Q1. Data preprocessing

a. 在試過 Spacy 及 NLTK 之後，我發現有時候 NLTK 會有將 wasn’t 斷成 was 及 n’t 等等的失誤，因此最後採用 Spacy 進行斷詞。

b. 嘗試過使用全部的 Text 及 Summary，但發現 Text 的長短差異相當大，因此 Padding 後的 Text 可能會使用過多 Memory，因此最後選擇最大上限 300 字。Summary 最大上限則是設定為 80 字，但每個 batch 中的 Summary 經常不會超過 50 字，因此幾乎不會因超出上限而被刪減。

c. 使用過 GoogleNews-vectors-negative300、glove.6B.100d 及 glove.840B.300d，最後的 glove.840B.300d 涵蓋的字較多。最後刪減 Pre-train embedding ，只留下 train、valid、test 的 Text 內所有字，減少 Pre-train embedding 所佔的空間。

Q2

a.

- w = Embedding(T), where T is model input
- Using Drop out prevent overfitting
  d(w) = Dropout(w)
  Using Bidirectional LSTM
- $h_t^{\rightarrow}, c_t^{\rightarrow} = \mathrm{LSTM}(d(w_t), h_{t-1}^{\leftarrow}, c_{t-1}^{\leftarrow}), h_t^{\leftarrow}, c_t^{\leftarrow} = \mathrm{LSTM}(d(w_t), h_{t-1}^{\leftarrow}, c_{t-1}^{\leftarrow})$

where w_t is the word embedding of the t-th token after drop out, and the arrow pointing the direction of the parameter passing.

- Output = Linear(L), where L is the output of LSTM

b.

"rouge-1": 0.19334546674937414,
"rouge-2": 0.028597444271097845,
"rouge-L": 0.1311372342894552

三項指標皆優於 Baseline(18.5, 2.6, 12.3)

c.

Loss Function 選用 BCEWithLogitsLoss 其 pos_weight 設定為 6.84。

d.

optimizer 選用 Adam，learning rate 設定為 0.00001，batch size 為 128

e.

Post-processing strategy : 取分數最高的兩句做為 extractive summarization

Q3

a.

Encoder :

- w = Embedding(T), where T is model input (text).
- Using Drop out prevent overfitting.
  d(w) = Dropout(w)
- $h_t^{\rightarrow}$= GRU(d(w_t), $h_{t-1}^{\leftarrow}$), $h_t^{\leftarrow}$= GRU(d(w_t), $h_{t-1}^{\leftarrow}$)
  where w_t is the word embedding of the t-th token after drop out, and the arrow pointing the direction of the parameter passing.

- E_H = tanh (Linear(h_f, h_b)) , where h_f is the last of the forwards RNN's hidden, and h_b is the last of the backward RNN's hidden.

Attention :

- α_t = Softmax(Linear(tanh (Linear (s_{t-1}, E_H))))

where α_t is the attention weight on the t-th word in target sentence, s_{t-1} is the previous hidden state of the decoder, and the E_H is the hidden state from Encoder.

Decoder :

- E(Y_t) = Embedding(Y_t), where Y_t is the t-th target word.
- Using Drop out prevent overfitting.
  X_t = Dropout(E(Y_t))
- $W_t = \alpha_t$ * E_H, where W_t is the weight on the t-th word in target sentence, α_t is the attention weight, and the E_H is the hidden state from Encoder.
- S_t = GRU((X_t, W_t^T),S_{t-1}), where concatenate the input word X_t and the Transpose W_t.

Seq2seq :

- E_O = Encoder (T), where E_O is the Encoder output, and T is the model input.
- α = Attention (E_O), where α is the attention weight
- Output = Decoder(Y, E_O, α), where Y is the target word.

b.

"rouge-1": 0.2576681530258227,
"rouge-2": 0.07242720161051307,
"rouge-L": 0.21364763862555033

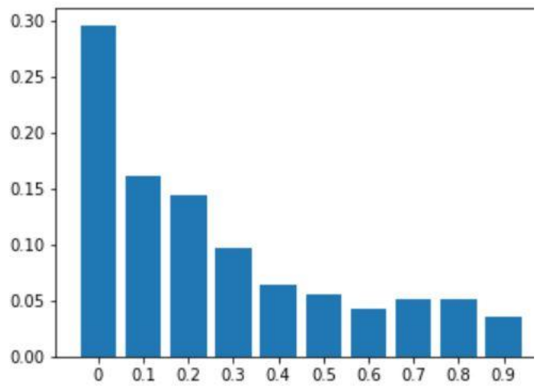
Seq2seq + attention 通過 baseline (25, 5, 20)

c. CrossEntropyLoss

d. Adam, learning rate 0.0001, batch size : 8 (模型龐大，容易 Out of memory)

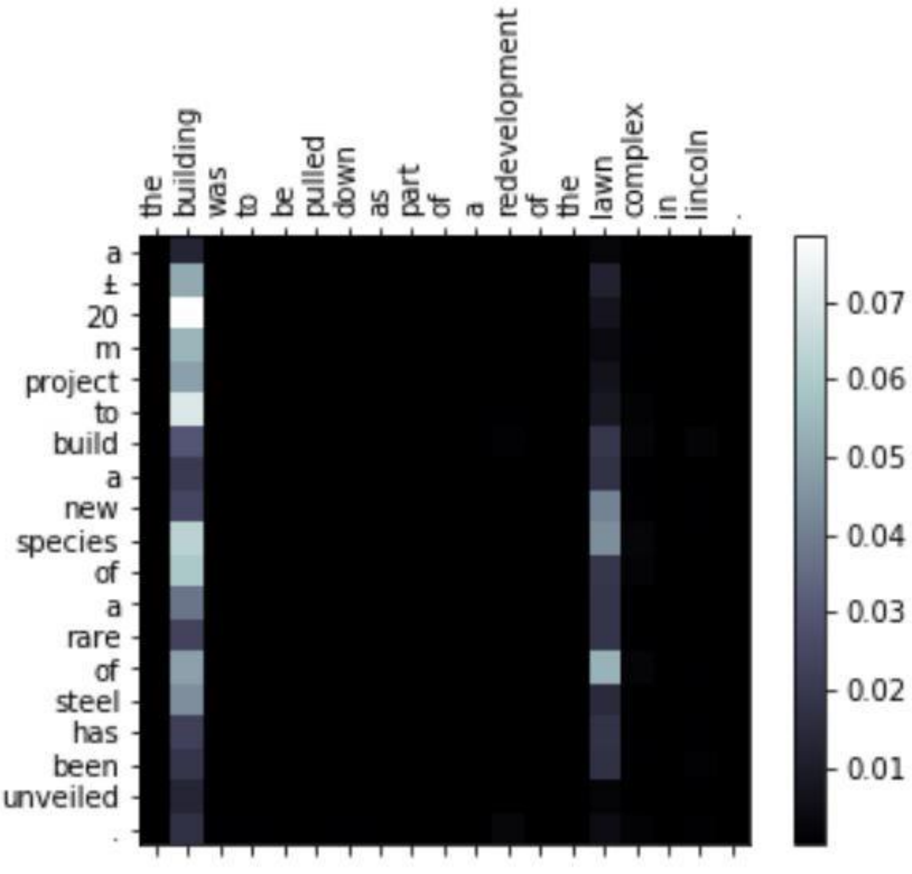
Q4

下圖為 extractive summarization 的相對位置分布圖。X 軸為其相對位置 (取到小數第一位)，而 Y 軸則代表該相對位置被預測為 extractive summarization 的比例。



由此圖可知，當每次挑選兩句做為 extractive summarization 時，有高機率會挑出第一句 (位置 0)，從分布也可以看出，愈靠前面的句子愈有可能被挑選。

Q5



1. 由右側的顏色表可以看出，顏色愈淺的字代表其 attention weight 愈高，該字也愈重要。
2. 從 inputs 列可以看出來，相較於 the was to be 這些無關緊要的字，在生成 summary 的時候 building 及 lawn 分配到的注意力更高，對生成出較好的 summary 更有幫助。

Q6 Rouge-L

Rouge-L = Longest Common Subsequence, based on F-measure, compute the recall and precision.

LCS(X, Y) is the length of Longest Common Subsequence between Sequence X and Sequence Y.

R_{lcs} = LCS(X, Y) / m , where m is the length of Sequence X.
P_{lcs} = LCS(X, Y) / n , where n is the length of Sequence Y.
$F_{lcs} = (1 + \beta^2) * R_{lcs} * P_{lcs} / (R_{lcs} + \beta^2 * P_{lcs})$, where β is set to a very big number.