Neural Networks and Applications_10920IEEM537300

Project 2: **Simple English Reader**

Group 2

109136502 **陳宗聖** 109033573 吳育葦 109033804 **張軼峯**

Test data input 流程

Test data 格式需依照助教所給範例格式輸入;
並在第一個 block 中分別確認輸入之克漏字與閱讀測驗 .txt 檔名無誤。

Project 2: Simple English Reader 作業目標

- Input: English reading test (CAP for junior high school students)
- Output: Correct answer choice (single answer for each question)
- Including reading tests (articles) and cloze tests.
- Public dataset and public pre-trained models are allowed.
- Evaluation: Not only based on answer, but also AI inference flow.

Implementation and Pytorch Model

本作業中我們主要採用 Huggingface 的 transformers 與 SBERT 的 Sentence-Transformers 之 package 來實作。

❖ Article Test

閱讀測驗之情境與 Extractive Question Answering task 的情境類似，需要在給定的文章中，提取、找出答案，回答相關問題。Huggingface 提供了相當簡潔的 API，使用 pipeline 即可快速解決各種NLP task；而我們選擇手動指定 model 與 tokenizer，使用上可有較大彈性。

首先 instantiate tokenizer 與 model，定義選用之 pre-trained BERT 模型。輸入 article 與 questions，透過 Tokenizer 將 text 輸入至 model，Tokenizer 可將 string 轉為對應 ID 與 back，負責 encoding 與 decoding。使用特定 model 需指定 tokenizer 以輸出正確的 model-specific token type，這邊指定 tokenizer 與 model 為相同 naming。

此時 Model 會依照 input 的 question sequence，計算整段 article sequence token 之分數，分別有 start score: 對應是否為答案句的開頭位置、end score: 對應是否為答案句的結尾位置，最後藉由 softmax 計算得出最大概率為答案句的開頭、結尾之 token 為何，最後
轉換為 string 並輸出答案句。

❖ Cloze Test

克漏字測驗之情境與 Masked Language Modeling 的情境類似，皆為在一個 sequence 中有一挖空的 (masking token) 單字，需填入適當的單詞或片語。

第一步一樣先 instantiate tokenizer 與 model, 接著定義要輸入的 sequence, 其中包含一個挖空 (masked token), 並用 `tokenizer.mask_token` 取代。Tokenizer 會將此 sequence encode 成 ID 並讓 model 得知挖空位置, pre-trained 好的 model 則根據其儲存的詞彙計算每個 token 可能出現在此挖空位置的分數, 最後 output 出 k 個最可能之答案。

❖ Semantic Textual Similarity

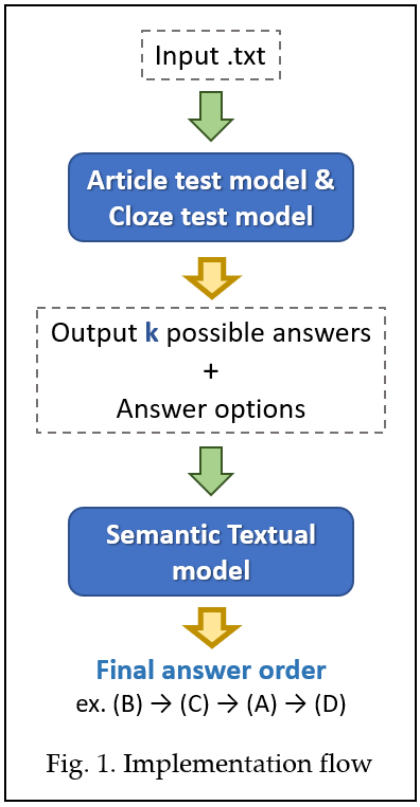

Fig. 1. Implementation flow

我們的工作流程實作如 Fig. 1. 所示, 在 article 與 Cloze model 輸出 k 個預測答案後, 我們使用 SBERT.net 的 `Sentence-Transformers`, 來將預測答案與題目上之選項進行相似度比對, 比對結果會 output 0 ~ 1 的數值表示相似程度，越高代表越相似，1 則為一模一樣之答案。

因有 k 個預測結果需要分別與題目上之 A, B, C, D 四個選項做比對, 相似度比對後的分數, 我們也嘗試了不同的計算方式, 詳細實驗結果可參照 Section: Performance of Different Model 之內容。

❖ Input/Output

- cloze:
  將 input file 讀進我們的 code 當中, 並利用每一行第一個字母分辨題目或選項 ("Q" 為題目, "A" 為選項), 最後把題目和選項分別儲存成 list , 並放進 cloze test model 得出 k 個預測結果 , 經過 cloze test 與 sematic textual similarity 的計算, 將每一題答案依照 similarity score 的高低順序依照題號存成 output 的 txt 檔 (檔名為原 input 檔 + _output)。

- article:
  讀取 input txt 檔的方式和 cloze 相同, 但多了 article 的 list 以及每篇閱讀測驗的題數，將 article (context) 和 question (query) 放進 SBERT.net 的 `Sentence-Transformers`, 經過 sematic textual similarity的計算, 將答案依照 similarity score 的高低順序存成 output 的 txt 檔 (檔名為原 input 檔 + _output)。

Performance of Different Model

因實作中嘗試了不同的 model，故猜測若使用多個 model 來做 ensemble，是否能提高準確率? 但在比較其結果後發現，較實際的作法可能還是針對單一 task 選用適當model、採取適當計分策略，結果會較 ensemble 來得好，所以最後並沒有使用 ensemble 的技巧。我們對於 model 的選用策略，我們希望能夠找到一個最好的 model 應用於兩種不同的問題上面，並且使用了108及109年的英文會考題本作為 test data 測試其答對的題數。

1. Cloze: 在克漏字的模型選用，我們發現使用 uncased model 會比 cased model 的 performance 好，我們推測其原因在於，克漏字的題目敘述短，具有大寫英文字母的出現情況較少，故使用完全沒有大寫字母的 uncased model 會較為適合。
   並且我們也對 4 種不同的模型做答對題數的測試，我們可藉由 Fig. 2. 圖表整理出以下結論:
   a. large-model performance 較 base-model 好
   b. 專門使用於 mask 預測的 performance 最好
   c. 使用閱讀長篇文章的 Fine-tune model 表現較為糟糕

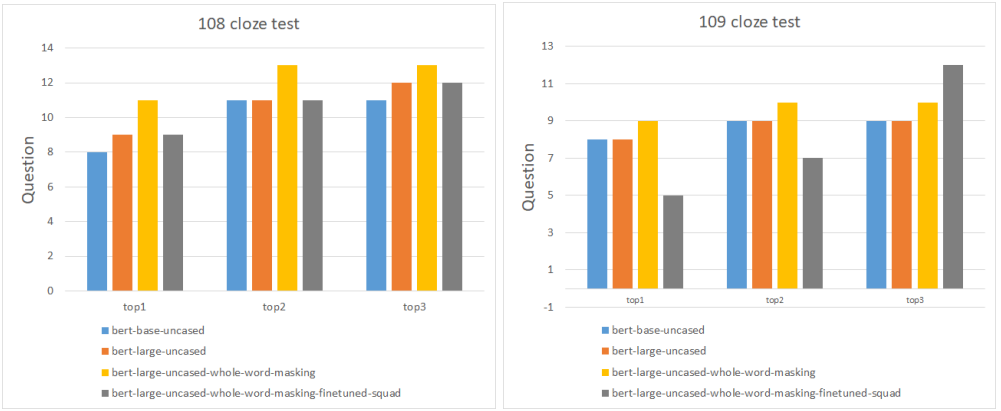

Fig. 2. Cloze test model comparison

故我們最後採用 `bert-large-uncased-whole-word-masking` 作為克漏字的 model。

2. Article: 因為題目的形式偏向長文以及針對題目做問答的動作，故我們使用 question answering 的 model 進行預測，而我們發現使用閱讀長篇文章的 Fine-tune model 表現比其他的模型要來的好或是持平，故在閱讀題型方面，我們採用 `bert-large-uncased-whole-word-masking-finetuned-squad` 作為預設的 model。

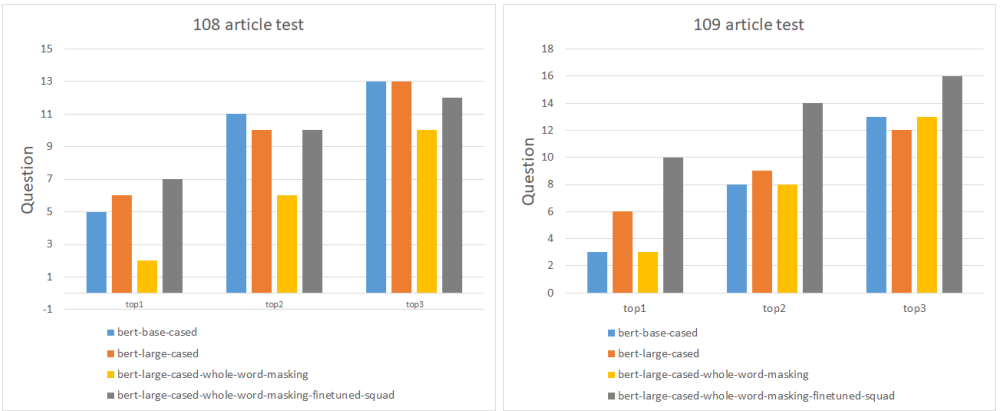

Fig. 3. Article test model comparison

Discussion

1. Bert fine tune

為了使 model 預測的更準，我們嘗試 Bert 的 Fine tune，希望透過文法的 dataset，來加強模型檢查語法是否正確的功能。而我們所用來訓練的資料集為 `The Corpus of Linguistic Acceptability`，並且希望透過此資料集，協助我們判斷克漏字裡關於文法的題目。

Sentence	He has listens to me.	I has listened to you.	I can listen (to) you.	I can hear to you	I can hear you.
Predict	Right	Right	Right	Wrong	Right
Ground truth	Wrong	Wrong	Wrong	Wrong	Right

從上面測試的表格來看，我們可以觀察到此資料集應該對於時態的變化較為不敏感，並且在介係詞的判定上是較好的。

但我們發現以下幾點問題:

a. Bert 本身的模型所訓練的 data 足夠完備，若 Fine tune 的資料集不夠龐大，或是參數有調整適當，會將原本的模型往更差的方向負優化。
b. 資料集所提供的錯誤文法資料大多為嚴重錯誤，對於時態等錯誤並不足夠，故無法應用於我們這次的作業當中。

故多方考量下，我們認為這項作法不具有通用性，所以並未採用。

2. summarization 表現差

由於一次將一篇文章丟進 question answering 的 model 當中，有可能因為沒辦法抓到有用的feature，故我們曾嘗試先將文章丟進 `pipeline("summarization")` 的模型裡，嘗試模擬 Feature extraction 的步驟來得到結論，將此結論作為 question answering model 所需要的文章。但我們實測過後發現，答對率明顯下降，並不符合我們的需求。

3. 我們的實驗結果發現，article test 答對率相較於 cloze test 低了許多，推測可能原因為:
   a. article test 部分題型，BERT 模型無法很好的掌握，如: 新詩的文章編纂方式，對於標點符號或是句子的分隔，更甚至可能因為文意的表達過於隱晦。使 article test 時無法取得良好的結果。
   b. 我們最終所使用的模型為 `bert-large-uncased-whole-word-masking-finetuned-squad`，由於其 Fine tune 的資料集為 SQuAD (Stanford Question Answering Dataset)，基於維基百科上的文章所進行訓練的，我們認為其 training domain 與英文會考的題本無論在文字的艱澀度、文章類型的變化等皆相差甚遠。

Reference

https://huggingface.co/transformers/v3.1.0/index.html

https://www.sbert.net/index.html

https://towardsdatascience.com/checking-grammar-with-bert-and-ulmfit-1f59c718fe75