使用 **Gensim** 訓練中文詞向量（**FastText**）

參考及引用資料來源

- [1] zake7749-使用 gensim 訓練中文詞向量 (http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)
- [2] gensim/corpora/wikicorpus (https://radimrehurek.com/gensim/corpora/wikicorpus.html)
- Word2Vec的簡易教學與參數調整指南 (https://www.kaggle.com/jerrykuo7727/word2vec)
- zhconv (https://pypi.org/project/zhconv/)
- jieba (https://pypi.org/project/jieba/)

In [1]:
```
%load_ext memory_profiler
```

確認相關 Packages

In [2]:
```
import os

# Packages
import gensim
import jieba
import zhconv
from gensim.corpora import WikiCorpus
from datetime import datetime as dt
from typing import List


if not os.path.isfile('dict.txt.big'):
    !wget https://github.com/fxsjy/jieba/raw/master/extra_dict/dict.txt.big
jieba.set_dictionary('dict.txt.big')

print("gensim", gensim.__version__)
```

```
gensim 4.3.1
jieba 0.42.1
```

準備中文訓練文本

訓練文本來源: **維基百科資料庫 (https://zh.wikipedia.org/wiki/Wikipedia:%E6%95%B0%E6%8D%AE%E5%BA%93%E4%B8%8B%E8%BD%BD)**

> 要訓練詞向量，第一步當然是取得資料集。由於 word2vec 是基於非監督式學習，訓練集一定一定要越大越好，語料涵蓋的越全面，訓練出來的結果也會越漂亮。
> [1] (http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)

- zhwiki-20210101-pages-articles.xml.bz2 (https://dumps.wikimedia.org/zhwiki/20210101/zhwiki-20210101-pages-articles.xml.bz2) (1.9 GB)

```
wget "https://dumps.wikimedia.org/zhwiki/20210101/zhwiki-20210101-pages-articles.xml.bz2"
```

目前已經使用另一份 Notebook (維基百科中文語料庫 zhWiki_20210101 (https://www.kaggle.com/bbqlp33/zhwiki-20210101)) 下載好中文維基百科語料，並可以直接引用

In [3]:
```
ZhWiki = r"C:\Users\user\Downloads\zhwiki-20230501-pages-articles-multistream.xml.bz2"

# !dir -sh $ZhWiki
# !CertUtil $ZhWiki
```

中文文本前處理

在正式訓練 Word2Vec 之前，其實涉及了文本的前處理，本篇的處理包括如下三點 (而實務上對應的不同使用情境，可能會有不同的前處理流程):

- 簡轉繁: zhconv (https://pypi.org/project/zhconv/)
- 中文斷詞: jieba (https://pypi.org/project/jieba/)
- 停用詞

簡繁轉換

wiki 文本其實摻雜了簡體與繁體中文，比如「数学」與「數學」，這會被 word2vec 當成兩個不同的詞。[1] (http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)
所以我們在斷詞前，需要加上簡繁轉換的手續

以下範例使用了較輕量的 Package zhconv (https://pypi.org/project/zhconv/)，
若需要更高的精準度，則可以參考 OpenCC (https://github.com/BYVoid/OpenCC)

In [4]:

Out[4]:
```
'這原本是一段簡體中文'
```

中文斷詞

使用 jieba (https://pypi.org/project/jieba/) `jieba.cut` 來進行中文斷詞，
並簡單介紹 jieba 的兩種分詞模式:

- `cut_all=False` 精確模式，試圖將句子最精確地切開，適合文本分析；
- `cut_all=True` 全模式，把句子中所有的可以成詞的詞語都掃描出來, 速度非常快，但是不能解決歧義；

In [5]:
```
seg_list = jieba.cut("我来到北京清华大学", cut_all=True)
print("Full Mode: " + "/ ".join(seg_list))  # 全模式

seg_list = jieba.cut("我来到北京清华大学", cut_all=False)
```

```
Building prefix dict from C:\Users\user\Downloads\dict.txt.big ...
Loading model from cache C:\Users\user\AppData\Local\Temp\jieba.u7157396f9872b4f171d0
922602b24c50.cache
Loading model cost 2.832 seconds.
Prefix dict has been built successfully.
```

```
Full Mode: 我/ 来到/ 北京/ 清华/ 清华大学/ 华大/ 大学
Default Mode: 我/ 来到/ 北京/ 清华大学
```

In [6]:

```
['中', '英', '夾雜', '的', 'example', '，', 'Word2Vec', '應該', '很', 'interesting',
'吧', '?']
```

引入停用詞表

停用詞就是像英文中的 **the,a,this**，中文的你我他，與其他詞相比顯得不怎麼重要，對文章主題也無關緊要的，
是否要使用停用詞表，其實還是要看你的應用，也有可能保留這些停用詞更能達到你的目標。[1] (http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)

- Is it compulsory to remove stop words with word2vec? (https://www.quora.com/Is-it-compulsory-to-remove-stop-words-with-word2vec)
- The Effect of Stopword Filtering prior to Word Embedding Training (https://stats.stackexchange.com/questions/201372/the-effect-of-stopword-filtering-prior-to-word-embedding-training)

以下範例還是示範引入停用詞表，而停用詞表網路上有各種各樣的資源
剛好 `kaggle` ，環境預設有裝 spacy (https://pypi.org/project/spacy/)，
就順道引用 `spacy` 提供的停用詞表吧 (實務上stopwords 應為另外準備好且檢視過的靜態文檔)

In [7]:
```
import spacy

# 下載語言模組
spacy.cli.download("zh_core_web_sm")  # 下載 spacy 中文模組
spacy.cli.download("en_core_web_sm")  # 下載 spacy 英文模組

nlp_zh = spacy.load("zh_core_web_sm") # 載入 spacy 中文模組
nlp_en = spacy.load("en_core_web_sm") # 載入 spacy 英文模組

# 印出前20個停用詞
print('--\n')
print(f"中文停用詞 Total={len(nlp_zh.Defaults.stop_words)}: {list(nlp_zh.Defaults.stop_w
print("--")
```

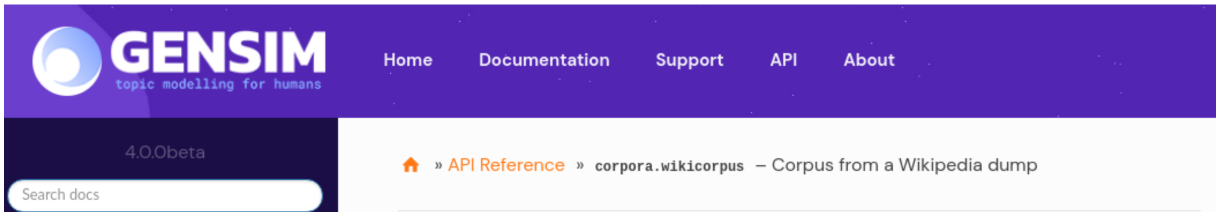
```
✔ Download and installation successful
You can now load the package via spacy.load('zh_core_web_sm')
✔ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
--

中文停用詞 Total=1891: ['有着', '替代', '最', '虽则', '从速', '及其', '即使', '基本上
', '———', '本着', '再其次', '宣布', '迄', '不下', '别管', '应该', '＜', '彻夜', '
针对', '那会儿'] ...
--
英文停用詞 Total=326: ['his', 'via', 'wherever', 'nobody', 'sometime', 'give', 'next
', 'seem', ' ‘s', "n't", 'hundred', 'but', 'whence', 'several', "'re", ' ‘d', 'such',
'n’ t', 'always', 'because'] ...
```

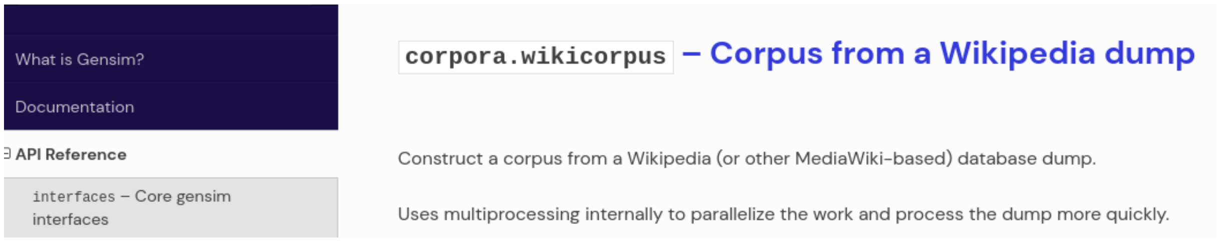
In [8]:
```
STOPWORDS =  nlp_zh.Defaults.stop_words | \
             nlp_en.Defaults.stop_words | \
             set(["\n", "\r\n", "\t", " ", ""])
print(len(STOPWORDS))

# 將簡體停用詞轉成繁體，擴充停用詞表
for word in STOPWORDS.copy():
    STOPWORDS.add(zhconv.convert(word, "zh-tw"))
```

```
2222
3005
```

讀取 wiki 語料庫，並且進行前處理和斷詞

維基百科 (`wiki.xml.bz2`)下載好後，先別急著解壓縮，因為這是一份 xml 文件，裏頭佈滿了各式各樣的標籤，我們得先想辦法送走這群不速之客，不過也別太擔心， `gensim` 早已看穿了一切，藉由調用 [wikiCorpus (https://radimrehurek.com/gensim/corpora/wikicorpus.html)](https://radimrehurek.com/gensim/corpora/wikicorpus.html)，我們能很輕鬆的只取出文章的標題和內容。[[1] (http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)](http://zake7749.github.io/2016/08/28/word2vec-with-gensim/)


GENSIM
topic modelling for humans
Home
Documentation
Support
API
About
4.0.0beta
Search docs
» API Reference » corpora.wikicorpus – Corpus from a Wikipedia dump


What is Gensim?

Documentation

API Reference

interfaces – Core gensim interfaces

`corpora.wikicorpus` – Corpus from a Wikipedia dump

Construct a corpus from a Wikipedia (or other MediaWiki-based) database dump.

Uses multiprocessing internally to parallelize the work and process the dump more quickly.

[2] (https://radimrehurek.com/gensim/corpora/wikicorpus.html)

Supported dump formats:

- `<LANG>wiki-<YYYYMMDD>-pages-articles.xml.bz2`
- `<LANG>wiki-latest-pages-articles.xml.bz2`

The documents are extracted on-the-fly, so that the whole (massive) dump can stay compressed on disk.

In [9]:
```
def preprocess_and_tokenize(
    text: str, token_min_len: int=1, token_max_len: int=15, lower: bool=True) -> List[s
    if lower:
        text  = text.lower()
    text = zhconv.convert(text, "zh-tw")
    return [
        token for token in jieba.cut(text, cut_all=False)
        if token_min_len <= len(token) <= token_max_len and \
            token not in STOPWORDS
    ]
```

In [10]:
```
print(preprocess_and_tokenize("歐幾里得，西元前三世紀的古希臘數學家，現在被認為是幾何之
print(preprocess_and_tokenize("我来到北京清华大学"))
```

```
['歐幾', '裡得', '西元前', '世紀', '古希臘', '數學家', '幾何', '父', '此畫', '拉斐爾
']
['來到', '北京', '清華大學']
['中', '英', '夾雜', 'example', 'word2vec', 'interesting']
```

In [12]:
```
%%time
%%memit
from utils import preprocess_and_tokenize
from typing import List


print(f"Parsing {ZhWiki}...")
```

```
Parsing C:\Users\user\Downloads\zhwiki-20230501-pages-articles-multistream.xml.bz2...
peak memory: 1925.79 MiB, increment: 555.05 MiB
Wall time: 2h 41min 5s
```

初始化 `WikiCorpus` 後，能藉由 `get_texts()` 可迭代每一篇文章，它所回傳的是一個 `tokens list` ，我以空白符將這些 `tokens` 串接起來，統一輸出到同一份文字檔裡。這邊要注意一件事， `get_texts()` 受 `article_min_tokens` 參數的限制，只會回傳內容長度大於 **50** (default) 的文章。

- **article_min_tokens** *(int, optional)* – Minimum tokens in article. Article will be ignored if number of tokens is less.

秀出前 3 偏文章的前10 個 token

In [13]:
```
g = wiki_corpus.get_texts()
print(next(g)[:10])
print(next(g)[:10])
print(next(g)[:10])


# print(jieba.lcut("".join(next(g))[:50]))
# print(jieba.lcut("".join(next(g))[:50]))
```

```
['歐幾裡', '西元前', '三世', '紀的', '古希臘', '數學家', '現在', '認為', '幾何', '之
父']
['蘇', '格拉', '底', '死', '雅克', '路易', '大衛', '所繪', '1787', '年']
['文學', '狹義上', '一種', '語言藝術', '語言', '文字', '為', '手段', '形象化', '客觀
']
```

將處理完的語料集存下來，供後續使用

In [14]:
```
WIKI_SEG_TXT = "wiki_seg.txt"

generator = wiki_corpus.get_texts()

with open(WIKI_SEG_TXT, "w", encoding='utf-8') as output:
    for texts_num, tokens in enumerate(generator):
        output.write(" ".join(tokens) + "\n")

        if (texts_num + 1) % 100000 == 0:
```

```
[2023-05-14 01:42:19] 已寫入 99999 篇斷詞文章
[2023-05-14 01:57:20] 已寫入 199999 篇斷詞文章
[2023-05-14 02:11:22] 已寫入 299999 篇斷詞文章
[2023-05-14 02:25:50] 已寫入 399999 篇斷詞文章
[2023-05-14 02:40:52] 已寫入 499999 篇斷詞文章
[2023-05-14 02:53:28] 已寫入 599999 篇斷詞文章
[2023-05-14 03:07:53] 已寫入 699999 篇斷詞文章
[2023-05-14 03:22:12] 已寫入 799999 篇斷詞文章
```

用**fastText**訓練 **Word2Vec**

In [21]:
```
from gensim.models import FastText
from gensim.models.word2vec import LineSentence

# from gensim.models import word2vec
import multiprocessing

max_cpu_counts = multiprocessing.cpu_count()
word_dim_size = 300  #  設定 word vector 維度
print(f"Use {max_cpu_counts} workers to train Word2Vec (dim={word_dim_size})")


# 讀取訓練語句
sentences = LineSentence(WIKI_SEG_TXT)

# 訓練模型
model = FastText(sentences, vector_size=word_dim_size, workers=max_cpu_counts)

# 儲存模型
output_model = f"fasttext.zh.{word_dim_size}.model"
```

```
Use 8 workers to train Word2Vec (dim=300)
```

儲存的模型總共會產生三份檔案

In [22]:

```
 磁碟區 C 中的磁碟是 OS
 磁碟區序號:  FC67-B6D0

 C:\Users\user\Downloads 的目錄

2023/05/12  下午 03:31        58,905,889 word2vec.zh.300.model
2023/05/12  下午 03:31     2,161,575,728 word2vec.zh.300.model.syn1neg.npy
2023/05/12  下午 03:30     2,161,575,728 word2vec.zh.300.model.wv.vectors.npy
               3 個檔案   4,382,057,345 位元組
               0 個目錄  15,767,293,952 位元組可用
```

In [23]:

```
 磁碟區 C 中的磁碟是 OS
 磁碟區序號:  FC67-B6D0

 C:\Users\user\Downloads 的目錄

2023/05/12  下午 03:31        58,905,889 word2vec.zh.300.model
2023/05/12  下午 03:31     2,161,575,728 word2vec.zh.300.model.syn1neg.npy
2023/05/12  下午 03:30     2,161,575,728 word2vec.zh.300.model.wv.vectors.npy
               3 個檔案   4,382,057,345 位元組

     檔案數目總計:
               3 個檔案   4,382,057,345 位元組
               0 個目錄  15,767,293,952 位元組可用
```

查看模型以及詞向量實驗

模型其實就是巨大的 Embedding Matrix

In [24]:
```
print(model.wv.vectors.shape)
```

```
(1801313, 300)
```

Out[24]:
```
array([[ 6.2352371e-01, -3.8689117e+00, -6.0808105e+00, ...,
         -7.0078617e-01,  5.2351685e+00, -4.3665843e+00],
       [ 3.7192154e+00,  2.7089843e-01, -3.7767277e+00, ...,
         -3.3715243e+00,  8.0675209e-01, -6.0544310e+00],
       [ 2.7178154e+00,  2.9948077e+00,  3.5245645e+00, ...,
         -3.3415356e+00,  6.1227312e+00, -6.3680973e+00],
       ...,
       [-8.4712386e-02,  1.3723016e-01, -9.8375186e-02, ...,
          3.0004183e-02, -1.2141043e-01,  4.6550050e-02],
       [ 4.9417309e-02,  6.7802534e-02,  1.7388929e-03, ...,
          3.6799662e-02, -1.7148748e-01,  1.1672581e-01],
       [-2.2792663e-01,  4.8828250e-01,  7.1294051e-01, ...,
         -3.3116922e-01, -5.4380304e-01, -6.7074078e-01]], dtype=float32)
```

收錄的詞彙

In [25]:
```
print(f"總共收錄了 {len(model.wv.index_to_key)} 個詞彙")

print("印出 20 個收錄詞彙:")
```

```
總共收錄了 1801313 個詞彙
印出 20 個收錄詞彙:
['年', '月', '日', '於', '為', '「', '與', '後', '臺', '中']
```

詞彙的向量

In [26]:
```
vec = model.wv['數學家']
print(vec.shape)
```

```
(300,)
```

Out[26]:
```
array([-2.13296390e+00,  2.08578706e+00, -1.80419767e+00,  1.08284962e+00,
       -8.70887160e-01,  3.55728656e-01, -1.43275723e-01, -1.93990707e+00,
       -1.16941081e-02,  1.44137096e+00, -3.96140963e-01,  7.31057644e-01,
       -2.05072045e+00,  1.53199124e+00, -8.28381360e-01, -2.64371801e-02,
        2.13499650e-01, -2.43354750e+00, -3.98861080e-01, -4.30927187e-01,
       -1.37037468e+00,  1.58980274e+00,  2.12647748e+00,  4.97391433e-01,
       -9.67977107e-01, -4.81504411e-01, -5.98746598e-01,  2.33286530e-01,
       -1.55410171e+00, -1.35444984e-01, -1.57270205e+00, -4.89924476e-02,
        8.16635311e-01, -2.41276550e+00, -4.05996293e-01, -2.55686450e+00,
        1.32473099e+00, -1.71091899e-01, -9.56757724e-01,  1.44891426e-01,
        1.54867721e+00, -7.99544692e-01,  2.28232145e-01,  8.94926727e-01,
        1.46105897e+00, -3.22066617e+00,  1.67939293e+00,  1.86104202e+00,
        6.42469049e-01, -2.16569155e-01,  4.40958440e-01, -1.17209435e+00,
       -1.66260278e+00, -6.09396636e-01,  5.00597656e-01, -1.09133685e+00,
        9.99951661e-01,  1.60997462e+00, -1.37811625e+00,  1.25027049e+00,
       -4.78484452e-01, -1.63081884e-01,  4.25951817e-04,  1.32616448e+00,
       -1.25643086e+00,  9.97701943e-01,  8.79492760e-02, -1.06327426e+00,
       -7.80789793e-01,  1.67294547e-01,  3.42821889e-02,  1.69165984e-01,
        2.53740162e-01,  2.91324043e+00,  3.09107095e-01, -1.48934948e+00,
       -3.58813435e-01,  2.44099110e-01,  1.48760104e+00,  1.60314584e+00,
       -1.58972001e+00,  1.35291290e+00,  1.82232285e+00,  4.30323660e-01,
       -1.78951132e+00, -4.77870971e-01, -1.91617835e+00, -2.58418012e+00,
        2.06151992e-01,  7.54278481e-01, -6.52611375e-01,  5.28277278e-01,
        8.20131183e-01,  5.48606813e-01,  1.05164802e+00, -1.41858411e+00,
       -2.95306277e-02, -3.93581055e-02, -3.22050124e-01,  5.37846863e-01,
        7.34474719e-01,  3.17440057e+00, -1.54985964e+00,  1.11193109e+00,
        3.77977538e+00, -1.66261077e+00, -6.69628084e-01,  3.47810411e+00,
        3.46134633e-01,  8.91721308e-01, -3.69780734e-02,  2.23749757e+00,
       -7.65153050e-01, -1.02997637e+00,  4.59361106e-01,  6.88673675e-01,
       -1.66743028e+00, -1.41925216e-01, -3.86390947e-02,  3.12792689e-01,
        7.25140929e-01, -7.37192482e-02,  1.30665809e-01,  1.49136797e-01,
        2.27321282e-01,  1.41856563e+00, -7.53293708e-02,  2.93569183e+00,
        1.24674058e+00, -7.23709643e-01, -5.13615906e-01, -2.17337370e+00,
        3.02701807e+00,  1.78629708e+00, -7.85275400e-01, -1.81608927e+00,
        5.54832876e-01, -1.54185331e+00,  2.44909239e+00, -1.23700869e+00,
        2.42501497e-01, -1.35701478e+00,  3.17305136e+00, -4.76702362e-01,
        5.19522727e-01, -1.67409444e+00, -1.90329742e+00,  1.44119227e+00,
        1.29305089e+00,  2.01229811e-01, -7.28152156e-01,  1.44406581e+00,
        6.93081856e-01,  2.39352679e+00,  7.33271122e-01,  7.33166993e-01,
       -1.50712416e-01,  7.61871278e-01,  9.97782886e-01, -7.98690856e-01,
       -3.40202165e+00,  3.96955442e+00,  1.16277313e+00,  1.86032021e+00,
        4.63948995e-01, -1.24318206e+00,  5.61743855e-01, -1.54467940e+00,
       -1.13721442e+00, -1.60597587e+00, -6.99905336e-01,  3.96340013e+00,
        2.17634821e+00, -7.98209071e-01,  1.66025683e-01, -1.36680305e+00,
        3.08071703e-01,  1.15431082e+00, -1.29435027e+00,  5.79328954e-01,
        1.79577518e+00,  1.90403044e+00,  1.18157601e+00,  3.37088294e-02,
        1.06327546e+00,  2.31122211e-01,  5.42936981e-01,  2.48109952e-01,
       -4.32565957e-01,  2.82973409e+00,  8.61261368e-01,  1.99071205e+00,
        2.21708417e+00,  1.82235742e+00,  6.42227590e-01,  1.98083413e+00,
       -2.25271201e+00,  2.06930375e+00,  8.28914642e-01, -4.20850329e-02,
        5.83185852e-01, -2.41519427e+00,  8.00874174e-01, -5.61632449e-03,
       -1.33718324e+00, -1.60220742e+00, -7.54322946e-01,  3.39590454e+00,
        1.34911454e+00,  5.49853519e-02,  2.78543353e+00, -8.42691511e-02,
        1.95571637e+00,  9.58757937e-01, -3.66281085e-02,  6.79617047e-01,
       -1.11788964e+00,  1.58858275e+00, -1.25660968e+00,  1.35547578e+00,
       -1.01983738e+00,  2.39330977e-01,  1.59593308e+00,  5.18609107e-01,
        2.78481102e+00, -1.53461611e+00,  2.43535614e+00, -7.32181013e-01,
        5.98852813e-01, -1.32738125e+00, -6.00022614e-01,  6.88536942e-01,
        6.13136649e-01, -1.01521957e+00,  6.77859962e-01, -1.88362396e+00,
       -1.08597863e+00,  6.94389224e-01, -2.47411394e+00,  1.31410849e+00,
        6.96344316e-01,  1.80009019e+00,  1.78651583e+00,  6.42327964e-01,
       -2.87031674e+00,  2.61717224e+00,  2.16547757e-01,  6.93990707e-01,
        1.23233214e-01, -2.82735914e-01,  1.00293612e+00,  6.97875559e-01,
        2.26708841e+00,  2.52055079e-01,  1.52647150e+00,  9.08343315e-01,
       -4.99709435e-02,  1.84688434e-01,  3.89294654e-01,  1.68524146e+00,
       -2.13796949e+00,  7.64285505e-01,  6.05083764e-01, -1.69081485e+00,
        2.02794957e+00,  2.86124170e-01,  6.14189744e-01, -2.31410265e+00,
       -2.53638339e+00, -3.69094700e-01, -3.00958157e-01, -4.15575564e-01,
        2.55126178e-01,  8.99742186e-01,  2.72460032e+00,  1.79514334e-01,
        2.52699542e+00,  2.57628024e-01,  1.40686321e+00,  7.14187026e-01,
        4.06636775e-01, -2.22442985e+00, -8.29771627e-03,  1.81991950e-01,
       -2.80413777e-01, -1.15925848e+00, -8.36147726e-01,  1.66928375e+00,
       -1.92164695e+00, -6.22458398e-01,  8.30292821e-01, -1.20590055e+00,
       -1.31618881e+00,  2.17316198e+00, -1.11067808e+00,  1.60193825e+00,
        2.40693346e-01,  7.00491369e-01,  1.33566403e+00, -1.48172510e+00],
      dtype=float32)
```

沒見過的詞彙

In [27]:
```python
word = "這肯定沒見過 "

# 若強行取值會報錯
try:
    vec = model.wv[word]
except KeyError as e:
```

查看前 **10** 名相似詞

`model.wv.most_similar` 的 `topn` 預設為 10

In [28]:

Out[28]:
```
[('精飲料', 0.970079243183136),
 ('輝劍', 0.9642638564109802),
 ('名松', 0.952566385269165),
 ('飲料則', 0.9492932558059692),
 ('飲料類', 0.9465901851654053),
 ('種飲料', 0.9463971853256226),
 ('飲料業', 0.9405738115310669),
 ('搖飲料', 0.9385449886322021),
 ('自飲料', 0.9377885460853577),
 ('軟飲料', 0.9219933152198792)]
```

In [29]:

Out[29]:
```
[('hcar', 0.8604835271835327),
 ('carcar', 0.8536281585693359),
 ('ccar', 0.8352298736572266),
 ('jetcar', 0.8216733336448669),
 ('boxcar', 0.8202365636825562),
 ('cars', 0.8176816701889038),
 ('tramcar', 0.8024346828460693),
 ('necar', 0.8009129166603088),
 ('zipcar', 0.8004717826843262),
 ('ucar', 0.7999507784843445)]
```

In [30]:

Out[30]:
```
[('youtubefacebook', 0.930781364440918),
 ('thefacebook', 0.9007937908172607),
 ('facebookpage', 0.8927770256996155),
 ('facebox', 0.8635411262512207),
 ('instagram', 0.8216885924339294),
 ('twitteryoutube', 0.7802010178565979),
 ('twitter', 0.7719414234161377),
 ('googleyoutube', 0.7637636661529541),
 ('lnstagram', 0.751848042011261),
 ('youtube', 0.750074565410614)]
```

In [31]:

Out[31]:
```
[('因欺騙', 0.8916229605674744),
 ('集欺騙', 0.8886957764625549),
 ('欺騙過', 0.8868732452392578),
 ('還欺騙', 0.8852323293685913),
 ('並欺騙', 0.8733299970626831),
 ('中莉', 0.8651049137115479),
 ('欺騙者', 0.843005359172821),
 ('破沙苑', 0.834650993347168),
 ('欺騙性', 0.7747063636779785),
 ('給欺騙', 0.7421086430549622)]
```

In [32]:

Out[32]:
```
[('合約爭', 0.9549153447151184),
 ('僱合約', 0.9542264938354492),
 ('商合約', 0.9533745646476746),
 ('止合約', 0.9533279538154602),
 ('合約機', 0.953099250793457),
 ('職合約', 0.9529709815979004),
 ('指合約', 0.9516268372535706),
 ('價合約', 0.9515151977539062),
 ('員合約', 0.9512284994125366),
 ('應合約', 0.9511519074440002)]
```

計算 **Cosine** 相似度

In [33]:

Out[33]:
```
0.40054494
```

In [34]:

Out[34]:
```
-0.04506902
```

讀取模型

In [37]:
```
print(f"Loading {output_model}...")
```

```
Loading fasttext.zh.300.model...
```

In [38]:

Out[38]:
```
True
```

In [ ]: