Machine Learning Final Project
Listen and Translate

r06942079 葉達駿
r06945003 林鈺盛
r06921062 蘇楷鈞
r05922058 陳語宸

Division of Work

1. 葉達駿 - 建立 retrieval 模型，並比較不同參數對準確率之影響
2. 林鈺盛 - 協助模型參數調整與驗證
3. 蘇楷鈞 - 建立其他模型與 attention weight 之觀察
4. 陳語宸 - 協助模型訓練

Data

1. Format
   train.data，test.data：利用 Kaldi 方法從音頻訊號中所抽取的 mfcc feature 序列，由 39 個維度所描述。
   train.caption，test.csv：音頻訊號所對應之中文翻譯結果及選項。

2. Preprocessing
   為了避免音頻訊號中每個欄位值的比例（scale)不一而對預測結果造成影響，我們分別對 39 個維度的 mfcc feature 做標準化的動作，使得每個欄位之平均為 0，標準差為 1。而在中文字串的處理上面，我們並無特別花心思的去作斷詞的處理，而是把每個中文字視為一個獨立的個體，統計共有 2389 個不同的中文字，也是我們在之後在做實驗時所使用的字典（vocabulary)大小。

Model 1: Retrieval Model

Methodology

此方法主要是將此問題視為一個 2 分類問題。整體架構上，我們會在輸入端 input 一組音頻訊號及一組中文翻譯文字序列，經過遞歸神經網路（recurrent neural network）後得到音頻訊號的向量表示法$\vec{a}$及翻譯文字的表示法$\vec{t}$。模型則會輸出這對 sample（音頻訊號，翻譯文字）之間的相關程度。為了在四個選項中選出最相關的翻譯，我們會分別計算四種選項的相關程度，並取分數最高的選項當作我們的解答。

1. Train word vectors：
   一開始我們是使用 facebook 的 pretrained word vector，但後來發覺 300 個維度有點太多了，效果不是太好，最後決定改用 gensim 的 Word2vec 來 train word vector，每個中文字的 dim 為 200，其他相關 training 的參數如下：
   min_count = 1， window = 3，iter = 5

2. Negative sample generation：
   為了提升詞向量的訓練品質，在訓練模型時我們會隨機產生負樣本並加入我們現有的資料中。其用意在於我們在最大化正樣本的機率時，同時要最小化負樣本的機率。換句話說，我們必須告訴模型哪些選項的翻譯文字對一個輸入音頻序號而言是相關的，而哪些是不相關的。從幾何學上來看，音頻向量 $\vec{a}$ 和它的翻譯向量 $\vec{t}$ 在向量空間上要是分常接近的，而和不相關的翻譯向量距離要是比較遠的。

   在這個問題中，一個正樣本定義為一個（音頻訊號，及其對應之翻譯文字）的組合。一個負樣本定義為一個（音頻訊號，及和其不相關之翻譯文字）的組合。我們認為如何選取負樣本將是整個模型的關鍵。目前我們的方法是隨機選取一段翻譯文字，它必須和正樣本的翻譯文字沒有任何的交集，當作負樣本。

3. 模型訓練：
   模型的基本架構主要是基於 [1]，並針對細節做些微改良，參見下圖一。利用 1D 的 convolutional neural network(CNN) 和 LSTM cell 的雙向 recurrent neural network (RNN)做為音頻訊號的 encoder；使用 LSTM cell 的雙向 RNN 做為翻譯文字的 encoder。經過 encoder 後的音頻向量與文字向量會各自經過 self attention [2]計算出每個 timestep 的權重，取其加權總和得到濃縮過的音頻向量 $\vec{a}$ 以及文字向量 $\vec{t}$。最後，利用 $\vec{a}$ 和 $\vec{t}$，我們可以經由以下公式計算出這對音頻、文字是否相關(相關：1，不相關：0)：

   $$\text{p(flag} = 1|a, t, M) = \sigma(a^T M\, t + b)\text{，}$$

   其中，M 是一個用來做線性轉換的矩陣，而 b 則是 bias 項。

   訓練參數：
   - cnn: kernel size = 3, filters = 200
   - rnn: LSTM cell, hidden unit = 200
   - loss: binary crossentropy
   - optimizer: rmsprop
   - epoch: 300
   - training data: 95% (randomly chosen)
   - validation data: 5% (randomly chosen)
   - The ratio of positive labels to negative labels is 1:1

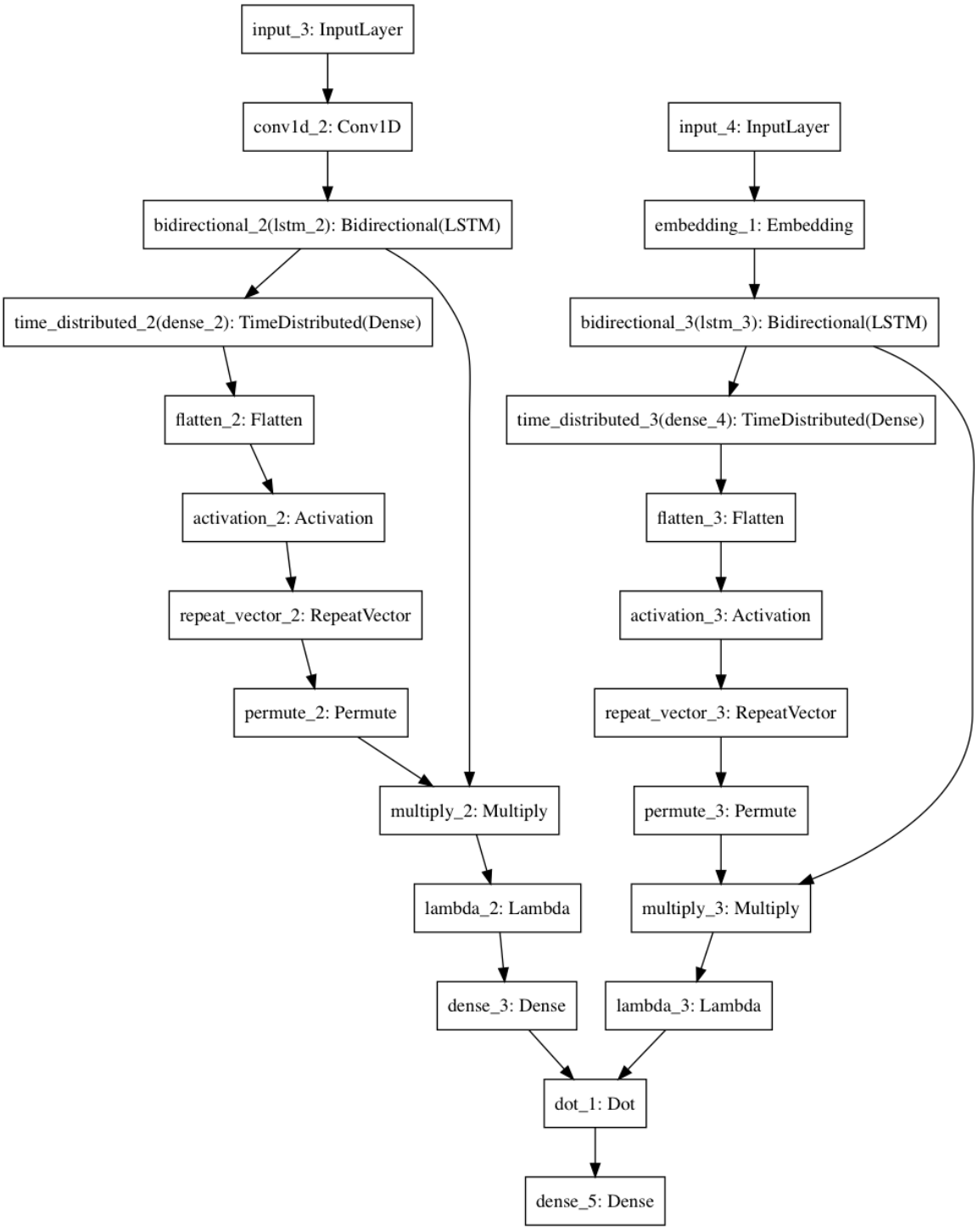

Figure 1: Diagram of our model.

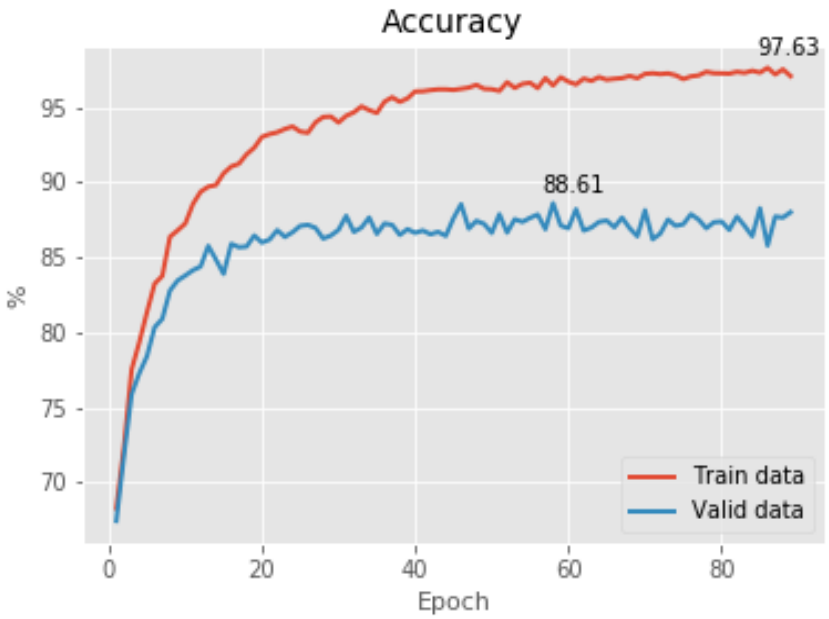

Figure 2: Training process of our model with early stopping.

Experiments

以下比較不同的模型、參數對預測準確率的影響：

1. 比較是否在 text input 端加上 1D cnn 的影響：

	加上 cnn	不加上 cnn (本模型)
kaggle public score	0.83400	0.87000

2. 比較不同的 dropout rate 對準確率造成的影響：

Dropout	0.3	0.4 (本模型)
kaggle public score	0.86200	0.87000

3. 比較不同的 epoch 數對準確率造成的影響：

epoch	100	300 (本模型)
kaggle public score	0.85700	0.87000

4. 比較是否在每個 epoch 前重新取 negative sample 的影響：

	不重新取	重新取 (本模型)
kaggle public score	0.67700	0.87000

Discussions

我們認為本題最關鍵的兩個部分便是加入 attention 機制以及針對負樣本如何作選取，另外經過實驗得知在訓練的過程中不斷更新負樣本，能夠讓機器看到更多錯誤的配對，對於本題使用 retrieval model 架構下的情況有非常顯著的幫助。

Attention visualization

Attention mechanism 是一種有效去抓住資料該注意的位置的機制，以翻譯而言，則是抓出每一次翻譯時須注意的被翻譯的文字，而用 self attention 則是把該強調的資料給增大權重，使得模型能抓住重點。

在這次台語翻譯的任務，先看音訊檔 attention 的 weight，猜測模型可抓出有聲音的位置，如 Fig 3、Fig 5，Fig 3 對應的翻譯如 Fig 4、Fig 5 對應的翻譯如 Fig 6，再觀察模型對各位置的權重時，有發現答案個字的權重通常在一些語助詞時會較低，可能是在 retrieval model 中，主要的任務並非把每個字都翻譯出來，因此可能語助詞與句子的解讀並無太大關聯，如 Fig 7。

Fig 3：第 120 個音訊檔

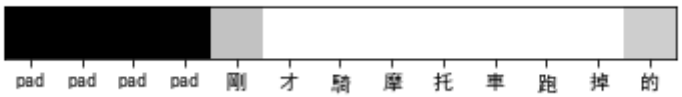

Fig 4：第 120 個音訊檔之翻譯

Fig 5：第 5648 個音訊檔

Fig 6：第 5648 個音訊檔之翻譯

Fig 7：第 50 個音訊檔之翻譯

Model 2:

Methodology

架構為 seq2seq 以 retrieval model 做文字(每個 time stamp)相似比較並加入 attention mechanism[2]，主要想法為每一段從音訊翻譯出來的字 $\vec{a_i}$ 和答案的字 $\vec{t_i}$ 越相近越好(i 為第 i 個 time stamp)，因此輸入音訊後會翻譯出 13 個字(包含 padding)進而和輸入的答案依每個 time stamp 做內積取得其相似程度，再以以答案所產生的 attention weight 做加權平均當作最後相似程度的分數。架構流程如 Fig 8。

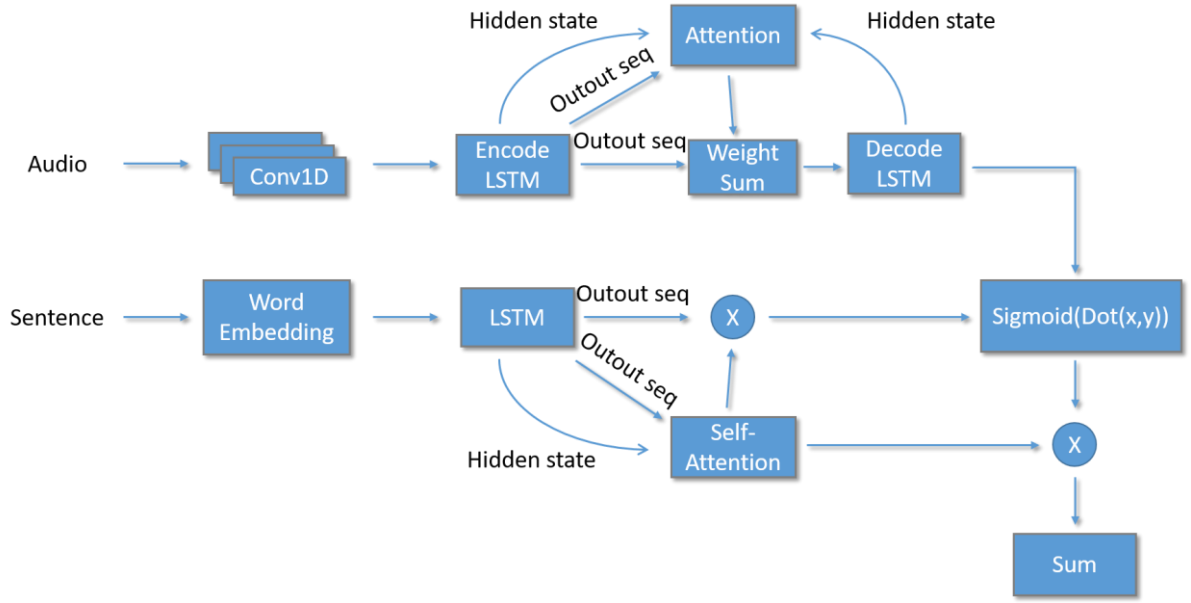

Fig 8：Structure of model 2

模型說明：

音訊(Audio)經過 Conv1D(kernel size = 3, filters = 192)後，送入 Encode RNN(units = 256)後，先對 output sequence 做一次 self attention 後，在每次 Decode RNN(units = 384)時，皆以產出的 hidden state 對所有 output sequence 做 attention weight sum 再給下一次 Decode RNN 做輸入得x_i。

答案句子(Sentence)經過事先訓練好的 word2vec 做的 Embedding layer 產出 200 維度的文字向量，word2vec 參數如 model 1 的設定，經過 RNN(units = 384)後，以 hidden state 做 self attention，產出每個文字的向量y_i。經由$Sigmoid(\ Dot(\ x_i, y_i\ ))$得$x_i$、$y_i$相似程度，原本以算術平均取得最後結果，但由於效果不是很好，更改為以 attention weight 做加權平均後效果較佳。

Experiments

訓練參數：

Loss function	Optimizer	Epoch	dropout	recurrent dropout
binary crossentropy	rmsprop	131	0.4	0.3

訓練方法：

在每個 epoch 會重新挑選 Negative sample，且事先劃分的 Train 和 Valid data 會在每三輪 epoch 後交換 0.5%的音訊與答案，以讓模型可以多看到一些音訊，但不至於 overfitting。Fig 9 為訓練過程。

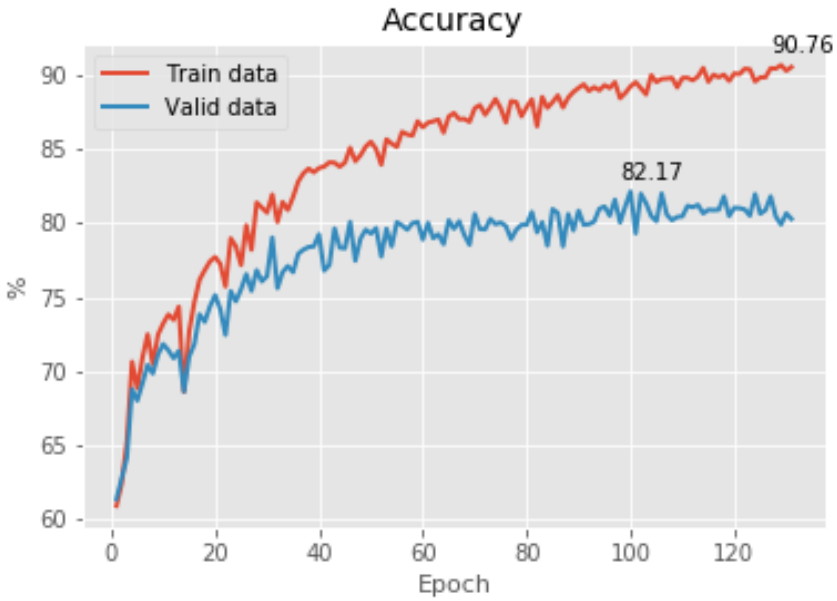

Fig 9：Training process of model 2

如果最後以算術平均為輸出，結果為 0.5679，而改用加權平均則提升為 0.6050，猜測是因為 padding 的部分在短句子時影響較大，因此，用 attention weight 則可以使 padding 部分的權重降低，進而專注在有文字的部分。

Sum Method	算術平均	加權平均
kaggle score	0.5679	0.6050

除了原本架構以外，嘗試了以答案的每個 time stamp 對音訊做 attention，以 attention weight 做最後加權平均，就結果而言，以答案做 attention 可得 0.6600。

Attention Method	hidden state	答案
kaggle score	0.6050	0.6600

Discussions

比較我們最好的模型(Model 1)來說，這模型複雜許多，加入了 seq2seq 的概念並無法使得準確率提升，或許在台語的翻譯裡，語意的了解會比字對字翻譯來的重要。另外，在無法用 dynamic RNN 的狀況下，用 attention 的機制來減少 padding 的權重，可降低影響結果的問題。

References

1. LOWE, Ryan, et al. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. arXiv preprint arXiv:1506.08909, 2015.
2. "Attention Is All You Need", Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, arxiv, 2017.