Machine Learning Final Report – TV conservation

----------組員名單----------

學號：R06921095　系級：電機所碩一　姓名：吳睿哲

學號：R05921116　系級：電機所碩二　姓名：林育琦

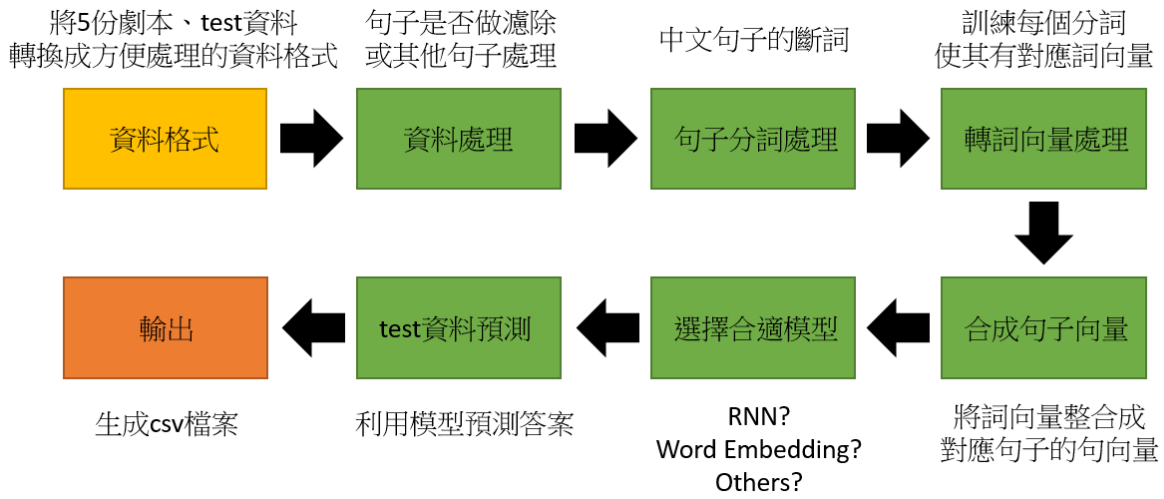
學號：R06921067　系級：電機所碩一　姓名：郭曜嘉

學號：R06921002　系級：電機所碩一　姓名：張哲誠

1. Introduction & Motivation

在這次的 final project 內，希望我們能獲得一段對話之後，從 6 個選項中選出下一句最可能的回應，為了能準確預估，如何將字轉換成向量，以及中文的分詞處理就變得格外重要，在完成這些重要的前處理再更進一步的尋找合適的 model 架構，並藉由目前 5 個連續劇的劇本作為訓練的來源。

Final project 的程序如下：



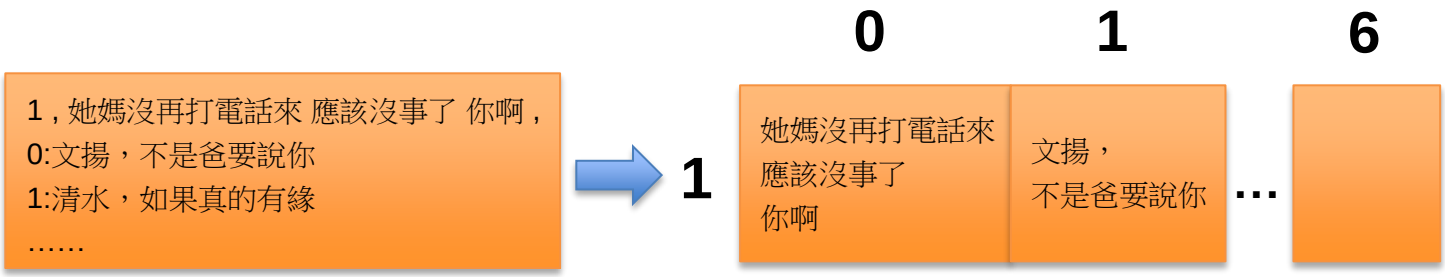
可以看出其每一個環節都有其重要性存在，為了使輸出答案能有更高的準確率，其模型和資料處理的考量都將在之後的章節敘述。

2. Data preprocessing / Feature Engineering

1. 資料格式

Training data：將原本一句台詞為一行的 5 份劇本，將其整理為 size 為 5 的陣列，其陣列內為一個 list 代表劇本內所有句子。

Testing data：部分則整理成(5060,7)的陣列格式儲存，0 為對話，1~6 為回應的選項。格式如下：



2. 資料處理

因為句子內有時穿插歌曲，而歌曲歌詞會以"歌詞"的形式在劇本裡出現，以及會有因應語言不同例如：英文 pencil 而台語聽成"便所"(廁所)這種劇情，處理部分只選擇將這些數字、英文、特殊符號利用正則表示式濾除。

3. 句子分詞處理

句子處理部分，使用 jieba 結巴分詞來做處理，jieba library 有三種模式，全模式、精準模式、搜尋模式，起初只使用全模式針對繁體語言做處理，
但發現以下情況：

句子：
你太抬舉我了 我哪有能力籌資呢 我還太年輕 根本沒有人脈
分詞結果：
['你', '太', '抬舉', '我', '了', '我', '哪', '有', '能力', '籌資', '呢', '我', '還太年', '輕', '根本', '沒', '有人', '脈']
可以發現分詞的結果並不佳，後來討論之後加入了 set_dictionary('dict.txt.big') 和 Opencc 轉成簡體。

想法來自[2]內提到，晚自習在繁體可能被分成"晚" "自習"，而在簡體則是被分成"晚自習"。

4. 轉詞向量處理

詞向量的訓練中，使用了 gensim 的 word2vec，起初依照如劇本內相同的將句子依序丟入訓練，但其訓練出來的詞向量，通常導致後續模型效果不佳，所以選擇將句子多句合成為一句之後再進行 word2vec 的訓練，將在下一頁以更明確的例子說明合成句子的作法。

	原本句子	新句子
1	關馬西在船上	關馬西在船上祈禱未來會一帆風順
2	祈禱未來會一帆風順	祈禱未來會一帆風順雅信也一樣衷心冀望
3	雅信也一樣衷心冀望	雅信也一樣衷心冀望多年來努力認真苦讀
4	多年來努力認真苦讀	多年來努力認真苦讀可以回家鄉服務
5	可以回家鄉服務	可以回家鄉服務其實雅信不知道
6	其實雅信不知道	其實雅信不知道在台灣
7	在台灣	在台灣....

以上例子為嘗試將兩句合為一句的例子，

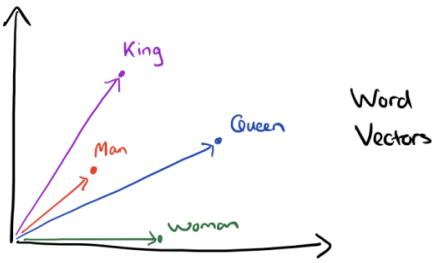
因此原本假設有 10000 句的劇本，可以藉由這樣 overlap 的方式生成 9999 句較長的句子，透過這樣方式似乎能讓 word2vec 訓練出較佳的詞向量反應詞與詞之間的關係。

也因為有了較長的句子，可以調大 word2vec 的 window 參數以及其他參數，將在後續討論其參數對於模型訓練結果的影響。

III. Model Description

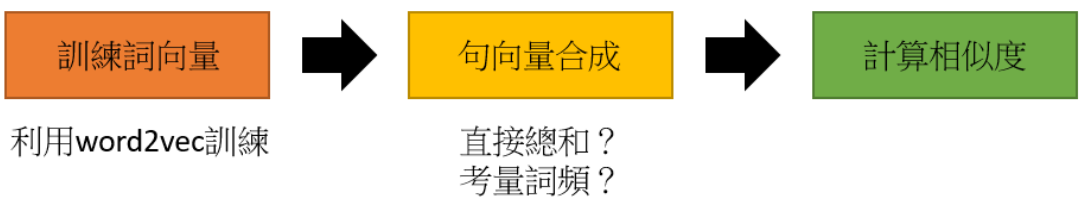
1. Word Embedding

透過 word2vec 的訓練後，可以得到對應的詞向量，word embedding 作法是將每一個句子都轉換成每一個分詞的向量總和，也就是說，假設每個分詞以 50 維做為向量，那此句子最後為句子內所有分詞的總向量，依然為 50 維，右圖來自 reference[2]。



此方式考量的是，對話與回應之間所用的詞彙，在詞向量上也許詞的向量接近，因此相加之後兩個向量應該所相差的角度應該會較小，使用餘弦相似度來計算每個答案的向量和題目的相似度，最高的則選為解答。

模型架構描述：



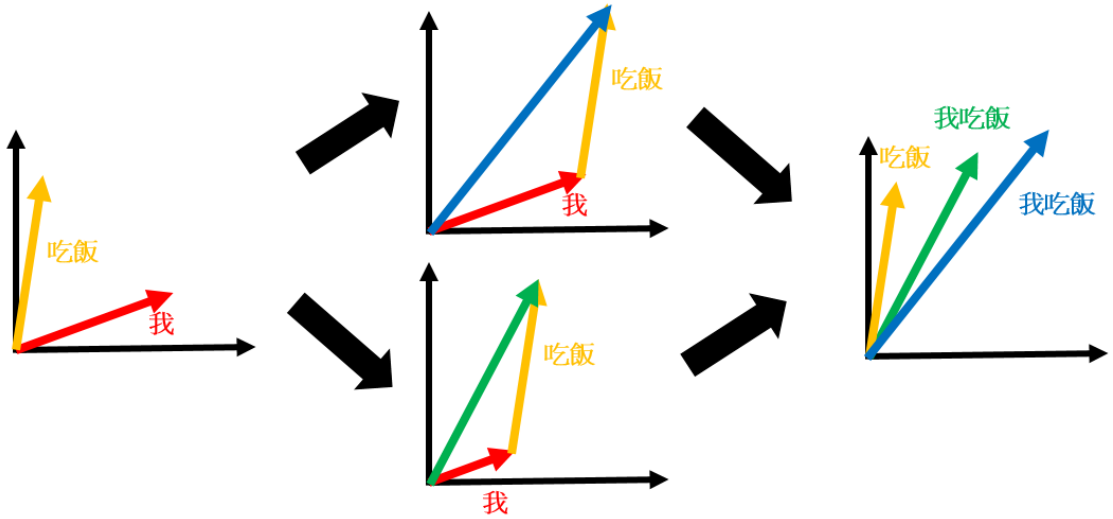
由模型架構上可以看出，其能影響模型準確率的因素主要有二：

一、是否訓練出有代表性的詞向量？

二、是否有適當的句向量合成機制，讓即使不同句子的組成，句向量合成仍能看出其相似度？

首先在訓練出詞向量的部分，於第二章有提到其相關方式，其訓練出來的詞向量影響準確率非常大，其影響結果探討將在下一章做討論，但在此模型架構上，詞向量的代表性占了非常重要的地位。

在句向量合成部分，也算是能影響相似度的一個關鍵，以下例子來說：

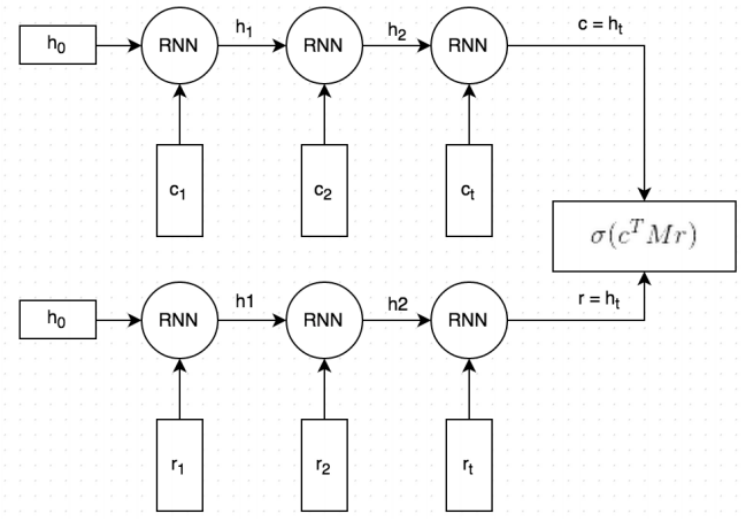


吃飯 v.s. 我吃飯

"我"這種字眼在句子裡有時只是因為句子結構但事實上不代表意義，如果能透過詞頻降低其 vector 影響度，換句話說影響其相加時的 weight，也許就能讓相似度更高一點。

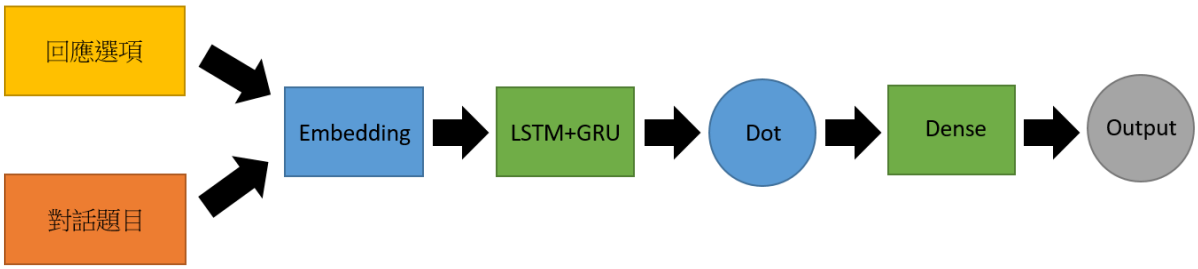
其想法來自 reference[6]，但只有引用其詞頻的想法，因不了解 SVD 而沒使用其 SVD 對句子處理的想法。

2. Dual Encoder



dual encoder[5][6][7](前頁圖片來自[7])是檢索式模型對話機器人的模型選擇之一，主要是透過將題目與回應都丟入同一個 RNN/LSTM 模型之後，兩者 Dot 後經過 Dense 的結果來推測是否有上下文的關係，透過有依序丟入字詞的 RNN 訓練方式，希望能做到和 Word embedding 只考慮字詞的向量總和但不考慮字詞順序的方式做區別。

模型架構描述：



首先必須將每一個句子轉成 index sequence，因考量在訓練劇本內句子分詞後最多有長達 14 個的 sequence 長度，因此選用以 15 為長度的 sequence 大小，透過句資料處理 train 出較好的詞向量 word2vec 字典，將每一句轉換成詞所代表的 index，再透過 embedding layer 轉換成其相對維度的向量，再進行 RNN 的 training，dot 兩個 sequence 進入 RNN 之後的結果，再經過 Dense 輸出結果。

為了產生可以 training 的資料，將五個劇本的每個句子(除了最後一個句子)作為題目，下一句做為答案，但訓練必須要有非正確的解答才能訓練，所以用一樣的題目，但給了 6 個並非下一句的句子作為回應，所以總共會有比原本題目多出 6 倍的資料。

此模型能夠訓練到接近 word embedding 方式的準確率結果，因為能夠 training 出接近 word embedding 的準確率，在 ensemble 上提供了不同模型的多樣化，讓 ensemble 的效果能更加有效。

IV. Experiment and Discussion

No.	model	Weighted Sum	Concat	Dimension	Window	min	kaggle
1	Embedding	X	1	100	5	1	0.36679
2	Embedding	X	1	100	5	1	0.38616
3	Embedding	X	5	100	5	1	0.40592
4	Embedding	X	5	50	5	1	0.41501
5	Embedding	X	8	50	5	1	0.41225
6	Embedding	X	6	64	10	1	0.42924
7	Embedding	X	6	64	20	1	0.43478
8	Embedding	X	6	50	6	1	0.42015
9	Embedding	X	6	64	20	1	0.43873
10	Embedding	O(a = 0.001)	6	50	20	1	0.36166
11	Embedding	O(a = 0.01)	6	50	20	1	0.44782
12	Embedding	O(a = 0.005)	4	50	20	1	0.47905
13	Embedding	O(a = 0.001)	4	50	20	1	0.47905
14	Embedding	O(a = 0.001)	6	50	20	1	0.483
15	Embedding	O(a = 0.005)	4	40	20	1	0.47865
16	Embedding	O(a = 0.0001)	4	50	20	1	0.47114
17	Embedding	O(a = 0.0025)	4	50	20	1	0.4869
18	Embedding	O(a = 0.0025)	4	50	20	1	0.48142
19	Embedding	O(a = 0.001)	4	50	20	1	0.47035
20	Dual Encoder	X	4	50	20	1	0.45415
21	Embedding	O(a = 0.0025)	3	50	15	1	0.48063
22	Ensemble	X	X	X	X	X	0.50079
23	Ensemble	X	X	X	X	X	0.49525
24	Dual Encoder	X	3	50	20	1	0.46758
25	Dual Encoder	X	3	50	20	1	0.47905
26	Ensemble	X	X	X	X	X	0.5245
27	Ensemble	X	X	X	X	X	0.52727
28	Ensemble	X	X	X	X	X	0.54387

Table 1 結果比較表

1. 句子合併

如同上述 II 部分的第 4 點, 轉詞向量處理, 從上表看出，我們亦嘗試多種不同的串接方式，並丟進 word2vec 進行訓練，如結果所示，我們認為在合併句子長度為 3 句且 overlap 重複合併時，不論是 word2vec 的參數改動、在 kaggle 的表現較佳。因此，我們最終的 model 即選擇以三句串接的方式來訓練 word2vec 模型。

2. 句向量合成

經由 reference[5]所提供的方法，我們在合成向量時，將分詞後的語詞向量做不同權重的相加，來達到強調重要語詞的功能。如下圖(引用自論文)，此權重則是經由 paper 所提到的公式，考量到了語詞的頻率計算而得。

$$\arg\max \sum_{w \in s} f_w(\tilde{c}_s) \propto \sum_{w \in s} \frac{a}{p(w)+a} v_w, \text{ where } a = \frac{1-\alpha}{\alpha Z}$$

No.	Weighted Sum	kaggle
9	X	0.3873
12	O(a = 0.005)	0.4791
21	O(a = 0.0025)	0.4806

從上表顯示，經由 Weighted Sum 過後所預估的結果，在 Kaggle 上的表現，遠比沒有做 Weighted Sum 的結果高出許多，如此彰顯將關鍵詞權重提升的重要性。而經由試誤法，所得到的結果將參數 a 調整為 0.0025 的結果最佳。

3. Ensemble

本次專題的 Ensemble 方法是透過結合 Word Embedding 和 Dual Encoder 的預測結果檔案，篩選多個 csv 檔進行選項投票，並任定得到最高分的選項為答案。本次專題的 Ensemble，本組使用多重排列組合與試誤，挑選出了最佳的七個 csv 檔並合併結果。

csv 檔案訓練模型內容概述：

i. hope0625.csv ： 主要是利用三個 word2vec 的模型, 對每一個句子產生相對應的 3 組向量，並相加做平均，最後再用 Cosine Similarity 得到 最後相似度最高的答案。
ii. 0624_49.csv : concat=3 句，dimension=50，window=15，weighted sum=0.0025 的模型參數。
iii. 0627_dualencoder_test_yuchi.csv：使用了 Dual Encoder 模型，RNN 的架構主要為一層 300 的 LSTM Layer 加上 300 的 GRU Layer，訓練了 11 個 Epoch。
iv. 777.csv: 使用了 Dual Encoder 的模型，但 RNN Model 改成一層 256 的 LSTM Layer 加上 128 的 GRU Layer，分別訓練了 5 個 Epoch
v. 0628_dualencoder_test_yuchi.csv: 使用了 Dual Encoder 的模型，RNN 的 Model 主要架構為一層 200 的 LSTM Layer (dropout=0.4)加上 200 的 GRU Layer (dropout=0.2)，訓練了 13 個 Epoch
vi. 49.csv: concat=3 句，dimension=49，window=15，weighted sum=0.0025 的模型參數。
vii. 0614_dualencoder_test1_yuchi: 使用了 Dual Encoder 模型，RNN 的架構主要為一層 300 的 LSTM Layer，訓練了 12 個 Epoch。

V. Conclusion

本次專題分析電視劇上下文對話，並分析選項與題目的相似性，來選擇對應的問答結果。在資料前處理的部分，我們使用 Jeiba 套件，並使用套件的參數調整與多重子句合併，製造出予 Word2vec 的訓練資料，以此增強前後句的關聯性分析，並從結果顯示，如此較以單句為訓練資料的結果較為優秀。在訓練 word2vec 語詞向量化模型後，利用此模型所造出的字典語詞向量，將測試資料之題目與選項，分別投影至固定長度的數字向量，數個詞向量合併成句向量的過程中，我們參考了[5]的做法，賦予每個詞向量出現頻率的權重分布，並且依照權重相加，構成句向量。最後透過選擇 RNN 模型或是向量間的相似度，預測相對應的問答句，最終完成作答。

實驗分析數據如 Table 2 結果比較表。從表中顯示出，強調句間的前後關係非常重要，因此我們採取了多重子句串接，更加強模型的強健性；句向量合成的部分，相較於等比種看待每個詞向量，我們認為透過 reference[5]提出的的方式，加上詞頻和權重之間的關係，如此更能找出句子中較重要的詞彙，並透過這些重要的詞彙，來找出對應的解答；就單一模型而言，雖然 Dual encoder 在 kaggle 上的分數結果略低於 word embedding 向量相似度比較的結果，但訓練時間 word embedding 遠比 dual encoder 還要短。

在 ensemble 的階段，若加入 Dual encoder 能比單純只有 word embedding 的 ensemble 效果更好，因此認為用 RNN 這種句子整體訓練的模型雖然準確率不高但能辨別出 word embedding 以詞向量總和而不考量詞的順序可能所不能辨別的對話。

最終，嘗試多種 Dual Encoder 與向量相似度檔案的接合與投票，我們分別選出兩者向量相似度與三者 Dual Encoder 的分析結果進行接合，做出能夠超過 5 成準確率的模型。

VI. References

1. Genism word2vec documentation
   https://radimrehurek.com/gensim/models/word2vec.html
2. 自然語言處理入門- Word2vec 小實作
   https://medium.com/pyladies-taiwan/%E8%87%AA%E7%84%B6%E8%AA%9E%E8%A8%80%E8%99%95%E7%90%86%E5%85%A5%E9%96%80-word2vec%E5%B0%8F%E5%AF%A6%E4%BD%9C-f8832d9677c8
3. 以 gensim 訓練中文詞向量
   http://zake7749.github.io/2016/08/28/word2vec-with-gensim/
4. Jieba 結巴分詞 Github
   https://github.com/fxsjy/jieba
5. a simple but tough-to-beat baseline for sentence embeddings
   https://openreview.net/pdf?id=SyK00v5xx
6. Implementation of dual encoder using Keras
   https://basmaboussaha.wordpress.com/2017/10/18/implementation-of-dual-encoder-using-keras/
7. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems
   https://arxiv.org/pdf/1506.08909.pdf
8. 聊天机器人中的深度学习技术之二：基于检索模型的实现
   http://www.jeyzhang.com/deep-learning-for-chatbots-2.html