ADL HW2 Report

資管四 B05705025 陳漢威

tags: ADL

1. Tokenization

Describe in detail how BERT tokenization works.
BERT使用了叫WordPiece的tokenize技巧，是BPE(Byte-Pair Encoding)的衍生，BPE會把完整的word拆成不同part的sub-word做成token，讓tokens之間能保有部分語意的情況下能進行組合衍生出不同的語意與文法上的變化，這種做法比起記住所有可能的word還要節省詞表的數量，而且在training的時候效果表現更好。
WordPiece的技巧只會應用在數字與英文的tokenize上面，中文則不太適用這個技巧。
BERT的Tokenizer也內建了Corpus方便我們將tokenize後的tokens轉換成indices

What happens when the method is applied on different strings (e.g. Chinese, English or numbers)?
Bert的tokenizer對不同的語言或型態都會做處理:

- 中文的話會以character-based一個字一個字切成tokens
- 英文與數字則是會會用WordPiece切成sub-word的tokens

2. Answer Span Processing

How did you convert the answer span start/end position on characters to position on tokens after BERT tokenization?
我會依據answer text經過tokenize過後拿去跟所有的context tokens做比較，找到完全吻合的token sequence後記錄新的start index & end index，如果找不到就讓start & end = 0 (代表無解)

After your model predicts the probability of answer span start/end position, what rules did you apply to determine the final start/end position?

1. 一開始會由argmax選擇score最大的index做為start與end的依據
2. 再看end-start 是否大於30，或是有start > end
    - 如果start = [CLS] -> 重新找score其次大的start
    - 其他情況就重新找start&end組合直到找到一組under constraint而且能最大化分數總合的start & end
3. 比較start&end==0的分數以及上面找到的最大化非0 start&end組合的分數誰比較大:
    - if start&end==0的分數比較大則unanswerable
    - else 預測有答案並output最大可能解

3. Padding and Truncating

What is the maximum input token length of bert-base-chinese?
Bert-base-chinese的maximum input是512個tokens

Describe in detail how you combine context and question to form the input and how you pad or truncate it.
我的作法是優先保留所有的questions text，並考慮進[CLS], [SEP]的長度之後，剩下的全部填滿contexts，如果contexts長度超過剩下的長度時會有兩種做法:

- 如果在training step，我會把確保答案的start與end都有被包含到，所以我會平均的取答案前後在長度限制內的contexts做為training input(Answer token會被包在正中間)
- 如果是testing step，我會直接從前面開始truncate符合的長度

4. Model

How does the model predict if the question is answerable or not?
我使用的做法是把unanswerable的start與end皆標示為0(指向到[CLS])，並在post-processing的時候做判斷

How does the model predict the answer span?
使用BertForQuestionAnswering，最後的一層hidden會經過linear轉換成(input_dim * 2)的vector，代表了input中每個字start&end的scores，ideally的情況 start&end取argmax找到的index range會在contexts之中，而被囊括在start index與end index中的就是answer span

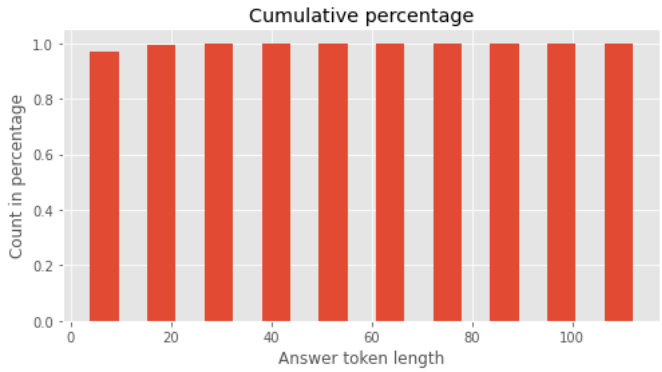
What loss functions did you use?
此次使用default的CrossEntropyLoss，會分別計算start label的loss與end label的loss做相加，再用加總的值放入optimzer

What optimization algorithm did you use?
此次使用Adam做為optmizer
經過多次實驗並參考ntu cool中的討論後發現:

- 跑完一個epoch後把bert的embedding層fix住不更新再train會有助提升模型表現
- Learning rate 做schedule逐漸降低會有助於降低overfit的速度
  - 從epoch1的5e-6 ->epoch2的5e-7
- 根據幾次的training經驗最多跑1.5~2個epoch就會收斂，再多的話就會overfit

5. Answer Length Distribution

Plot the cumulative distribution of answer length after tokenization on the training set.



- 有將近96.7%的答案token長度落在10個字以內，99.38%的答案在20個tokens以內
- 在post-processing的時候應該要選擇長度小於20的start-end組合，有機會能讓predict出來的結果變好

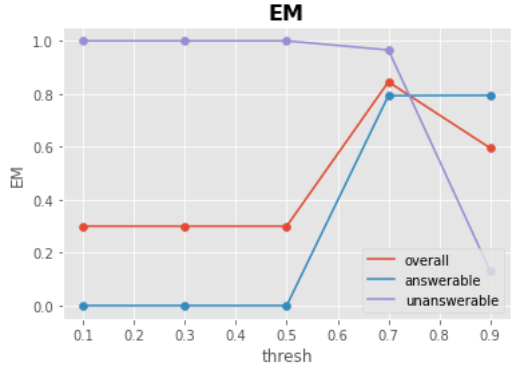
6. Answerable Threshold

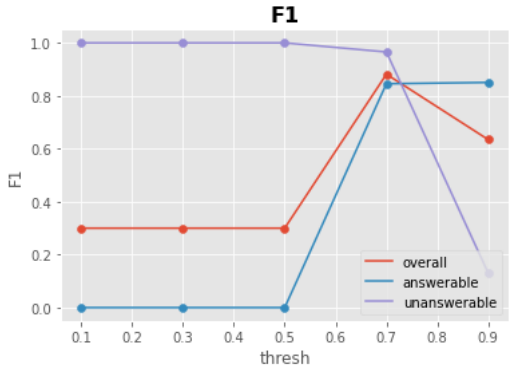
For each question, your model should predict a probability indicating whether it is answerable or not. What probability threshold did you use?

因為我的作法並不是一般的classification，而是看start&end argmax都是0的情況做為判斷answerable的依據，因此我採用了以下的做法得出類似classification的機率分布:

- 給定任一一個input，unanswerable = start_score[0]+end_score[0] (start&end都是0的分數)
- answerable = argmax(start_score)與argmax(end_score)兩個的分數相加(兩個的index都不可為0)
- 將兩個scalar concat起來放進sigmoid裡面得到類似classification的分布
- 並用這個分數比較threshold並進行預測

得出來的結果如下:





結論:

- 在我的情況來說，thresh設在0.7的時候效果最好，在0.5之前會嚴重傾向output出unanswerable，0.7之後會因為強制output出答案而造成unanswerable的F1與EM下降
- 用Threshold=0.7的效果並不差，但實驗發現直接比較answerable或unanswerable的score大小來決定要不要output答案會稍微比設定threshold好

7. Extractive Summarization

Describe in detail how you can frame the extractive summarization task and use BERT to tackle this task?
我認為extractive summary的task與QA很相似，或許可以直接input整篇文章，並用bert標出最重要部分的start與end，但這樣可能會遇到的問題是沒辦法切割段落(因為重要的句子可能會四散在文章各處)
比較好的作法可能是做到sentence-level的representation與classification，讓bert分類這個句子是或不是重要的，這樣就可以解決上面無法切割段落的問題