一、資料集分析

1. 含兩個 Column: text 和 generated
2. 487235 個唯一文本(text)，當該文本為 AI 生成，則 generated 值為 1；若為人類生成則 generated 值為 0
    (1) 0 的數量：305797；1 的數量：181438

text	generated
Cars. Cars have been around since they became famous in the 1900s, when Henry Ford created and built the first ModelT. Cars have played a major role in our every day lives since then. But now, people are starting to question if limiting car usage would be a good thing. To me, limiting the use of cars might be a good thing to do. In like matter of this, article, "In German Suburb, Life Goes On Without Cars," by Elizabeth Rosenthal states, how automobiles are the linchpin of suburbs, where middle class families from either Shanghai or Chicago tend to make their homes. Experts say how this is a huge impediment to current efforts to reduce greenhouse gas emissions from tailpipe. Passenger cars are responsible for 12 percent of greenhouse gas emissions in Europe...and up to 50 percent in some carintensive areas in the United States. Cars are the main reason for the greenhouse gas emissions because of a lot of people driving them around all the time getting where they need to go. Article, "Paris bans driving due to smog," by Robert Duffer says, how Paris, after days of nearrecord pollution, enforced a partial driving ban to clear the air of the global city. It also says, how on Monday, motorist with evennumbered license plates were ordered to leave their cars at home or be fined a 22euro fine 31. The same order would be applied to oddnumbered plates the following day. Cars are the reason for polluting entire cities like Paris. This shows how bad cars can be because, of all the pollution that they can cause to an entire city. Likewise, in the article, "Carfree day is spinning into a big hit in Bogota," by Andrew Selsky says, how programs that's set to spread to other countries, millions of Columbians hiked, biked, skated, or took the bus to work during a carfree day, leaving streets of this capital city eerily devoid of traffic jams. It was the third straight year cars have been banned with only buses and taxis permitted for the Day Without Cars in the capital city of 7 million. People like the idea of	0

3. 數據集處理方式：打亂資料順序，並且以 7:2:1 分割為訓練集、驗證集與測試集

二、Tokenizer & Embedding

1. 將文字資料轉換成機器學習模型可以理解的數值格式
2. Tokenizer：將文字資料轉換成整數序列，為每個單詞分配一個唯一的整數索引
    (1) num_words=MAX_VOCAB_SIZE：指定了詞彙表的最大大小，Tokenizer 只會保留最常見的 MAX_VOCAB_SIZE 個單詞。以此參數減少模型的複雜性和記憶體使用量。
    (2) oov_token="<unk>"，Out-Of-Vocabulary，當 Tokenizer 遇到一個不在詞彙表中的單詞時，它會用 <unk> 標記來代替
    (3) tokenizer.fit_on_texts(df['text'])：將詞彙建立索引，並且存放在 tokenizer 中。
    (4) sequences = tokenizer.texts_to_sequences(df['text'])：將文字資料轉換成整數序列

(5) padded_sequences = pad_sequences(sequences, maxlen=MAX_SEQ_LENGTH, padding='post')：用於填充序列，使其具有相同的長度。'post' 表示在序列的末尾填充零。

3. Embedding：將「詞彙索引」轉換為「詞向量」，也就是將整數序列（Tokenizer 的輸出）轉換為輸出是浮點數的密集向量，每個向量代表一個單詞的語義訊息

(1) 使用 Word2Vec 模型從文字資料中學習詞嵌入（word embeddings），並建立一個嵌入矩陣（embedding matrix），將 Word2Vec 模型學習到的詞嵌入儲存起來

(2) Word2Vec：分析大量的文字資料，學習單詞之間的語義關係，並將每個單詞表示為一個密集向量

(3) EMBEDDING_DIM ：定義詞嵌入向量的維度。如 EMBEDDING_DIM=100 表示每個單詞將被表示為一個 100 維的向量

(4) window：定義 Word2Vec 模型在訓練過程中考慮的上下文單詞的窗口大小如，window=5 表示模型將考慮目標單詞前後各 5 個單詞的上下文

(5) min_count：定義單詞在訓練資料中出現的最小次數，出現次數少於 min_count 的單詞將被忽略

(6) workers 定義訓練 Word2Vec 模型時使用的並行執行緒數量

三、模型架構

1. Attention-LSTM + CNN

(1) 結合 LSTM、Attention 機制、CNN

(2) 包含嵌入層、BiLSTM 層、Attention 層、CNN 層、全連接層 + Dropout。Bi LSTM 能捕捉全句順序資訊，Attention 能找出重要詞，CNN 能補充強調 1-gram 的模式，全連接輸出層可應用到多種分類任務。

(3) 嵌入層：使用 Word2Vec 預訓練的 embedding_matrix：將詞轉為向量。freeze=False，設置嵌入層的權重在訓練過程中會被更新，使得可以對預先訓練好的詞嵌入進行微調

(4) BiLSTM：用來捕捉序列資料的上下文語意，並透過雙向 LSTM 同時考慮過去與未來語境。

(5) Attention：為每個時間步的輸出加權，給予重要的詞更高權重。首先使用 tanh 將注意力分數縮放到 -1 到 1 的範圍，接著使用 softmax 將注意力分數轉換為權重。最終將權重與 LSTM 輸出相乘以取得「上下文向量（context vector）」

(6) CNN：創建多個 nn.Conv1d 層，filter_sizes 為[2,3,4]，表示將創建三個卷積層，其卷積核大小分別為 2、3、4。這些卷積層用於提取 1-gram 特徵

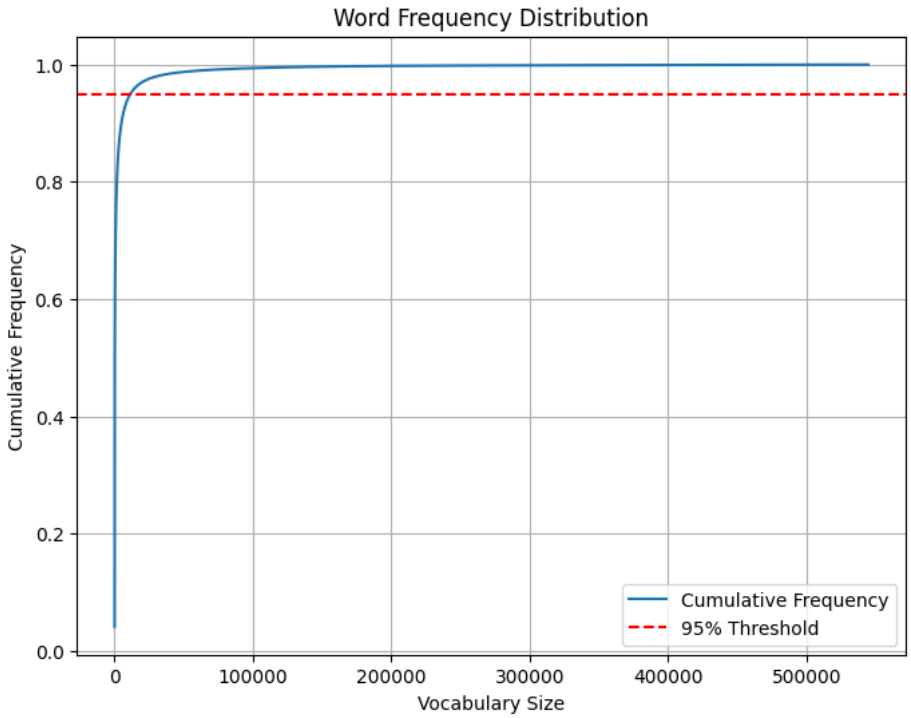
(7) 全連接層 ＋Dropout：全連接層輸出分類結果，並透過 Dropout 來減少 overfitting，最後使用 sigmoid 來計算二元分類的機率

四、超參數選擇

1. MAX_VOCAB_SIZE：決定了使用多少個最常見的詞，過小會丟失重要詞彙，過大會增加計算量且引入稀有詞彙（可能無助於學習）。選擇能覆蓋 95% 的文本的詞彙數量

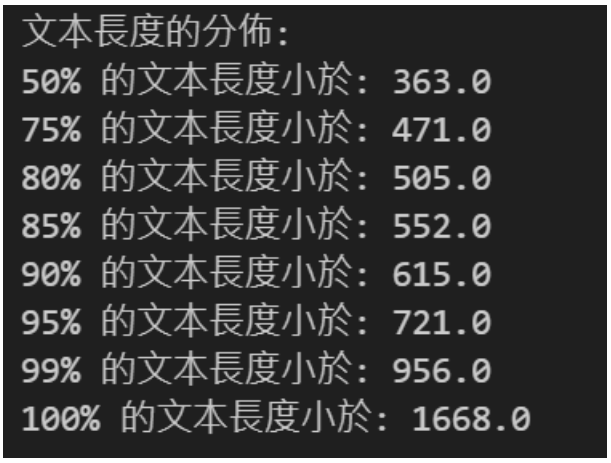
(1) 共有 544,828 個不同的詞

(2) 前 11,394 個詞已經覆蓋 95% 的文本



2. MAX_SEQ_LENGTH：影響 LSTM 的計算效率，過短會丟失訊息，過長會導致計算負擔大且無意義填充

(1) 透過計算文本長度分布來選擇最佳值



(2) 各測試之參數設定：為測試參數對於模型效果的影響，並追求在測試資料集較高的準確率。由測試一當作基準，測測試二~測試五為測試各個參數的影響；測試六選擇在測試一-五中表現得較好的參數做整體測試。

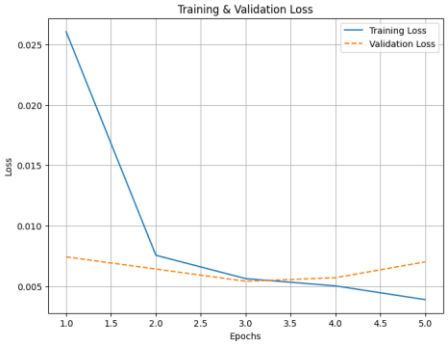
參數	測試一	測試二	測試三	測試四	測試五	測試六
MAX_VOCAB_SIZE	12000	30000	12000	12000	12000	12000
MAX_SEQ_LENGTH	300	300	650	300	300	650
EMBEDDING_DIM	100	100	100	200	100	200
HIDDEN_DIM	64	64	64	64	128	128
NUM_FILTERS	100	100	100	100	100	100
FILTER_SIZES	[2,3,4]					
BATCH_SIZE	32					
NUM_EPOCHS	5					

Test Loss	0.0054	0.0075	0.0037	0.0051	0.0042	0.0039
Test Accuracy	0.9987	0.9983	0.9991	0.9988	0.9990	0.9989

五、結果分析

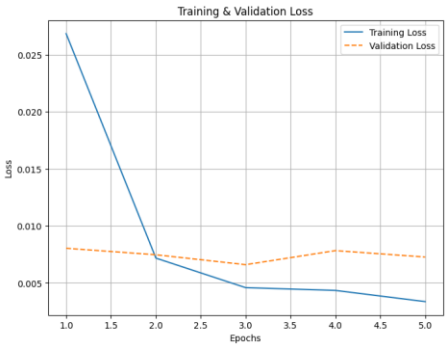
1. Attention-LSTM + CNN：

(1) 測試一：Test Loss: 0.0054, Test Accuracy: 0.9987

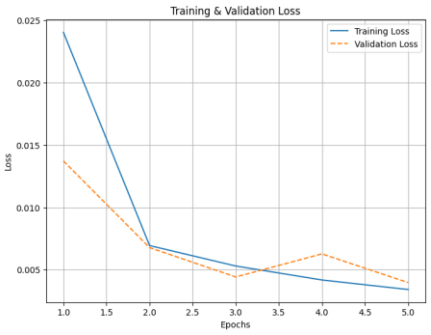


.

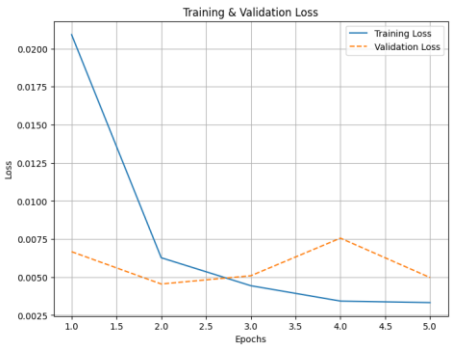
(2) 測試二：Test Loss: 0.0075, Test Accuracy: 0.9983



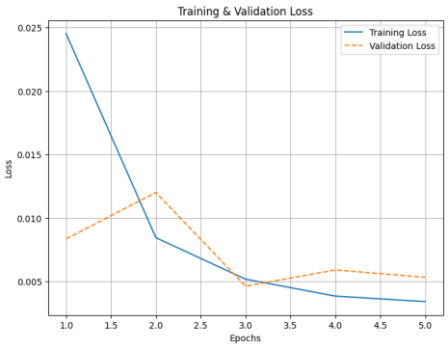
(3) 測試三：Test Loss: 0.0037, Test Accuracy: 0.9991



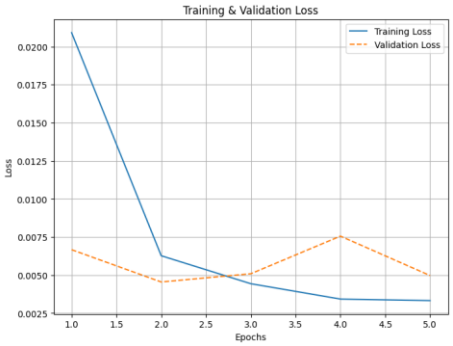
(4) 測試四：Test Loss: 0.0051, Test Accuracy: 0.9988



(5) 測試五：Test Loss: 0.0042, Test Accuracy: 0.9990



(6) 測試六：Test Loss: 0.0039, Test Accuracy: 0.9989



2. 準確率討論及可能的改進方法

(1) 整體來看，六次測試結果皆非常理想，Test Accuracy 皆高於 99.8%，顯示出模型具有極高的分類準確率，且在不同的參數組合下仍表現穩定，說明結合 BiLSTM＋Attention＋CNN 架構對於 AI 與 Human 文本判別具有極佳效果。

測試編號	Test Loss	Test Accuracy
測試一	0.0054	0.9987
測試二	0.0075	0.9983
測試三	0.0037	0.9991
測試四	0.0051	0.9988
測試五	0.0042	0.9990
測試六	0.0039	0.9989

(2) MAX_VOCAB_SIZE 的增加(測試二)，使得準確率降低，但這有可能是因為過擬合了，因為對照組設值為 12000 已經覆

蓋了 95%的文本。而測試二設為 300 不僅訓練時間較長，且增加了模型複雜度，導致結果較差。

(3) 測試三擁有最低的 loss（0.0037）和最高準確率（0.9991），其更動參數為 MAX_SEQ_LENGTH ，設值為 650，約可覆蓋 90%多的文本長度。而其他對照組設值為 300，僅可覆蓋約 50%的文本長度。因此在此模型和資料集中，文本長度覆蓋率為 90%時比 50%時，準確率較高。但該值須透過文本進行分析，不然反而會導致記憶體浪費或效能較差。

(4) EMBEDDING_DIM(測試四)設值為 200，相比對照組的 100 進步幅度不大，因此 100 可能就有足夠的語意表達能力了

(5) HIDDEN_DIM(測試五)設值為 128，而準確率相比 64 有所提高，這是因為增加 LSTM 隱藏層維度可以增強特徵提取能力。

(6) 測試六使用每個測試中有較佳準確率的參數，但是效果排名第三，因此模型的參數選擇需要綜合考慮，而非一昧的選擇在某個測試中表現較好的參數。