HW2 110652036 邱立昕

For Part 0: Briefly explain the method you implemented and give an example (such as the E.g in the remove_stopwords function) in the report:

text = text.lower()變成小寫

text = text.replace("
", " ") 移除

text = "".join([char for char in text if (char not in string.punctuation)])移除標點符號

english_stemmer = SnowballStemmer(language='english') 找出每個詞

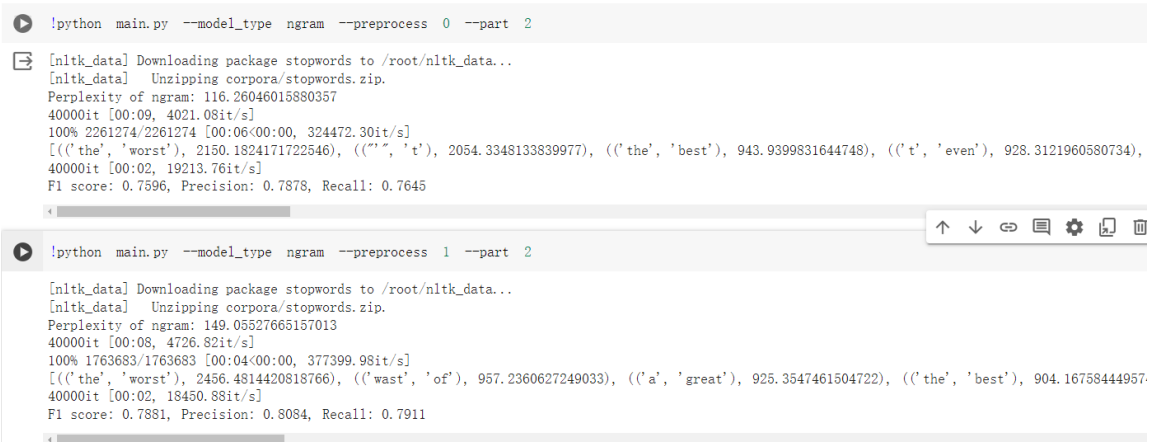
text = " ".join([english_stemmer.stem(i) for i in text.split()]) 用空格分開

For Part 1:

1. Briefly explain the concept of perplexity in report and discuss how it will be influenced.
   困惑度是一種用於評估語言模型（Language Model）或其他概率模型效能的指標。它通常用於衡量模型在給定一個事件序列後預測下一個事件的能力。困惑度越低代表模型對於該序列的預測能力越好。
   受到以下因素的影響：模型的訓練質量、模型的複雜度、訓練數據的質量和多樣性、事件序列的長度
2. screenshot the outputs and tell your observations about the differences in the perplexity caused by the preprocessing methods(1. without preprocess 2. with remove stopwords 3. with your method).

```
[3]  !python  main.py  --model_type  ngram  --preprocess  1  --part  1

     [nltk_data] Downloading package stopwords to /root/nltk_data...
     [nltk_data]   Package stopwords is already up-to-date!
     Perplexity of ngram: 149.05527665157013
```



For Part 2:

1. Briefly explain the two pre-training steps in BERT. Self-supervised Learning by Hung-yi Lee tutorial

   BERT 的預訓練過程包括兩個主要步驟

   Masked Language Model（MLM，遮罩語言模型）：

   BERT 的目標是預測句子中被隨機遮罩（mask）的一部分詞彙。這個遮罩過程使得模型必須根據句子中其他詞彙的上下文來預測被遮罩的詞彙

   Next Sentence Prediction（NSP，下一句預測）：

   用於訓練模型對於兩個句子之間的關係進行理解和預測。BERT 接收兩個句子作為輸入，並預測第二個句子是否是第一個句子的後續句子（即下一句）
2. Briefly explain four different BERT application scenarios

   文本分類（Text Classification）：

   BERT 可以用於文本分類任務，例如情感分析、文本推斷、主題分類等。

   問答系統（Question Answering Systems）：

   通過將問題和相關上下文文本餵入 BERT 模型，可以訓練模型來預測問答對。BERT 的雙向注意機制使其能夠理解問題和文本之間的關聯，從而能夠準確地回答問題。

   命名實體識別（Named Entity Recognition，NER）：

   通過微調 BERT 模型，使其能夠辨識和標記出文本中的命名實體，進而實現自然語言處理中的信息提取任務。

   語言生成（Language Generation）：

   通過將 BERT 模型與生成模型結合，可以生成具有語法結構和語義含義的自然語言文本。BERT 能夠生成具有上下文一致性和邏輯性的文本，提高了生成模型的性能和效果。

3. Discuss the difference between BERT and distilBERT?

   模型大小和效能：

   distilBERT 是對 BERT 模型進行了輕量化和壓縮，通常只有 BERT 模型大小的一半左右（例如 distilBERT-base 約為 BERT-base 的一半大小），從而在保持較小模型體積的同時，提供了相當不錯的效能。

   計算效率：

   distilBERT 的輕量化設計使得其計算效率更高，推理速度更快。由於模型更小，可以在更少的計算資源下運行

   預訓練過程：

   BERT 在預訓練過程中使用了完整的 MLM 和 NSP 任務，以學習語言表示。而 distilBERT 精簡了這些過程，僅使用了部分的 MLM 任務來訓練模型，從而加快了訓練速度和模型收斂。

   效能表現：

   distilBERT 在大多數自然語言處理任務上表現相當不錯，同時保持了較小的模型體積和更高的計算效率。但因為其模型規模較小，可能在一些需要複雜上下文理解的任務上稍遜於 BERT。

4. Screenshot the required test F1-score.

```
[ ]  !python  main.py  --model_type  BERT  --preprocess  0  --part  2
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Epoch 1 Training: 100% 5000/5000 [30:09<00:00,  2.76it/s]
Epoch 1 Testing: 100% 10000/10000 [01:49<00:00, 91.19it/s]
Epoch: 1, F1 score: 0.9329, Precision: 0.933, Recall: 0.9329, Loss: 0.2284
```

```
[ ]  !python  main.py  --model_type  BERT  --preprocess  1  --part  2
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
2
Epoch 1 Training: 100% 5000/5000 [29:21<00:00,  2.84it/s]
Epoch 1 Testing: 100% 10000/10000 [01:44<00:00, 95.54it/s]
Epoch: 1, F1 score: 0.9078, Precision: 0.9089, Recall: 0.9079, Loss: 0.2794
```

5. (BONUS 5%) Explain the relation of the Transformer and BERT and the core of the Transformer. Application of deep learning 7-1 to 9-1 by Vivian

   ransformer 的核心概念：

   自注意力機制（Self-Attention Mechanism）：

通過將輸入序列分別映射為三個向量：輸入向量（Query）、鍵（Key）向量和值（Value）向量，然後計算注意力分數並加權獲取加權後的值，以捕捉全局上下文。

多頭注意力機制（Multi-Head Attention）：

Transformer 使用多個注意力頭並行處理輸入，從而增強了模型對不同表示空間的表達能力。每個注意力頭產生獨立的注意力分數，最後將它們連接起來表示。

BERT 和 Transformer 的關聯：

BERT 的主要創新之一是將 Transformer 模型應用於預訓練的 NLP 任務中，並展示了 Transformer 模型在大規模文本表示學習中的卓越性能。BERT 採用了 Transformer 的編碼器結構，並通過遮罩語言模型（Masked Language Model，MLM）和下一句預測（Next Sentence Prediction，NSP）等任務來預訓練模型。

For Part 3:

1. Briefly explain the difference between vanilla RNN and LSTM.

   遞歸神經網絡 (RNN)：

   RNN 是一種基本的序列模型，其中每個時間步都有一個隱藏狀態。它的更新規則簡單，可以表示為：

   $$h_t = \sigma(W_{hx}x_t + W_{hh}h_{t-1} + b_h)$$

   其中 h_t 是時間步 t 的隱藏狀態，x_t 是輸入，W_{hx} 和 W_{hh} 是權重矩陣，b_h 是偏差項，σ 是激活函數（例如 sigmoid 或 tanh）。

   RNN 存在梯度消失或梯度爆炸的問題，導致長序列的建模能力受限。

   RNN 的記憶能力有限，對於長期依賴關係的序列效果較差。

   LSTM 有三個主要的閘控單元：遺忘閘（forget gate）、輸入閘（input gate）和輸出閘（output gate）。這些閘控單元能夠控制記憶單元的信息流動，從而避免梯度消失或梯度爆炸問題。

   LSTM 的更新規則如下：

   $$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
   $$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
   $$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$
   $$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$
   $$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
   $$h_t = o_t \odot \tanh(C_t)$$

   其中 f_t、i_t、o_t 分別是遺忘閘、輸入閘和輸出閘，C_t 是記憶單元，$\odot$ 表示逐元素相乘。

2. Please explain the meaning of each dimension of the input and output for each layer in the model. For example, the first dimension of input for LSTM is batch size.

   Input Dimension:

   序列長度（Sequence Length）

   這是指輸入序列中的時間步數或詞彙數量。對於一個序列模型，每個輸入都是一個時間步的特徵。

   批量大小（Batch Size）

   表示每個批次中有多少個序列或樣本。批次大小影響了模型的訓練效率和內存需求

   特徵維度（Feature Dimensions）

   如果輸入包含其他特徵（例如詞嵌入的維度），則這些維度用於表示每個時間步的特徵向量。

   Output Dimensions

   序列長度（Sequence Length）

   輸出與輸入具有相同的序列長度，每個時間步都有一個對應的輸出

   批量大小（Batch Size）

   與輸入相同，表示每個批次中有多少個序列或樣本

   特徵維度（Feature Dimensions）

   輸出通常具有與輸入相同的特徵維度，但可能具有不同的特徵表示或激活

Discuss the innovation of the NLP field and your thoughts of why the technique is evolving from n-gram -> LSTM -> BERT

n-gram -> LSTM：

n-gram 模型是一種基於統計的語言模型，通常用於處理自然語言的特徵提取和建模。它根據鄰近的 n 個單詞（n-gram）來預測下一個單詞，但它無法捕捉單詞之間的長期依賴關係和語義信息。

LSTM 專門用於處理序列數據。LSTM 通過引入記憶單元和遺忘門機制，能夠有效地捕捉長期的序列依賴關係，解決了傳統 RNN 的梯度消失和梯度爆炸等問題，使得模型能夠更好地理解語義和上下文。

LSTM -> BERT：

LSTM 當處理更大、更複雜的語言任務時，LSTM 的效果可能受限。

BERT 基於 Transformer 的語言模型，通過 Transformer 的注意力機制實現。BERT 通過對整個句子或文本進行訓練，可以更好地理解語境和語義，並在預訓練的基礎上進行下游任務的微調，取得了在多項 NLP 任務上的優異效果。

BERT 的創新之處在於雙向的語境理解和預訓練模型的可遷移性。它不僅通過 MLM 和 NSP 等任務進行無監督預訓練，還利用大規模無標籤的語料庫，使得模型能夠自動學習語言結構和語義信息，從而在各種 NLP 任務上取得優異的效果。

遇到問題:剛開始對各種方法都不了解，通過觀看老師給的影片才慢慢了解整個作業要幹嘛，但在實作時常會卡住，尤其是遇到 bug，因為要 call 的 function 很多，bug 出在哪一層並不一定，要 de 很久才行，比預計多花了很久才寫完