Q1 Tokenization

Bert Tokenization 首先先透過 _clean_text() 將無效文字與空格進行清除，接著對於中文進行 Tokenization，使用的是 _tokenize_chinese_chars()，其做法為判斷是否為 Unicode 中的 CJK character 裡面，屬於中文字的字符，是的話將其 Tokenization，不是的話便直接輸出。

再來切割標點符號，最後透過空白把剩下尚未 Tokenization 的 Text (如英文單字) 進行 Tokenize。

若是使用 bert-base-chinese 則是對中文字、英文字以及數字分別採不同的處理方式，以下分別描述 BERT Tokenization 如何處理這三種不同的情況。

1. 中文字：BERT Tokenization 如上面所述，若被判斷為中文字，則是直接把每一個字切割。

2. 英文單字：直接以空白做切割。

3. 數字：數字在 Tokenize 階段並沒有做太多的處理，最多就是由於會先處理標點符號，因此會將小數點與原本數字分開，舉例而言：

6 → “6”

2000 → “2000”

3.14 → “3” “.” “14”

較為特別的地方則是在 encode 的部分，例如 2400000 則在 encode 時會分成

“2400”及 “##000” 其中##則為串接上一個字符。

Q2 Answer Span Processing

1. Tokenization 之後如何轉換 start/end 的位置

首先先將 Answer start 前的文字進行 Tokenization，而得到的長度+1 即是 Answer start 開始的位置。

在將答案的部分進行 Tokenization (context[start : end])，得到 Answer 在 Tokenization 後的長度。

最後將 Answer start + Answer 在 Tokenization 後的長度即可得到 Answer end 的位置。

2. 我並沒有再將 Span position 做轉換，而是直接將預測出來的 start 與 end 直接對應到 Tokenization 後的 context 的位置，例如 context[pred_start : pred_end] 再將得到的資訊做 decode，並去除空白，而得到最終答案。

Q3 : Padding and Truncating

1. maximum input token length : 512，包含[CLS]，[SEP]等字元。

2. Combine context and question step : 首先先將 context 及 question 進行 encode，會得到包含[CLS]及[SEP]的結果，並將 question 的[CLS]拿掉。

如果 encode 後的 context length + question length ≤ 512，則直接將兩者合併。

如果 encode 後的 context length + question length > 512，則：

如果 question 超過 30 字，則只取前 29 字 + [SEP]，若不滿 30 字則全取

接著將剩餘的空位留給 context，得到最終的 inputs。

3. padding inputs : 將全部 inputs 補 0 補至該 batch 裡面的 max inputs length，再透過 mask 避免 padding 影響模型學習。

Q4　Model

1. How to predict if the question is answerable or not
CLS_output = Softmax (Linear (BertModel(inputs, segment, mask)))
首先先使用 Pre-trained model : BertModel，產生 hidden layer (size : [512,768])，取出第一個 hidden layer(CLS)，再透過兩層 Linear，最後輸出一組 2 維的數字 [x, y]，其中 x, y 分別代表為 unanswerable 或為 answerable 的機率。最後取出機率高的代表其結果。若結果為 answerable，則另外判斷 若 end span < start span 或是 answer length >80，則將其改為 unanswerable。

2. How to predict the answer span
Start_output = Softmax (Linear (BertModel(inputs, segment, mask)))
End_output = Softmax (Linear (BertModel(inputs, segment, mask)))
以 BertModel 產生的 hidden layer (size : [512, 768])，分別將其放入兩層 Linear，最後輸出為 512 維，分別每個字為 Start / End 的機率，取出最高機率作為 Start / End span。

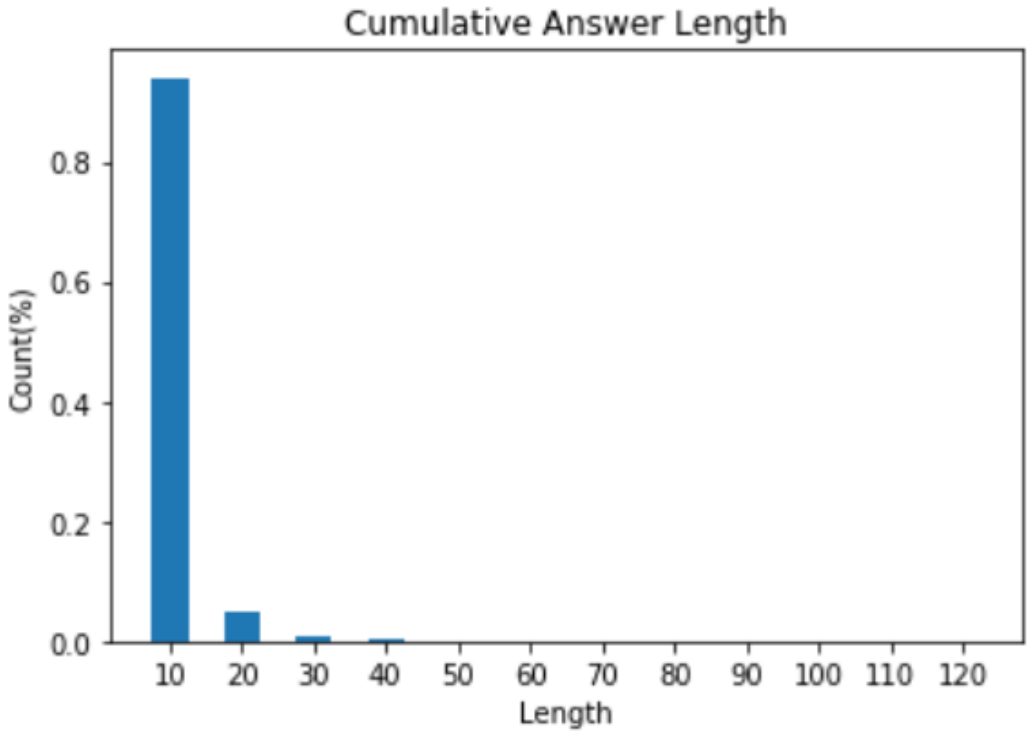
3. Loss function
Answerable Task : BCEWithLogitsLoss with pos_weight = 0.4447869
Start span loss : CrossEntropyLoss
End span loss : CrossEntropyLoss

將三者分開計算，分開進行 Backpropagation。

4. Optimizer :

Adam, with learning rate 0.00001, and weight decay 0.0001

Q5 Answer Length Distribution



1. 結果如上圖，X 軸代表 Answer token length，Length 10 表示長度在 0~10 之間，Length 20 表示長度在 11~20 之間，以此類推。Y 軸表示該長度區間的累積數量占所有資料的比例。

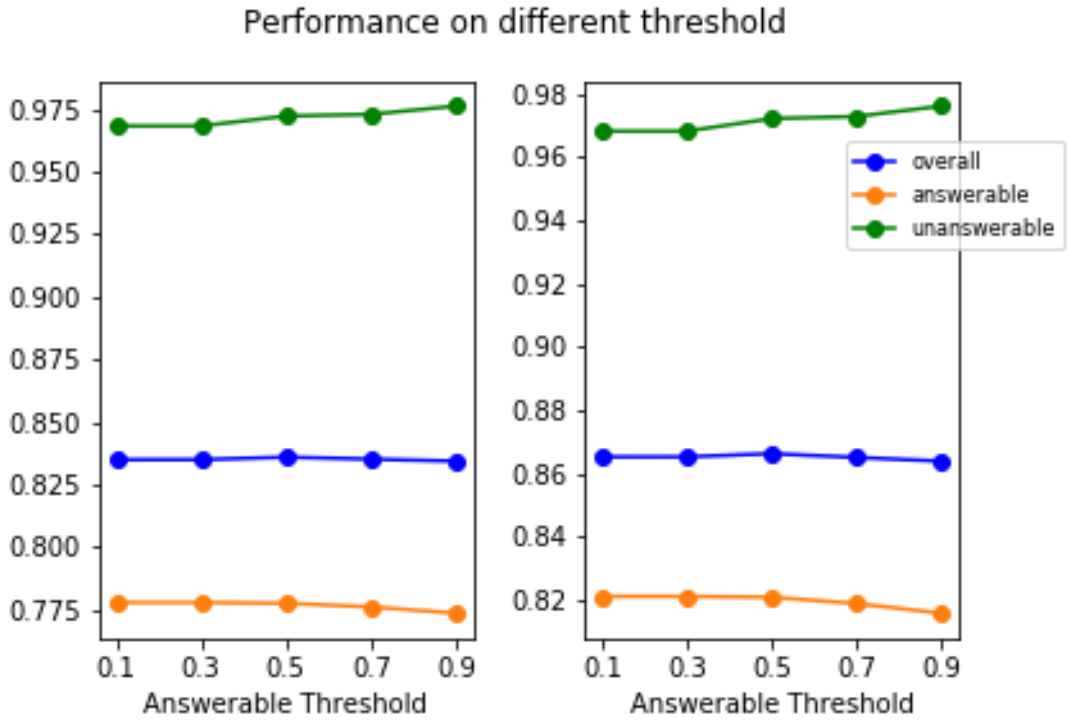
2. 此一結果相當明顯，幾乎所有答案的長度都是 0~10 字，偶爾有 11~30 字，因此我在做 post processing 時幾乎只留下小於 30 字的答案做為 answerable。

Q6 : Answerable Threshold

1. 如何訓練及預測 answerable

在訓練時將 answerable label 成 [0, 1]，unanswerable label 成 [1, 0]，最後 model 會 output 出 [x, y]，分別代表兩者的機率，若 x 大則判定為 unanswerable，y 大則判定成 answerable。

2.



Q7 Extractive summarization

Extractive summarization with Bert : 作法如上次雷同，透過 pre-trained Bert model，inputs 長度可為 encode 後的 512 字，再將 Bert model 所 output 的 hidden layer (size : [512,768]) 通過 Linear 層，接著透過 Sigmoid 層算出每個字是否可能為 Extractive summarization。Label 的方法則是將 text 裡面每個字都做 label，若該句為 summarization，則該句每個字都 label 為 1，其餘則 label 為 0，再透過 BCEWithLogitsLoss 來調整正負比例不平衡的問題。

最後 Post-processing 的方法則是有很多種，例如挑出每個字平均最高分的句子，或是挑出每個字平均大於某個值的句子。