ML2017FALL-Final Project Report

一、題目

conversations in TV shows

二、Team name, members and your work division

Team name: NTU_b03801039_MLfinal

Members: 楊福恩 B03801039
劉翔瑜 R06942095
陳貞霓 R06548051
張漢維 B01901181

Work division:

楊福恩: 想法討論，模型設計，模型實作與測試，報告撰寫

劉翔瑜: 想法討論，模型設計，模型實作與測試，報告撰寫

陳貞霓: 想法討論，模型設計，模型實作

張漢維: 完全沒做事，也沒有參與討論

三、Preprocessing/Feature Engineering

首先將 5 個劇本合成一個完整的檔案，因為這個題目的 training data 以及 testing data 皆為繁體中文，所以預處理的部分主要是利用 jieba-tw 來做分詞，將每個句子中的中文詞語做適當的分割，再將做完分詞的句子存入一個新檔案 trainSeq.txt 中，最後以 gensim 的 word2vec.Text8Corpus 將 trainSeq.txt 重新讀入，以進行接下來的模型訓練。

特別一提的，分詞所使用的 jieba 斷詞台灣繁體特化版本，來自於 https://github.com/APCLab/jieba-tw.git，是採用和原始 jieba 相同的演算法，替換其詞庫及 HMM 機率表所製做出針對台灣繁體的 jieba 斷詞器，較適合用於繁體中文的分詞。

四、Model Description (At least two different models)

(1)模型 1: word embedding model

word embedding 利用 gensim 的 word2vec.Word2Vec 來實作，方式為直接使用如下的程式碼:

```
models = word2vec.Word2Vec(sentence, size=word_dim, min_count=5, workers=4, iter=25)
```

其中有手動設定修改的參數如下:

- sentence: 這個參數代表的是所要輸入模型進行訓練的文本資料。
- size:最終設定值為 300。這個參數的意義為經由這個 word2vec 的模型訓練出來的詞向量的維度。
- min_count: 最終設定值為 5。此參數是代表若在訓練的文本中，某個

詞出現的次數小於 min_count，那麼這個詞就不會被視為訓練的對象。

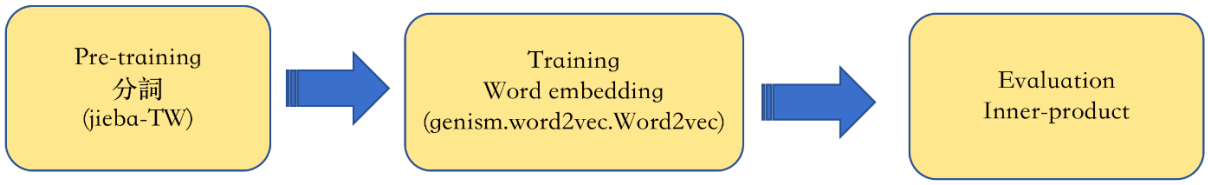
- workers: 設定值為 4，代表執行緒的數目。
- iter: 設定為 25，代表訓練迭代的次數，原本參數預設值為 5

而所使用的 word2vec 演算法為 CBOW 模型(Continuous Bag-Of-Words Model)。

利用全部的 training data 將 word embedding model 訓練完成後，把 testing data 中的題目與每個選項各別句子中每個詞所對應的詞向量相加起來，若 testing data 中有語詞是 training data 中沒有出現的，就將其對應的詞向量設為零向量，最後再把題目句子對應的詞向量相加結果與每個選項對應的詞向量相加結果作內積比較相似度，內積結果最大的那一個選項，就選為答案。

最佳結果:

Kaggle public score: 47.826%



(2)模型 2: Seq2seq model

將問題視為一個類似聊天機器人的系統，意即將 training data 的劇本中的句子視為上下句都具有對話關係來做訓練，在 testing 時便是以輸入題目的句子，讓 model 產生出最有可能是下一句的句子，再用這個產生出來的句子，和各個選項比較 word embedding 後的詞向量相似度，將相似度最高的視為答案。

訓練方式為輸入劇本中的某一句，希望經過模型所產生的句子要盡量接近劇本中的下一句，

模型利用一層的 LSTM 作為 Seq2seq 部分的主要 network，

activation function 為 tanh

loss function 為 mean square error

optimizer 為 adam

epochs 設為 30

而將句子變成詞向量的 word2vec model 和模型 1 相同，將產生出的詞向量和每個選項對應的詞向量作內積，內積結果最大的選項即為答案。

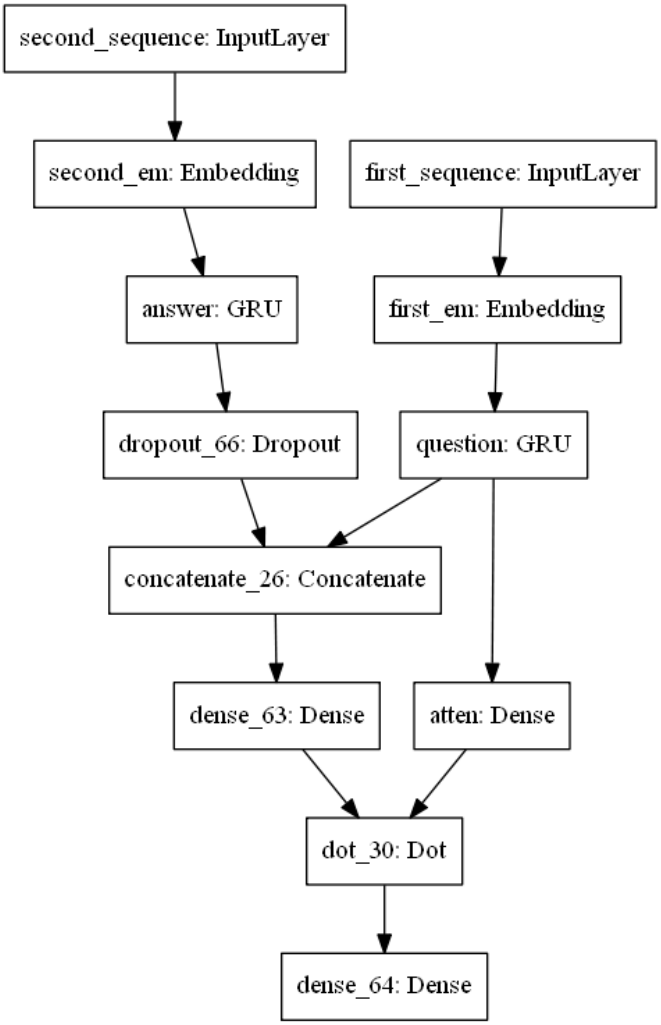
最佳結果:

Kaggle public score: 18.814%

從 Kaggle 分數可看出，這個模型的嘗試並沒有成功，發現較大的問題在於原本訓練的 data 中，劇本的上下句並不全然是對話的關係，利用這樣的方式較難訓練出好的對話系統，且可能生成的句子和選項的句子有很大的差異，造成比較相似度時無法準確判斷出結果，導致準確率不高。

(3)模型 3: GRU(RNN) model

這個 model 的目標為，觀測兩 seq 的關係以計算出此兩 seq 互為上下句的機率



首先使用 Word2Vec(data,size=64, window=10, min_count=5, workers=4,sg=1)訓練出的 word2vec 為 embedding matrix，接入 GRU 後進行內積，最後為 Dense(1,activation='sigmoid')，由於是要辨別出上下兩句的關係，training data 產生的方式為:

1. first_sequence 為 data 中所有的 seq，second_seqence 為

first_sequence 的下一句，此時為上下句關係，label=1

2. first_sequence 為 data 中所有的 seq，second_seqence 為隨機取出的 seq，前後兩句不相關，label=0

最後將使用訓練出的 model 對題目及每個選項作預測，輸出最大的即為答案，kaggle 上的分數約為 46%左右。

五、Experiments and Discussion

因為最終在 Kaggle 上結果以模型 1 最佳，故以下實驗與討論皆是以上述的模型 1 作為基礎所做的實驗與討論。

(1)調整不同的詞向量維度

以下實驗結果為 iter=5(預設值)下的結果

size	150	250	300	350	400
Kaggle public score	46.086%	45.889%	46.126%	45.889%	45.810%

可發現此模型在詞向量維度 150~400 間，準確率都在 46%左右，且此模型以 size= 300時的準確率為最好，若詞向量維度太小可能讓詞向量所蘊含的資訊太少，無法訓練出好的 word embedding，而詞向量太大可能會有冗餘的雜訊在其中，導致準確率下降。

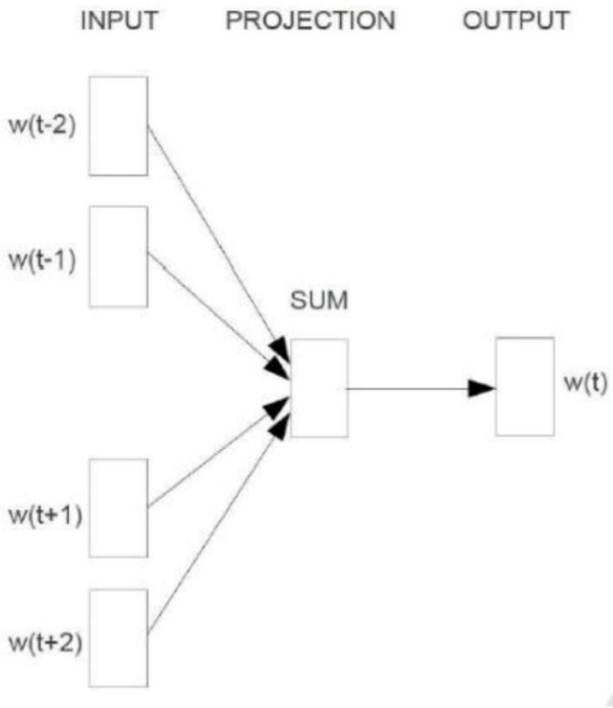
(2)採用不同 word2vec 模型

gensim 中的 word2vec.Word2vec 主要支援兩種常用的 word2vec model，分別是 CBOW(Continuous Bag-Of-Words) model 與 Skip-Gram model，以下將實驗兩種模型來比較準確率

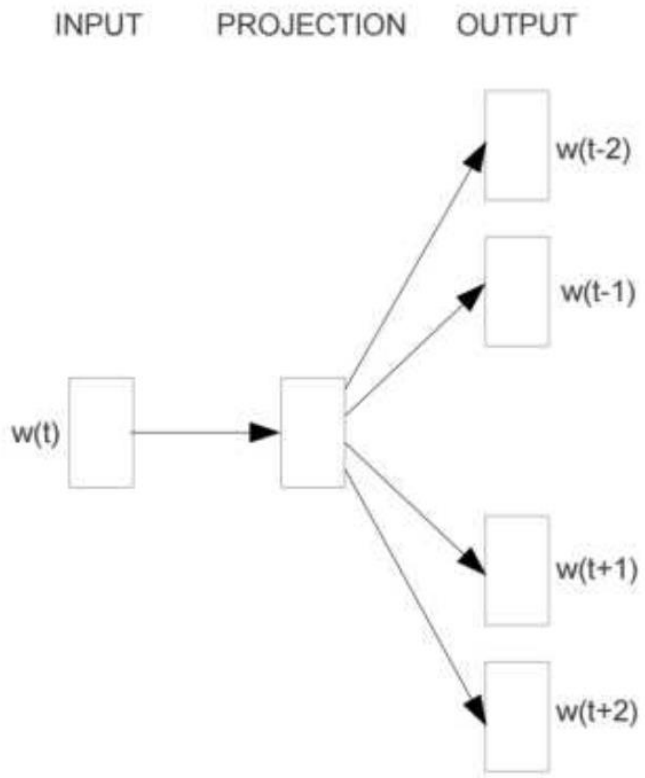
以下實驗在詞向量維度皆為 300 且 iter=5(預設值)條件下進行

(a)CBOW model，即 word2vec.Word2vec 的參數 sg=0，

Kaggle public score=46.126%



(b)Skip-Gram model，即 word2vec.Word2vec 的參數 sg=1，
Kaggle public score=41.857%



由實驗結果可發現在其他條件相同的情況下，使用 CBOW model 可得到較佳的準確率。

(3)嘗試不同的 window 大小

window 的意義為當前詞與預測詞的最大距離。

在 gensim 的 word2vec.Word2vec 中，預設值為 window=5

以下實驗結果為 iter=5(預設值)下的結果

window size	3	5	7
Kaggle public score	45.335%	46.126%	45.968%

由實驗結果可知，window size 在此架構下設定為 5 有較佳的結果，window size 過大或過小都會讓結果變差。

(4)有無使用停用詞

在分詞過程中，有自行設計停用詞表來濾除一些英文、標點符號或是可能與語意較無關的詞，所設計的停用詞表如下:

「the of is and to in that we for an are by be as on with can if from which you it this then at have all not one has or that A B C D , ? 、 。 “ ” 《 》 ！ ， ： ； ?: ; \t 嘎 嘎登 吭鏘 我 你 妳 他 她 它 我們 你們 他們 的 了 耶 呢 嗎」

實驗結果:

(a)有使用停用詞: Kaggle public score=47.826%

(b)沒有使用停用詞: Kaggle public score=44.150%

可發現使用了停用詞後，結果變好，推測原因可能為這些停用詞本身並不帶有特定重要的文意，出現在句子中反而會造成整體句意被判斷錯誤，所以將其濾除有助於準確率的提升，讓結果變好。

(5)使用不同 jieba 版本

在本次專題實作中嘗試了兩種不同版本的 jieba

(a)原始的 jieba 版本，並且將字典替換成「dict.txt.big」，

在size = 300時，Kaggle public score=44.505%

(b)jieba 斷詞台灣繁體特化版本，來自於

https://github.com/APCLab/jieba-tw.git，

在size = 300時，Kaggle public score=47.826%

由此實驗結果可推測，因為原始的 jieba 是以簡體中文來設計，雖然替換了字典，但 HMM 機率表的部分依然為適用於簡體中文，所以對於繁體中文的分詞結果較差。而 jieba 斷詞台灣繁體特化版本則是採用和原始 jieba 相同的演算法，並且替換其詞庫及 HMM 機率表所製做出的針對台灣繁體的 jieba 斷詞器，因此更能有效地對於繁體中文進行分詞，而得到較佳的結果。

(6)使用不同比較相似度的方式

在最後比較題目與各個選項的相似度時，嘗試了兩種不同的方法。

方法	內積	Cosine similarity
Kaggle public score	47.826%	39.723%

可發現在此架構下，詞向量維度=300 時，使用內積來比較相似度的結果會優於使用 Cosine similarity。

(7)Iteration 不同

iteration 代表的是 word2vec model 訓練時迭代的次數，在 gensim 的 word2vec 中以參數 iter 來代表，在詞向量維度=300 的條件下

iter	3	5	10	15	20	25
Kaggle public score	44.268%	46.126%	47.351%	47.628%	47.747%	47.826%

以上為嘗試過的 iteration，由上方數據可發現 iter 愈大，準確率會提升，推測原因應為 iter 愈大代表訓練較多次，讓詞向量之間的關係可以被訓練得更好，所以會讓結果變好。