張鈞閔	資料前處理、特徵抽取、實驗統計

問題描述

這份期末專題的目標是希望以機器學習的方法，解決跨語言的語音辨識問題。給定一段台語的聲音訊號特徵 (MFCC，梅爾倒頻譜係數特徵)，我們希望跳過文字翻譯的步驟，直接將台語的音訊辨識輸出成中文字。這個問題已被簡化成四選一的單選題，因此我們提出的模型以檢索模型 (retrieval model) 為主。

資料前處理／特徵抽取

1) 序列襯填 (Zero Padding)

資料集包含長短不等的訓練資料共 45036 筆、測試資料共 2000 筆。其中，最長的音訊樣本 (MFCC) 長度為 246，而最長的字幕共 13 個字。使用 Keras 提供的運算架構時，我們採取序列襯填的方式，將所有序列於前方補 0 至最大長度，得到的每一筆音訊特徵大小為 (246, 39)、字幕長度為 (13,)。

2) 平移負採樣／隨機負採樣 (Negative sampling)

在訓練檢索模型 (retrieval model) 時，訓練資料需要包含正確 (positive) 的配對以及錯誤 (negative) 的配對。我們透過對正確配對的資料做任意平移，讓每一個 MFCC 都對應到錯誤的字幕 (caption) 來產生錯誤配對的資料。每次 epoch 藉由不同的平移距離，可以產生不一樣的負樣本，讓模型學到更多資訊。另外，我們發現正確配對和錯誤配對之間的比例，對模型訓練過程也有不少影響，相關的討論會在後面的章節詳述。

除了以上兩點是訓練檢索模型必須的前處理外，我們也嘗試了以下列舉的方法在訓練資料集上實作資料增強 (data augmentation)：

3) 降採樣 (downsampling)

對任意音訊樣本的 MFCC 特徵序列進行降採樣，以每兩個音框中只取一個的方式，將音訊樣本的採樣率減半，如此原本包含 164 個音框的樣本就會變成 82 個音框，再作為新的訓練資料一起訓練模型。

模型描述

在這次的期末專題中，我們嘗試實作了幾種不同的神經網路模型。在檢索模型的部分，嘗試了 RNN 架構和 CNN + RNN 兩種架構；考慮到更普遍的使用情境 (非選擇題的情況)，也嘗試實作基於 RNN 的 Seq2seq 模型。

1) RNN 檢索模型 (RNN retrieval model)

首先想到以 RNN 模型來抽取時間序列資料的特徵向量。我們採用的 RNN 檢索模型架構如下圖所示，音訊的部分是拿 MFCC 特徵通過兩層雙向 GRU 後，得到其特徵表示 (feature representation) ；配對的字幕則需要先經過一層詞嵌入層，將字幕的每個詞對應到實數詞嵌入向量空間，再通過兩層雙向 GRU 得到字幕的特徵表示。將兩者做向量內積後通過 sigmoid 映射到 [0, 1] 之間，最後得到的輸出代表兩者為正確配對的分數。

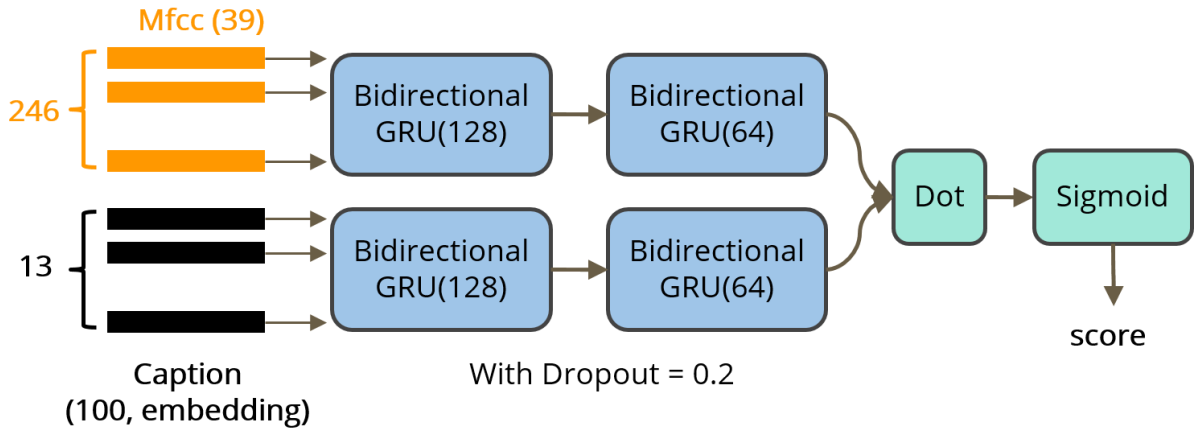

Figure 1. RNN retrieval model architecture

模型訓練的相關超參數 (hyper-parameters) 設置如下表所示：

Loss function	binary cross entropy
Optimizer	Adam, with learning rate 1e-3
Number of epochs	50, with early stopping (patience = 4)
Batch size	512
Training-validation ratio	19 : 1
Word embedding dim	100

Table 1. RNN retrieval model hyer-parameter

2) CNN RNN 混合檢索模型 (CNN-RNN-hybrid retrieval model):

我們也可以透過卷積運算來掌握相鄰的音框之間的交互關係；這可以透過 Keras 的一維的卷積層 (Conv1D) 來實作，好處是可以大幅減少模型訓練的時間和大小。不使用二維卷積的原因是，我們預期 MFCC 的每個維度之間有足夠的獨立性，使用二維卷積反而會把不同區段的特徵混在一起。將音訊樣本的網路架構以 CNN 改寫後，得到的模型架構如下圖所示。模型訓練的超參數設置和前者雷同，但改成訓練 150 個 epoch ，取驗證準確率最高的模型參數。

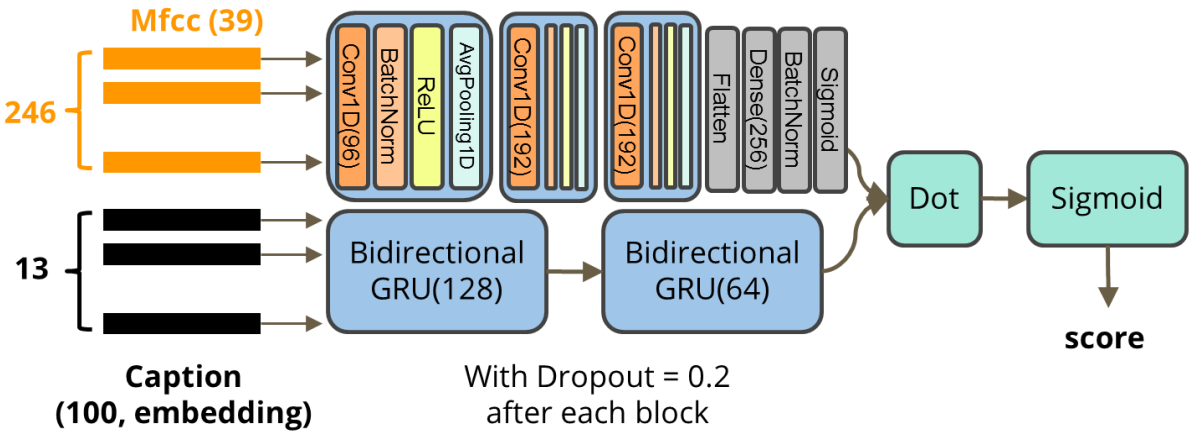

Figure 2. hybrid retrieval model architecture

模型訓練的相關超參數 (hyper-parameters) 設置如下表所示：

Loss function	binary crossentropy
Optimizer	RMSProp, with learning rate 1e-3
Number of epochs	150
Batch size	512
Training-validation ratio	19 : 1
Word embedding dim	100

Table 2. CNN-RNN-hybrid retrieval model hyper-parameter

3) Seq2Seq 模型

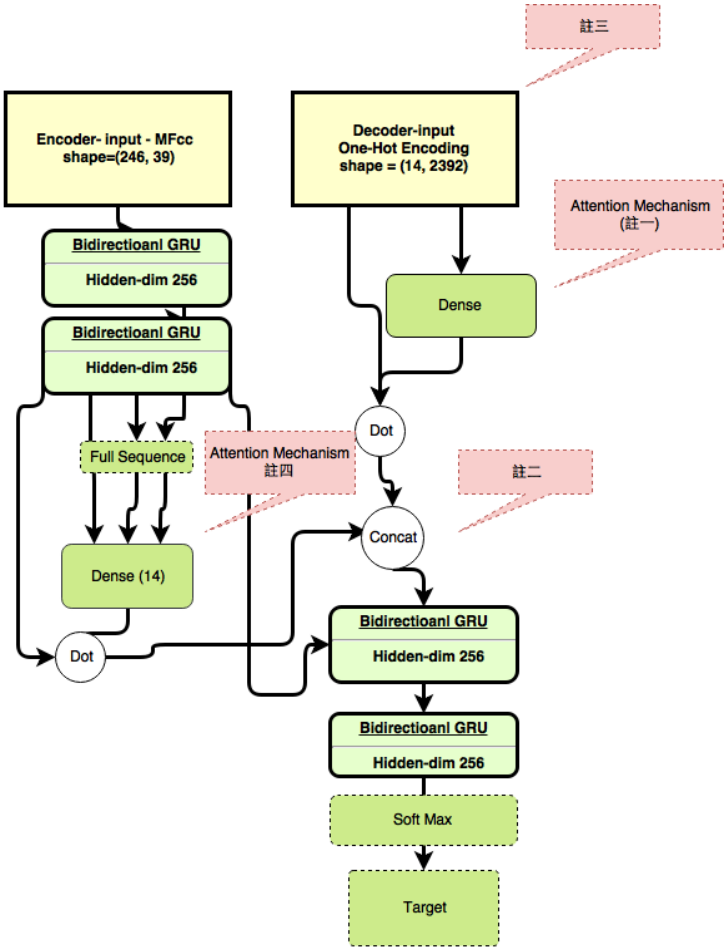

Figure 3. Seq2seq architecture

註一：Attention mechanism 於 Encoder sequence。Performance 提升約 10%。把原先由 attention weight 計算分數之 Multiply 運算改為 Dot Performance 提升約 6%。推測此運算有降維效果，使得 Decoder 分析更順利。

（ reference : https://github.com/philipperemy/keras-attention-mechanism ）

註二：Decoder 最後一層 layer 的 Full Sequence 過 SoftMax 降維後與 Encoder Input Concate 後再餵進 Decoder。Performance 提升約 6%，原意想要讓 Encoder Input state 參考 Decoder Sequence。（ 實驗於 Concat 後過 Dense 再餵進 Decoder、與只參考 Encoder Final state。但效果不如直接 Concat 後餵給 GRU Layer.

註三：Encoder Input 使用 One hot Encoding。曾實驗過 Embedding 與過 FastText word2vec，Performance 皆下降。推測現 Model 之 Input 降維效果比 Embedding Layer 更好 (註一)

註四：此 Layer，原意希望由 Decoder Full Squences / Final State，與 Decoder Input 計算 Weight 權重。實驗於此 Layer 之 Dense Layer 餵進 Encoder Input，再進 Deocder Layer，但 Performance 下降，推測此資訊 GRU Layer 處理能力大於 NN。此 SoftMax Layer 推測是由 Encoder Layer 之 Full Sequence (Length 246) 計算出 Encoder 之各 Input (Length 14) 之權重。

實驗與討論

1) 模型訓練過程

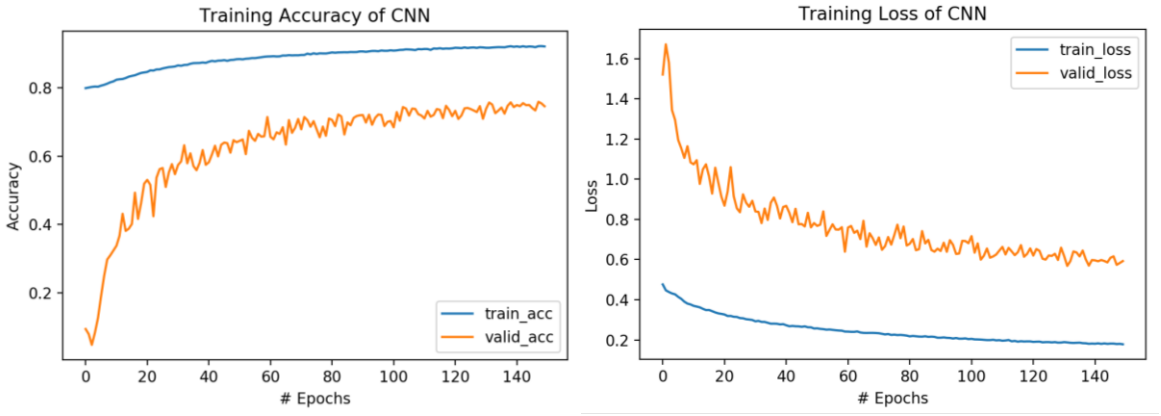

Figure 4. Training process

2) 模型架構比較

Retrieval model 中，RNN 疊兩層會有比較好的準確度，但是疊到第三層就沒有更多的進步，並且訓練時間更久。

3) 正負樣本比例的影響

由於測試資料是四選一的選擇題，大多數的答案都是錯的，因此我們認為增加錯誤配對的樣本數兩可以提高模型的準確率。以下是我們針對「正負樣本的比例」在 RNN 檢索模型上進行的實驗結果：

正負樣本的比例	Kaggle categorization accuracy (public)
1:1	0.36000
1:2	0.42900
1:3	0.50300
1:4	0.57999
1:5	**0.72200**
1:6	0.70800

Table 3. Positive/Negative ratio experiment

從上表可以看出，當比例來到 1：5 的時候，我們的模型能夠有最好的表現。即使再增加負樣本的比例，也無法能夠有更多的準確度提升。

4) 訓練速度

如前所述，採用 CNN 取代部分 RNN 的好處在於，CNN 的模型較大部分現行的 RNN 模型而言運算比較單純、更新參數的速度也相對較快。我們觀察到 CNN 模型雖然在處理有時間相依性的資料上，表現略差於 RNN 的模型，但大幅減少了訓練時間。

另外，我們也分別比較了相同架構下使用 LSTM 和 GRU 實作 RNN 的差異。一般認為 GRU 的參數量雖然較 LSTM 少，在大部分的問題上可以達到差不多的表現。下表為分別使用 CNN、GRU 和 LSTM 處理 MFCC 特徵的實驗結果。

	CNN	GRU	LSTM
每個 epoch 耗時*	~30 s	490 s	> 30 mins
單一模型 val_acc		0.8802	0.8913

* GPU = GTX 1080 Ti / batch size = 512

Table 4. Training speed comparison

5) ensemble

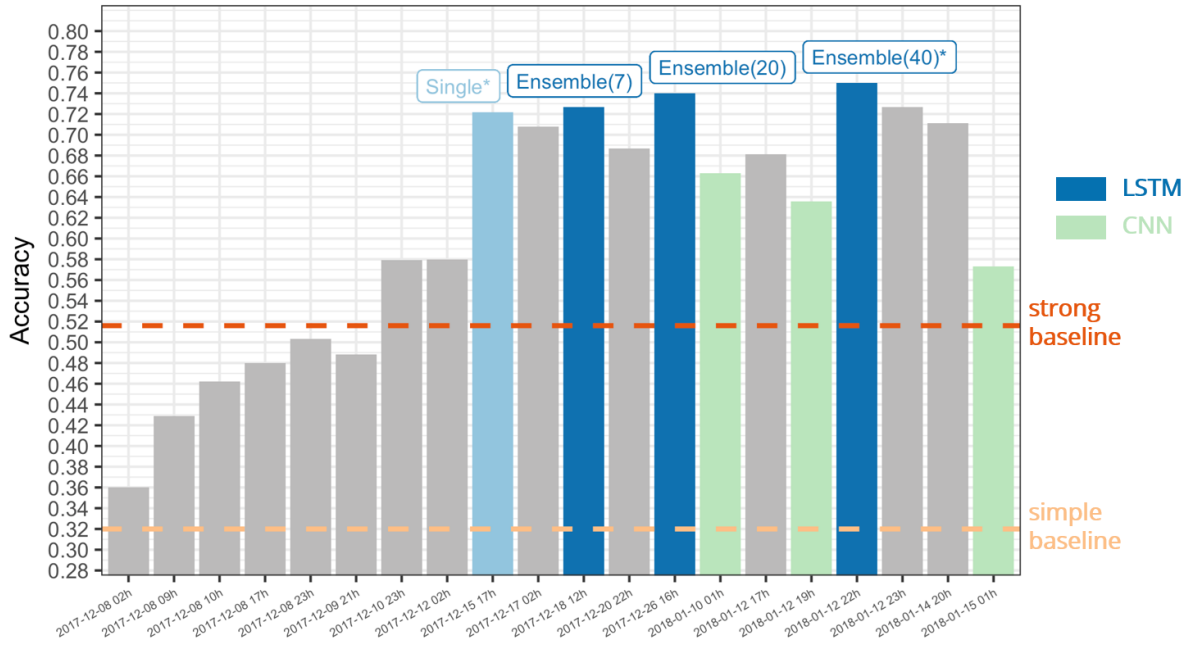

Figure 5. Performance evolution until 1/19

藉由不斷的隨機抽樣，我們可以創造出許多基於不同樣本訓練而成的模型，在將他們預測出來的機率相加，取 argmax 所得的那個 class (選項) 當做我們最終的 output。上圖展示我們於 final presentation 前的模型準確度。可以發現 ensembling 可以有效的提高準確度。最後於 final presentation 前，我們使用 40 個模型 ensemble 起來，達到 0.75900 的準確率。

經過以上實驗我們得出的最佳模型訓練方式，我們發現能夠大幅提升準確度的操作是增加負樣本的比例，並且是度的加深類神經網絡。而使用 CNN 雖然準確度不及 RNN，但是在訓練時間上有很大的優勢。

1) Data augmentation:
   在得到 mfcc 資料後先將 frame 個數做降取樣，取原本長度的一半，並且保留原始樣本。這樣一來可以得到多一倍的資料量。
2) Negative sampling:
   透過上述實驗我們發現 1:5 的正負樣本比例是個不錯的選擇，因此這邊我們隨機挑選 5 個不同的 index 做為 rolling shift 的開始以產生多樣化的負樣本。相對於原始資料量，擴充後的正負比例為 2:10。
3) Using GRU:
   由於 LSTM 需要花費較多的時間，因此我們改用 GRU 做為 RNN layer。

4) Ensembling:

由於是隨機取樣，因此每個模型訓練時所看到的負樣本組合都不盡相同。藉由 ensemble 20 個模型，我們讓準確度達到 0.858。