2017 Machine Learning Final Project

r06922097 鄭雅文/ r06922128 楊碩碉/ r06944025 陳品君

- **Team name, members and your work division (1)**

NTU_r06922097_各位同學大家好咳咳

R06922097 鄭雅文：研究 LSTM model，整理、繳交作業

R06944025 陳品君：討論 model，繪製 model 圖

R06922128 楊碩碉：train model，improve model

- **Preprocessing/Feature Engineering (3)**

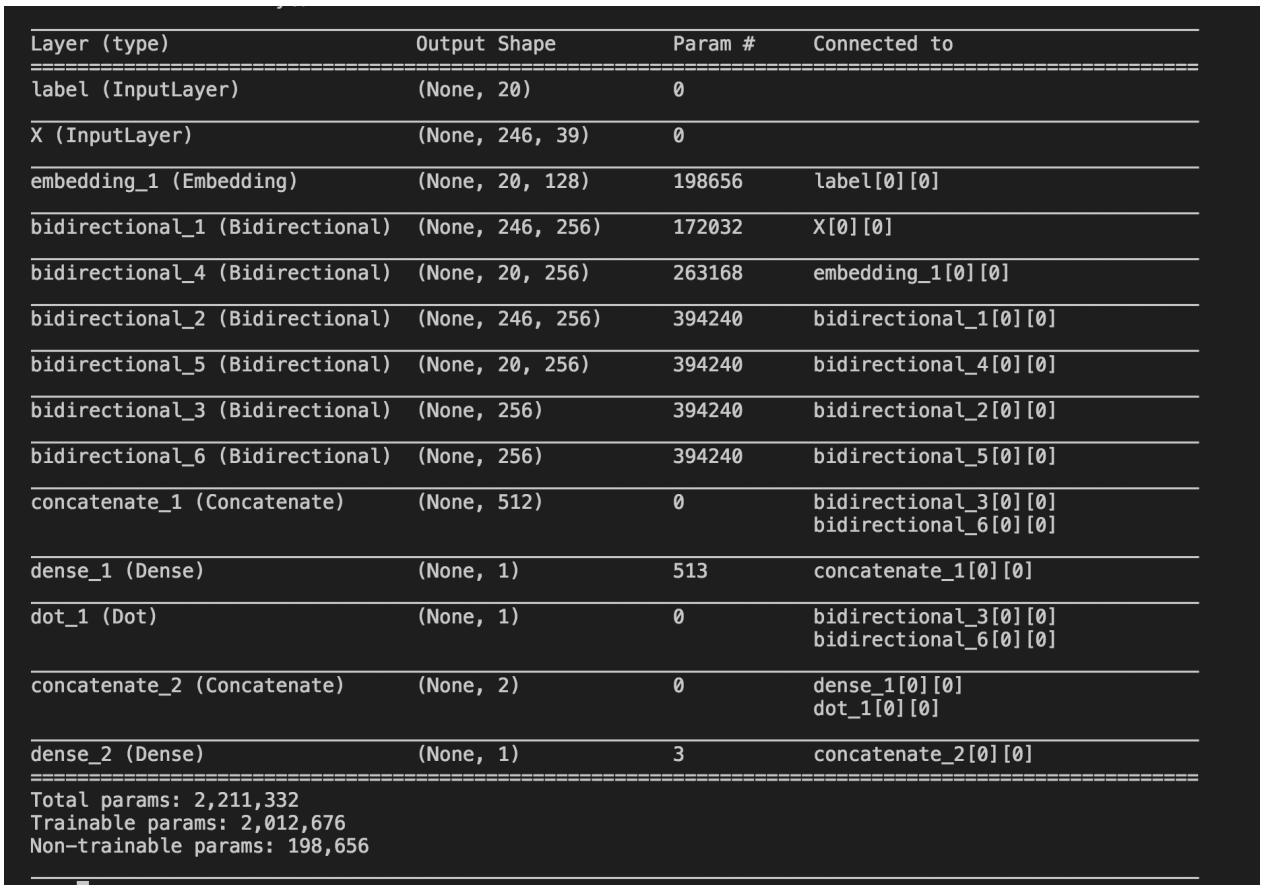
1. 用 gensim 的 Word2Vec 做 word embedding，沒有用 jieba 分詞，因為怕 testing 的斷詞結果沒有在 training data 中出現。
2. 文字部份：
   對 training data 建字典，並再加入<bos> <eos> <pad>以及<unk>來處理 oov 的問題，embedding 部份跟 model 一起 train。
3. 聲音部份：加入額外資料
   對 mfcc 做 downsampling (只取偶數 time step 的 feature)
   加 noise (Normal Distribution, stddev=0.05)
   Negative Sampling (Retrieval Model Only):
   對每一筆 training data，都取 5 筆錯誤的 caption 當作 label=0 的輸入，不考慮字數，每一個 epoch 都重新選取。
4. 增加 test 資料的方式用相同字數的 caption 當作一起 train 的資料，如果音訊檔跟文字是符合的，output 的 label 就是 1，如果不符合，則 output 的 label 是 0

- **Model Description (At least two different models)　(7)**

1. 用 LSTM 做 sequence to sequence，input 為經 padding 過後的音訊檔，output 為 embedding 過後的文字檔，model 架構如下：

```
>>> model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_10 (LSTM)               (None, 128)               86016
_________________________________________________________________
repeat_vector_6 (RepeatVecto (None, 20, 128)           0
_________________________________________________________________
lstm_11 (LSTM)               (None, 20, 128)           131584
_________________________________________________________________
time_distributed_5 (TimeDist (None, 20, 50)            6450
_________________________________________________________________
activation_5 (Activation)    (None, 20, 50)            0
=================================================================
Total params: 224,050
Trainable params: 224,050
Non-trainable params: 0
_________________________________________________________________
```

2. 用 LSTM 做 sequence to sequence，參考「挖洞給我跳」組的同學的 model，一個 input 為 padding 後的音訊檔，一個 input 為 embedding 後的文字檔，兩者做 dot 跟 concatenate，再把 dot 跟 concatenate 的結果再 concatenate 在一起，output 為 label，其中 label 為 1 是正確，0 是錯誤，因為會先製作測試 data，用同一音訊檔，把相同字數的 caption 找出來，如果 caption 對應到原本的音訊的 label 為 1，若不是則為 0。Model 架構如下：



Layer (type)	Output Shape	Param #	Connected to
label (InputLayer)	(None, 20)	0	
X (InputLayer)	(None, 246, 39)	0	
embedding_1 (Embedding)	(None, 20, 128)	198656	label[0][0]
bidirectional_1 (Bidirectional)	(None, 246, 256)	172032	X[0][0]
bidirectional_4 (Bidirectional)	(None, 20, 256)	263168	embedding_1[0][0]
bidirectional_2 (Bidirectional)	(None, 246, 256)	394240	bidirectional_1[0][0]
bidirectional_5 (Bidirectional)	(None, 20, 256)	394240	bidirectional_4[0][0]
bidirectional_3 (Bidirectional)	(None, 256)	394240	bidirectional_2[0][0]
bidirectional_6 (Bidirectional)	(None, 256)	394240	bidirectional_5[0][0]
concatenate_1 (Concatenate)	(None, 512)	0	bidirectional_3[0][0] bidirectional_6[0][0]
dense_1 (Dense)	(None, 1)	513	concatenate_1[0][0]
dot_1 (Dot)	(None, 1)	0	bidirectional_3[0][0] bidirectional_6[0][0]
concatenate_2 (Concatenate)	(None, 2)	0	dense_1[0][0] dot_1[0][0]
dense_2 (Dense)	(None, 1)	3	concatenate_2[0][0]

Total params: 2,211,332
Trainable params: 2,012,676
Non-trainable params: 198,656

3. Simple Baseline Model:

   轉化為 classification 問題，輸入為 mfcc，輸出為 caption 的「第一個字」。

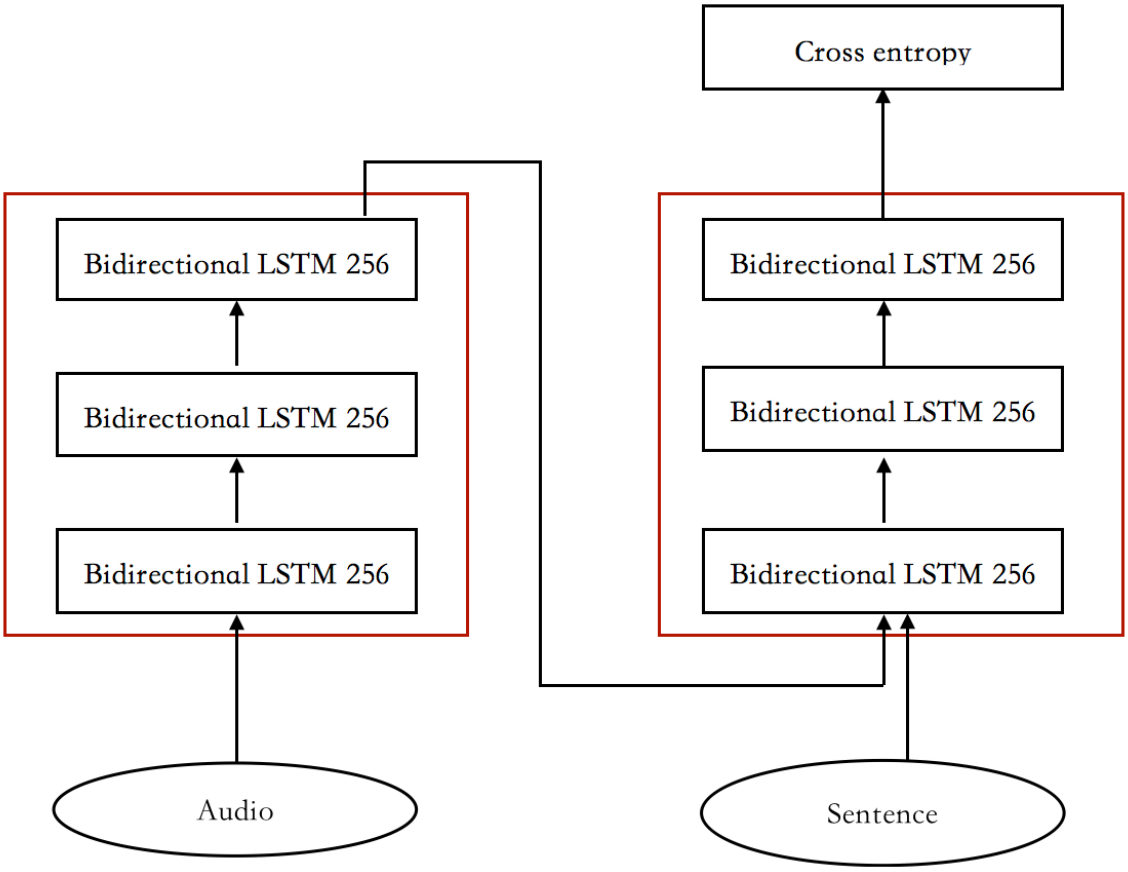
   先把 mfcc 通過(多層)rnn，再通過一層 fc 得到每個字的 probability，取 softmax 後得到輸出。

4. Strong Baseline Model:

   採用 Sequence-to-Sequence Model，輸入為 mfcc，輸出為 caption 的「sequence」。

   Encoder：mfcc 通過(多層)bidirectional rnn，並且把每一個 time step 的 output 記下之後做 attention。

   Decoder：一樣用多層的 rnn，把 encoder 的 final state 當作 encoder 的 initial state，並且對 encoder 做 attention，attention mechanism 為 LuongAttention。



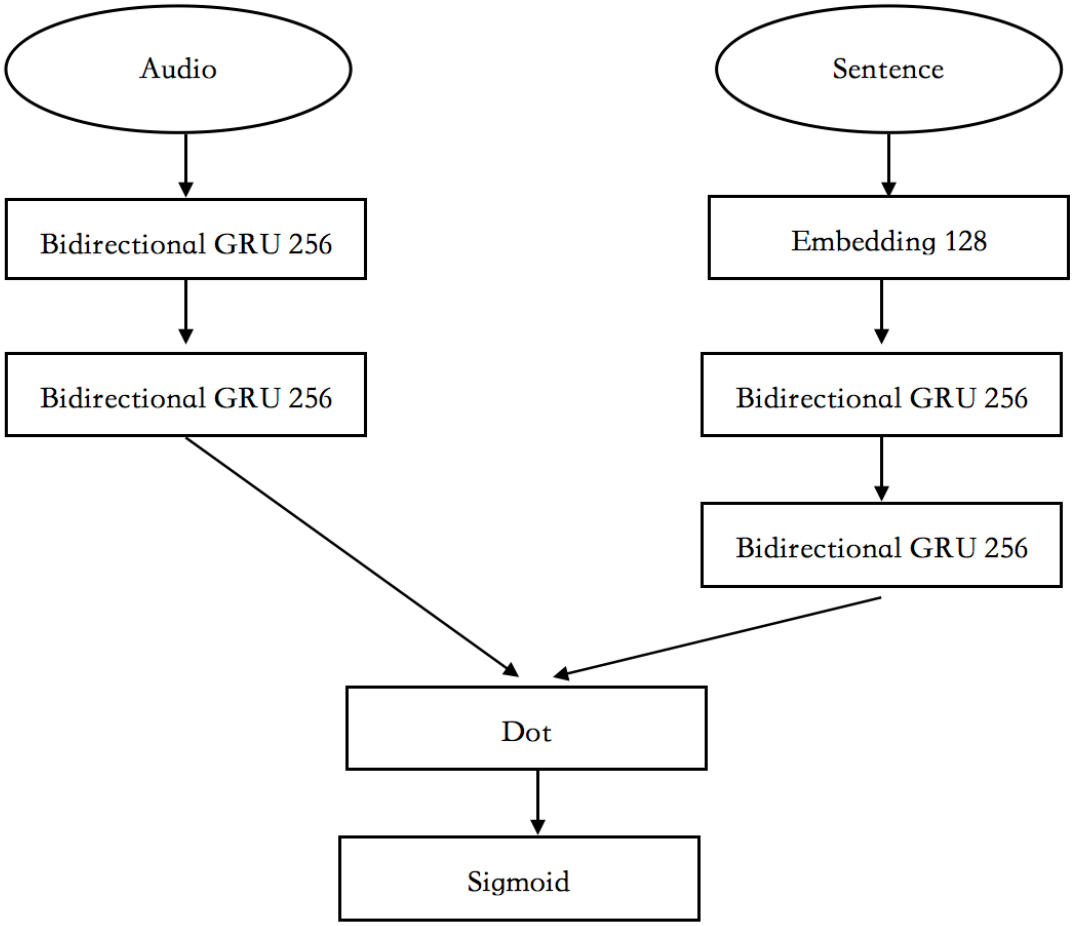
5. Best Model:

   採用 Retrieval Model，輸入為 mfcc 跟 caption，輸出為 label (0, 1)。

   Encoder (mfcc): mfcc 通過多層的 bidirectional GRU 得到一個 vector

   Encoder (caption): caption 先經過 embedding layer 得到 word embedding，在通過多層的 bidirectional GRU 得到一個 vector

   Similarity: 把前 2 個 vector 做內積，再取 sigmoid 做為 label



- **Experiments and Discussion (8)**
  1. Kaggle Accuracy: 0.246
     比全部猜第三個的結果還差，猜測因為除了模型太簡單之外，embedding 的文字有太多 0 的資料，所以可以估計的東西很少，很沒有參考性。
  2. 尚未上傳 Kaggle，但 validation 的資料準確率大概 0.4。
  3. Simple Baseline Model:
     Kaggle Accuracy: 0.328
     只預測第一個字就能過 simple baseline，出乎意料，但是 model capacity 也就只有這樣，再怎麼調都不能更高。
  4. Strong Baseline Model:
     Kaggle Accuracy: 0.724
     把 mfcc 輸入後對每一個選項算 loss，取最低的當作答案。試過 2 層 3 層的 bidirectional lstm，維度從 64 到 512，最高大概 0.72 附近。
     有個有趣的現象是，當 validation loss 到最低點時，正確率不會是最高

的，要在多 train 幾個 epoch 才會有最高的正確率。

另外在 training 的時候也有加入 dropout，但是很難調到一個最佳解，因為即使把 dropout rate 調高，validation loss 下降，accuracy 也不會再增加。

5. Best Model:

   Kaggle Accuracy: 0.84 (single) 0.864 (ensemble)

   把 mfcc 跟 4 個選項輸入到網路後算出 4 個相似度，取高的當作答案。

   Retrieval Model 是在聽完 final 發表後才開始寫的，一開始沒想過可以表現的這麼好，屌打之前做的 sequence-to-sequence。