Applied Deep Learning

Homework 2

R07631043
毛信翔

Tokenization

- Bert-base-chinese

vocab_size 21128、unused 1~99 (可以針對目前的 task 新增想要的 word，embedding 會是 random initialization)

- What happens when the method is applied on different strings ?

tokenize 會針對目前 sentence 中可以 tokenize 出來的最長的 word 為優先。英文、數字的部分如果是連再一起的但 vocab 裡面沒有，會再分割成 vocab 裡面有的 token，然後用##來連接原本應該相連的 word。

Ex: `('appledas')→ ['apple', '##da', '##s']`
`('3219fa') → ['321', '##9', '##fa']`

- 補充
  1. 英文的部分還需要設定 do_lower_case=True，不然大寫字母會被 tokenize 成 UNK
  2. 中文的部分就會直接從 vocab 裡找看看有沒有字沒有的話，就是 UNK。
  3. 有一部分的 answer text 有空格，這部分在 tokenize 之後會被去掉，所以會造成和 answer text 不一樣的結果(差在空格)。

- On task tokenize:

針對 training data 為了後續 tokenize 方便我會先針對 data 的 answer start、answer end 做 sorting 之後才進行後續處理，這樣比較方便處理 answer start 一樣、跟 answer text 影響 tokenize 的問題(ex: data 9 paragraphs 9 qas 3 的 entry 因為答案是 9 但在文章中是 90...，如果直接針對 context tokenize 會有問題，所以必須把 answer 也考慮進來這樣 tokenize 出來的結果才會包含 answer)。

Answer Span Processing

- How did you convert the answer span start/end position on characters to position on tokens after BERT tokenization?

  延續上一題，sort 完之後先針對每個 paragraphs 作處理，先對部分 context(index 0->answer start)做 tokenize，再對 answer 做 tokenize，再對 answer end 之後的 context 做 tokenize，其中 answer start 一樣的 qas 做額外處理，照這樣的方式一直延續下去做完整筆 data 的處理，而在做 tokenize 的過程中就會記錄 answer bond，當成 start/end index。

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

  我使用 answer start/end topk(30)當成所有可能的選項，並且以 end length-start length 大於 0 小於等於 40 當成必要的條件，最後從 answer start/end topk(30)中取出相加 probability 最高的選項。

Padding and Truncating

- What is the maximum input token length of bert-base-chinese?

  Max input token length = 512

- Describe in detail how you combine context and question to form the input and how you pad or truncate it.

  這部分參考討論區助教 basline 作法，其中只有 context 會被 truncate，question 不會變動，padding 則是會加在 question 後面，由於使用這個 truncate 所以在 training data 部分有些 answer bond 不在 context 的情況出現，這部分會直接刪掉不使用，38480 entry 中刪掉 1017 個。

  Padding、truncate 結果舉例:

  input_id = [cls, context_id….(truncate), sep, question_id….sep, pad….]

  attention_mask=[1……..,1(question_id)…,1(sep),0……]

token_type_id=[0……,0(sep),1(question_id)….1(sep),0(pad)….]

context_mask= [-inf,0…..,-inf(sep),-inf(question_id)….]

answer_bond=tokenize_answer_bond

Model

- How does the model predict if the question is answerable or not?
  model 會先從 bert 的 output 通過 Linear_answerable(hidden_size,1)把 hidden 變成一個 unit，然後取 sequence 第一個位置來預測是否為 answerable，經過 sigmoid 後就會是一個 0~1 之間機率，>=0.5 是 answerable、<0.5 是 unanswerable。

- How does the model predict the answer span?
  model 會先從 bert 的 output 通過 Linear_start ,Linear_end(hidden,1) 變成一個 unit，然後 start/end 分別取整個 sequence 出來然後加上 context_mask 來讓 non-context 部分的 probability 為 0，經過 softmax 之後會是機率分布，取 start/end 最高的 probabilty 就是 start/end span.

- What loss functions did you use?

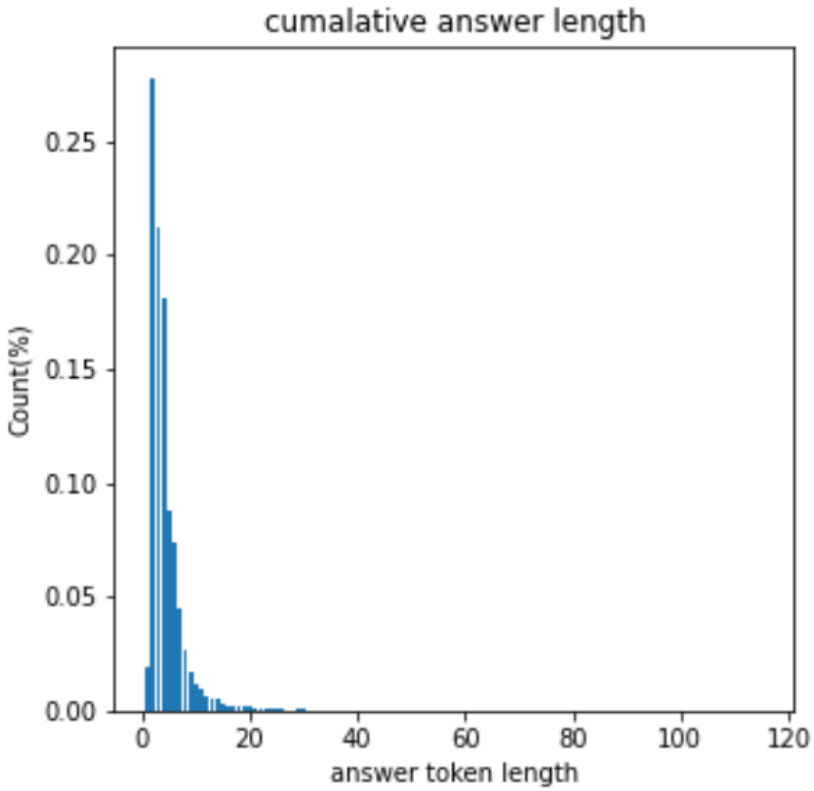
  Answerable: BCEWithLLogitsLoss(pos_weight=0.4)

  Answer start/end: CrossEntropyLoss(ignore_idx=-1) (ignore_idx 用來處理 unanswerable(unanswerable answer start/end 皆為 -1))

- What optimization algorithm did you use?

  Adam lr=2e-5

Answer Length Distribution

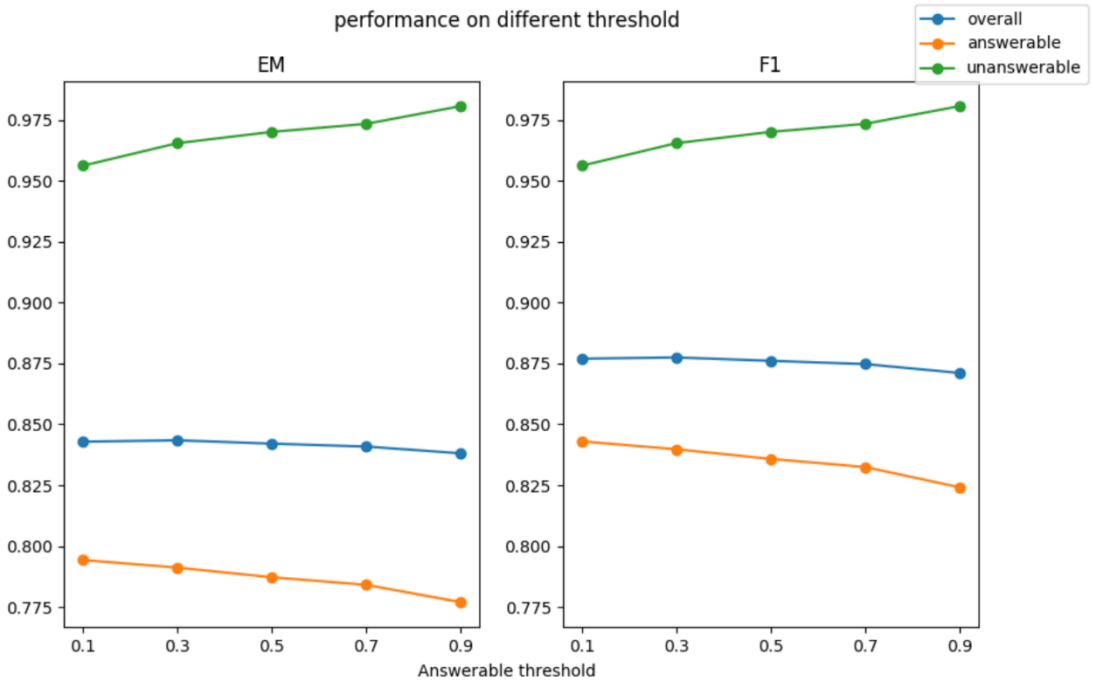


從分布可以看到在 length>40 之後幾乎數量很少，answer length 最長是 115 只有一筆，而在 answer length=3 的時候數量最多，超過了 answerable 的 1/4，根據這張圖我選擇了 length 30,35,40、topk(30)為 start/end 的可能性、answerable threshold 0.5 下去計算，發現在這三個數值下 performance 結果差距不到 0.1%，所以最後區間設在 30。

Length: 30 的結果

Overall → em: 0.8420738974970202, f1:0.8761242943028446
answerable →em: 0.7871736662883088, f1: 0.835814329602871
unanswerable →em: 0.9701986754966887, f1: 0.9701986754967
answerable accuracy →0.9513309495431068

Answerable Threshold



從圖中可以看出 answerable 跟 unanswerable 的 EM、F1 是不可兼得的，雖然不同 threshold 之間 overall 差距不大，但 threshold 越低 unanswerable EM、F1 越低、answerable EM、F1 越高，answerable 的 F1 又會比 EM 高個 5%左右，unanswerable 的 F1、EM 則差距沒有那麼明顯。

Extractive Summarization

可以由兩種方法處理 extractive summarization 的問題:

1. input 不需要任何的 cls 或是 sep，input 就是文章 tokenize 過後的 id、再經過一些 padding、truncate 處理放進 bert，再從 bert 的 output sequence 經過 Linear(hidden,1)，來決定 token include 或 not include，最後進行 BCELoss 最佳化參數，所以這個方法是針對每

個字進行 binary classificaiton 的問題，最後再進行 post processing 看是否要把這個句子納入 extractive summary，.

2. input 不需要任何的 cls 或是 sep，input 就是文章 tokenize 過後的 id、再經過一些 padding、truncate 處理放進 bert，從 bert ouput sequence 分別經過 start/end Linear(hidden,1)變成很像這次 start/end span 的問題，再用 CrossEntropyLoss 來最佳化參數，最後可以根據想要的 start/end span 做一些 post processing