**Assignment1 Report**

Name: 李威辰

Institution (school): 國立清華大學

Student ID: 112062591

Platform (Colab/Kaggle/Local): Local

Python version: 3.12.3

Operating system: Ubuntu 24.04.2 LTS

CPU:

CPU 型號：AMD Ryzen Threadripper 7960X（24 核心 48 線程）

架構：x86\_64（支援 32/64 位元）

核心數 / 線程數：24 核心 × 2 線程/核心 = 48 線程

GPU requirement:

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

AI 產生的內容可能不正確。

1. Which embedding model do you use? What are the pre-processing steps? What are the hyperparameter settings? (5%)

Answer:

Embedding model : Word2Vec

pre-processing :

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

AI 產生的內容可能不正確。

這部分有使用ChatGPT詢問如何實作，上圖SentenceIter中進行pre-processing，先將文字做基本清理保留小寫英文字母，接著使用nltk去將文字裡的stopword去除，其中maxline是可以控制所取的行數如果None就是取全部。

hyperparameter settings: 以下是我測試baseline的參數，其餘都照原本函數的default，並且底下實驗若沒說明則參數跟baseline一樣。以下介紹此模型比較特定的參數:  
sentence : 作育訓練的語料。

vector\_size : 向量的維度，通常越大越好(但佔記憶體以及運算較慢)。

window : 前後字預測的寬度，2~5偏語法關係，5+偏語句關係。

min\_count=20：低頻詞門檻，丟掉出現次數小於 20 次的詞。

sg : 0時用「周圍的字」去預測中心詞，速度快、較省資源；對高頻詞效果好；大語料常用作預設/baseline。1時用「中心詞」去預測周圍的字特性：較慢、但對低頻詞/小語料常比較好；語義/類比任務有時表現更佳。

negative : 採納負樣本數的個數，以此作業詞彙關係向量訓練來說，負樣本是完全沒有出現在所選定中間詞彙的詞彙樣本集，這裡使用skip-gram，將中心word有關的(正樣本)向量內積再與負樣本的內積取負號總和相加，可以看出會把負樣本的內積數值取負號減少和正樣本內積數值增加提高字詞之間的關聯性。所以negative越大模型的準確率應當會提高但速度較慢。

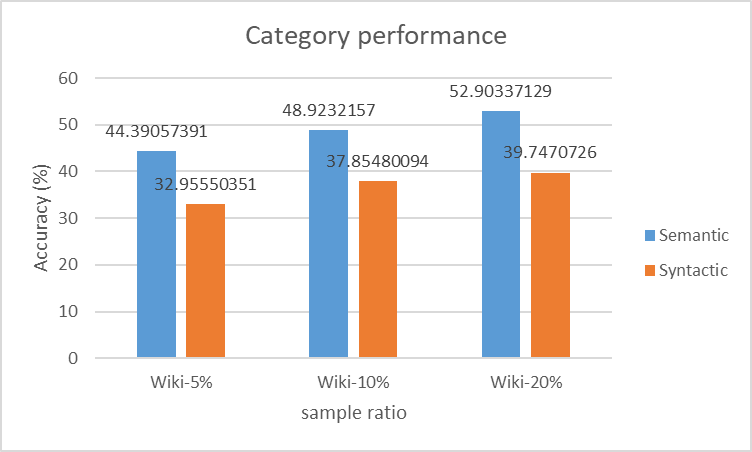
sample : ， 是詞彙 出現之頻率， 是可調控參數， 是丟掉詞彙 的機率，所以詞彙的出現頻率越接近 越容易被丟，可用來調控stop word等高頻出現無異議詞彙。

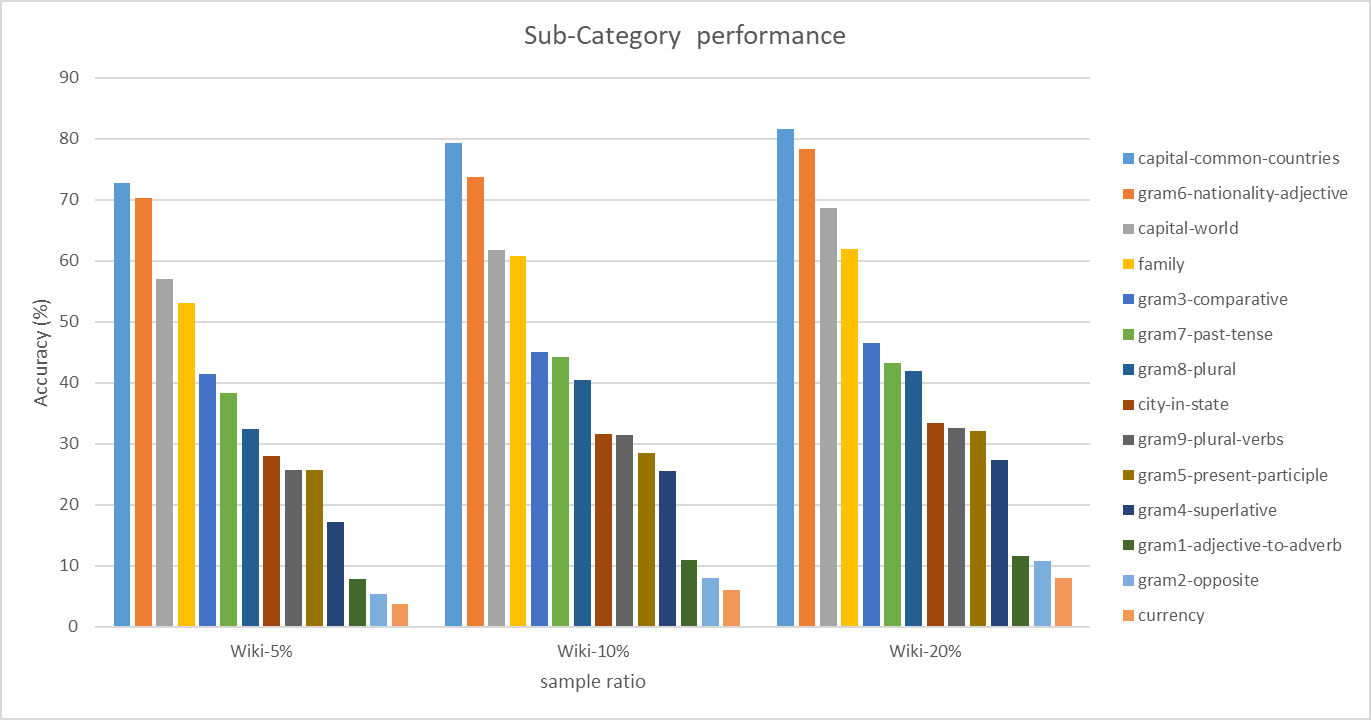
一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

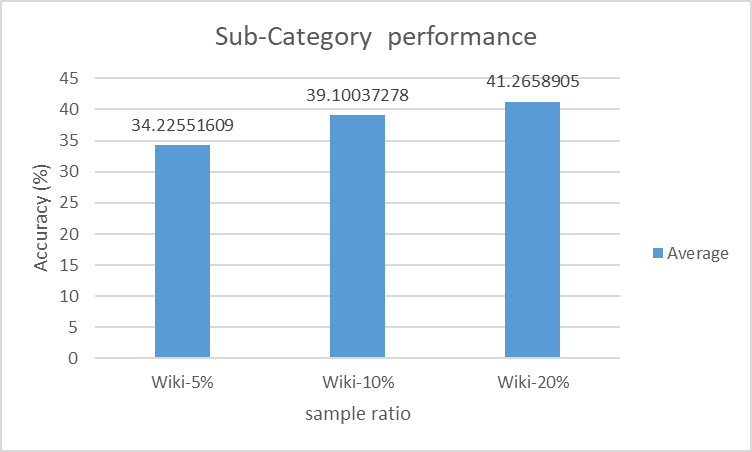
AI 產生的內容可能不正確。

1. What will the performance be like if you sample 5%, 10% and 20% of wiki text in TODO4? (10%, 3% for each)

Answer: 下面是用wiki text sample 5% 10% 20%所訓練出來模型的準確率，可以看到文本的資料詞彙越多，模型的準確率會越好，應該跟negative 和positive的資料數量有關，越多則調整數學式子的參數會更精確。



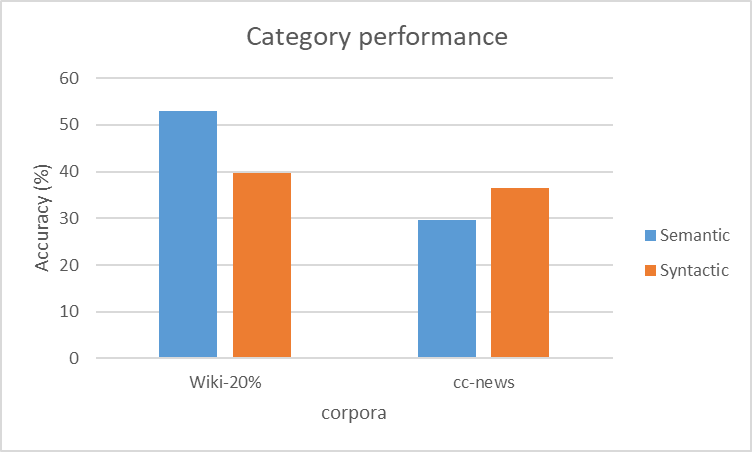


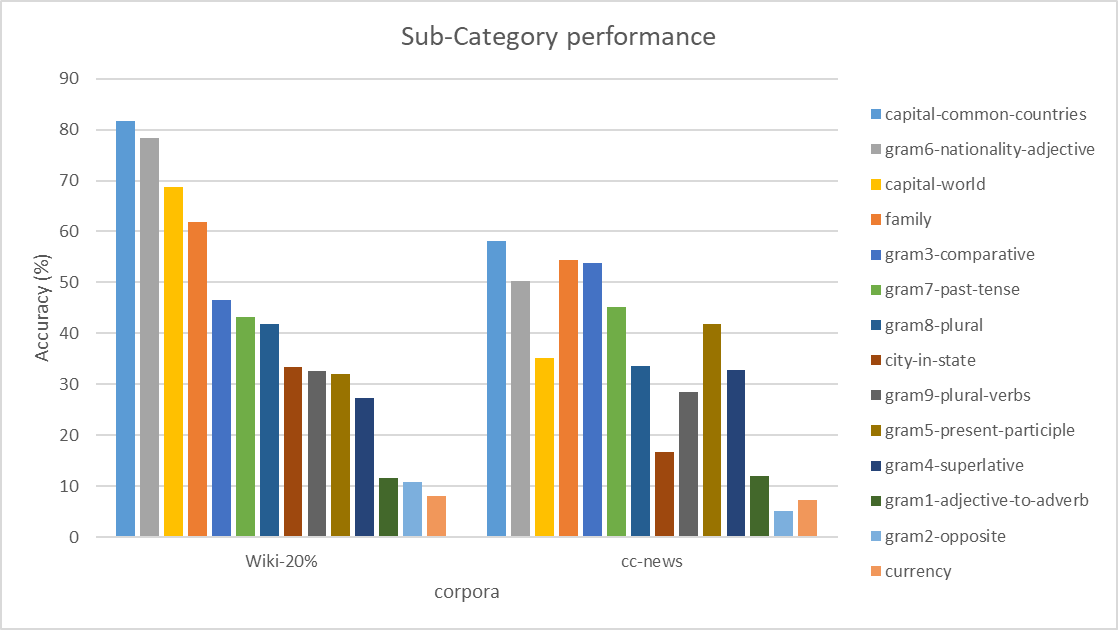
太多了所以平均

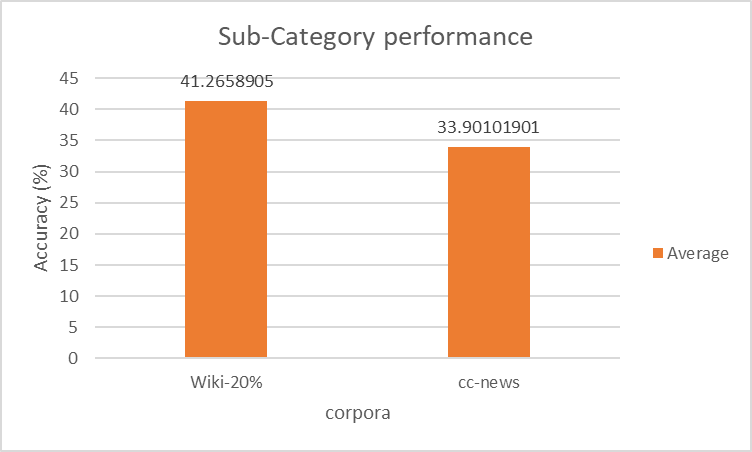
1. What is the performance for different categories or sub-categories when trained on different corpora? (15%)

3.1 Present your results. (5%)

Answer: 另一個語料庫取自CC-News







3.2 Introduce the corpus you selected and explain the differences between the Wikipedia corpus and your corpus. (including data size, topic difference, structural difference … ) (5%)

Answer:

|  |  |  |
| --- | --- | --- |
|  | Wiki-20% | CC-News |
| data size(line) | 1,124,733 | 708,241 |
| Topic | 內容偏中立、百科全書（地理、歷史、科學、文化…） | 英文新聞文章（政治、商業、體育、科技等） |
| structural | 章節、段落、列表 | 標題、引導語、短段落 |
| 專有名詞 | 通常為標準名稱 | 重複轉載、多樣拼寫 |
| 詞彙與內容相關性 | 較乾淨相關性高 | 含版權聲明、記者署名等較語內容無關之詞彙 |

3.3 Explain why the accuracy increases or decreases. (5%)

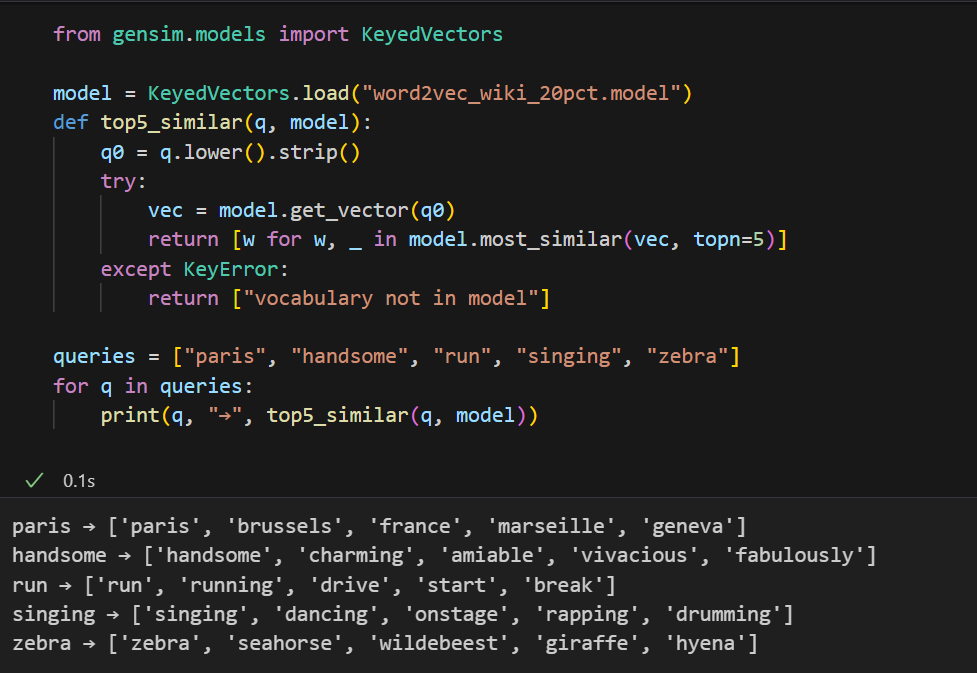
Answer:

語意類（semantic）：Wiki-20% 較好，因為其內容專有名詞(像是地名)相較CC-News標準化，CC-News容易OOV出現未訓練詞彙，所以performance會較差。

語法類（syntactic）：CC-News 較好，因為新聞動詞與生動的形容詞副詞使用種類較多，所以訓練語法成效較好。

以Subcategory去看的話可以發現Wiki-20% 較好，可以用每項的細列條去看，發現CC-News對專有名詞(capital-world、city-in-state等)會有不同的名詞表達，所以performance 明顯較標準化的Wiki-20%差，所以拉低了總體準確。

1. Select a few words and use their embeddings to retrieve the five most similar words and present the results. What do you observe? (10%)

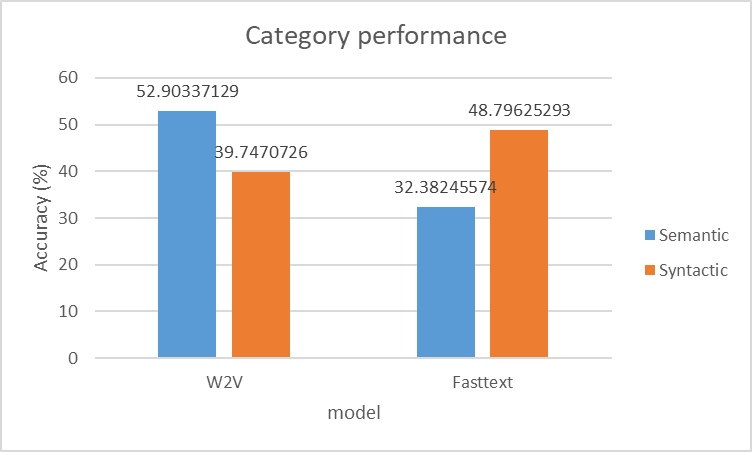
Answer :

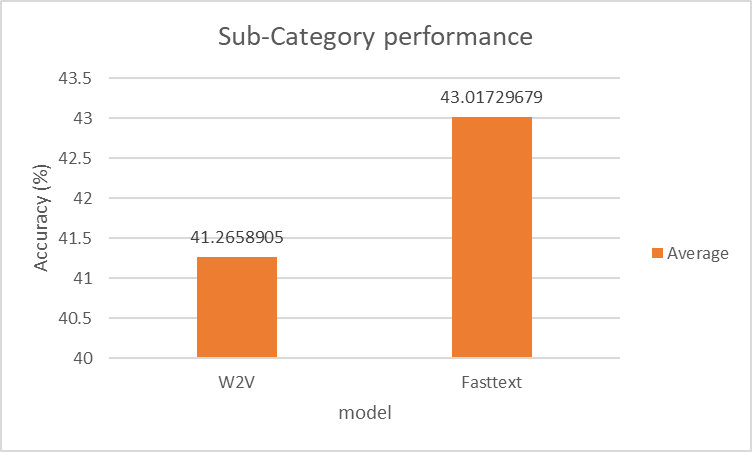
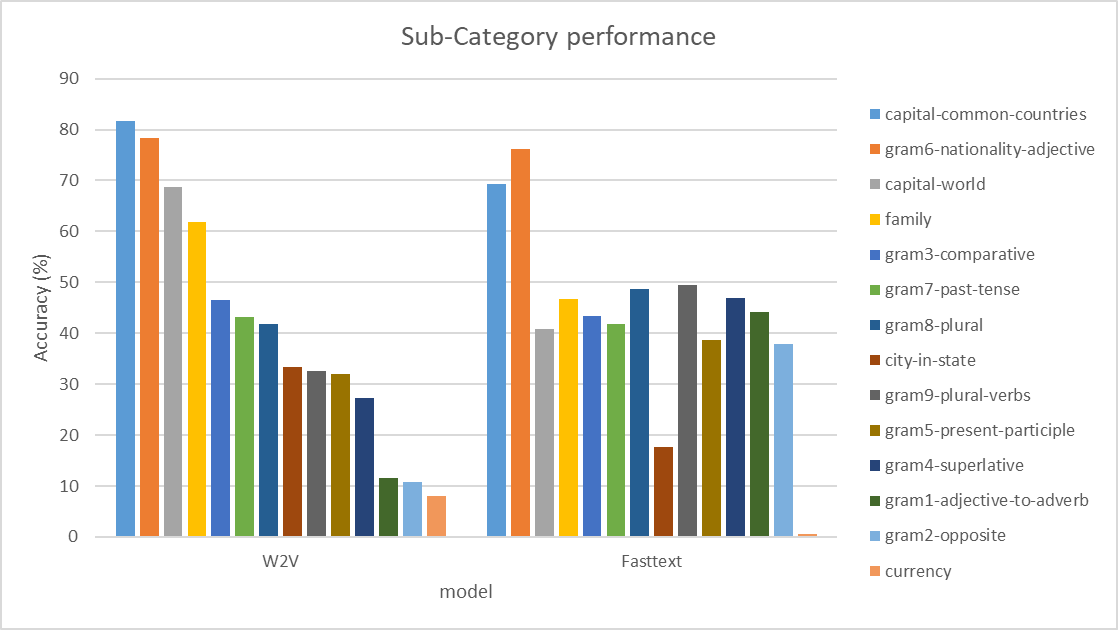
選了五個英文詞去跑top5\_similar函數(使用20%wiki的W2V model)會回傳前五相似的詞彙，選了專有名詞paris和一些形容詞動詞名詞，發現第一相似的詞彙是自己本身，然後詞性並沒有完全考慮像是sing-singing、handsome-fabulously，然後關聯性也不太一樣，paris 和bruessls是首都而paris 和Marseille都是France裡面的區域，然後zebra後面關聯的沒有什麼跟horse有關，都是列出與斑馬類似有表面線條紋路的動物，這點滿奇怪的，因為我是在用語言文字模型，很有可能跟wiki有用動物的外觀分類有關。

1. Anything that can strengthen your report. (5%)

Answer:

1. 這裡有去做另一個vector embedding model : fastText的比較實驗，fastText與W2V不太一樣的是向量組成，fastText有額外將每個word拆成字母片段做成vector並允許詞彙有OOV的狀況，所以在預測更general 的文章段落字詞fastText會比較穩定。以下是使用20%wiki text 的Accuracy比較:

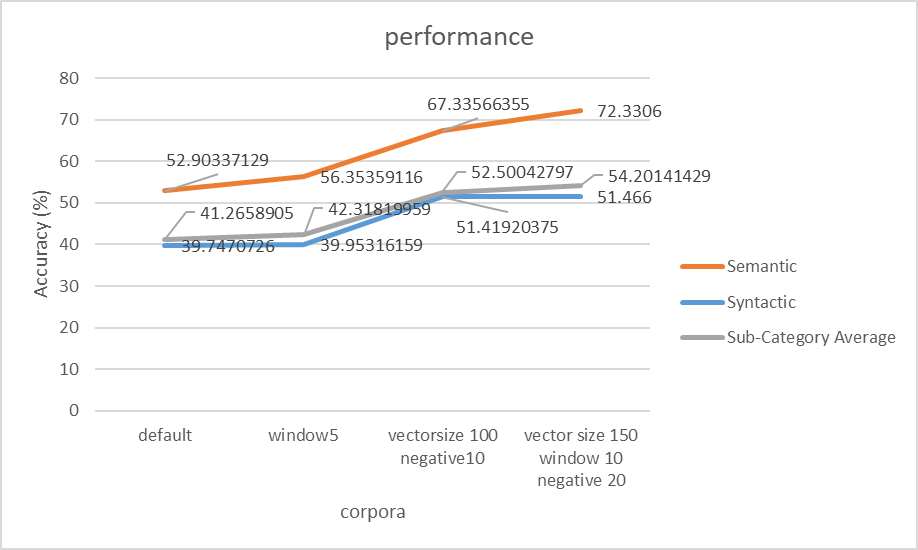




由圖中可以發現fasttext的整體平均表現較好，由於他的穩定性，並不會讓一些沒出現過的詞彙(尤其是形容詞副詞)OOV出現準確率特別低的情形，但就如果這些與廖有過專有名詞的統一像是wiki 20% text，W2V的表現可能就會比fasttext好，可以由Semantic欄位看出來。

1. 這裡有參考上述W2V之參數特性，去調整參數去看Accuracy有無顯著提升， 以下是各類名稱的參數





由上圖可以發現，提高window、negative、vector size大小都有提升，增加前後文字數量、提高負樣本數去增加關聯性、提高運算向量維度理論上accuracy會有提升，實驗符合結果。只是syntactic的準確率還是較低，主要歸因於W2V會對陌生的字詞較預測不準確以及wiki的語料中形容詞副詞這些較不豐富以至於訓練的模型不準確。Plot t-SNE for sub-categories : family

一張含有 文字, 螢幕擷取畫面, 字型, 圖表 的圖片

AI 產生的內容可能不正確。