自然語言處理
分詞與表示

Instructor: 馬豪尚

什麼是一個詞?

- 字符(character):
  - 獨立的字符就是字符表裡能找到的字
  - 例如:蜻、蜓、T 、H 、I 、S
- 詞(word):
  - 句子裡最小具有獨立意義的單位
  - 例如:蜻蜓、葡萄、this 、cat

英文:

Lemma: cat = cats

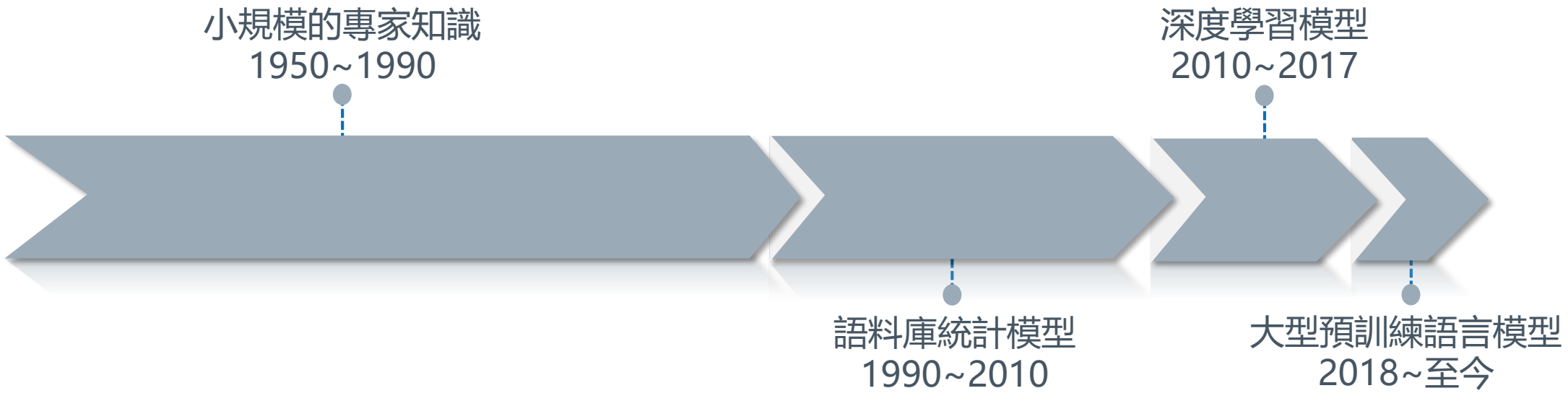
Wordform: cat != cats

中文:

葡萄、蜻蜓、蚯蚓、車門

自然語言的表示

- 語意在電腦內部是如何表示的？
- 根據表示方法的不同，自然語言處理經歷了四個時代變遷

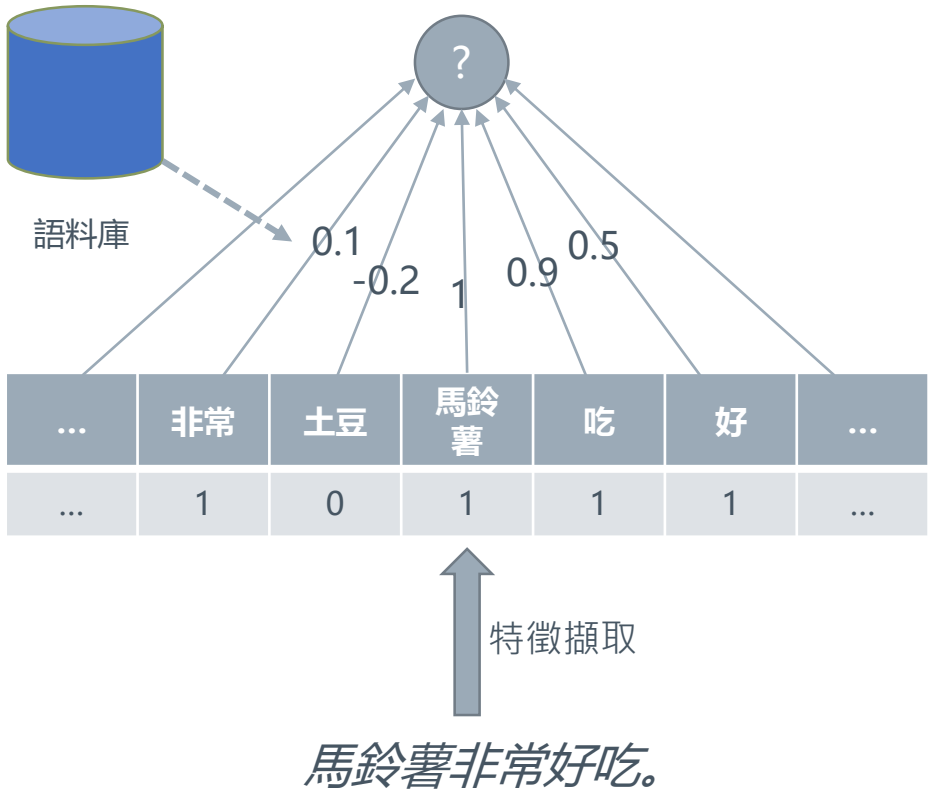


基於符號（字串）表示的專家知識

- 「馬鈴薯非常好吃。」的情感傾向性？
- 如果：出現褒義詞「好」「喜歡」等
  - 結果為正向情緒(褒義)
- 如果：出現"不"
  - 則結果傾向性為負面

- 優點
  - 符合人類的直覺
  - 可解釋、可干預性好
- 缺點
  - 知識不夠完備
  - 需要專家建構與維護
  - 不便於計算

基於向量表示的統計模型



- 獨熱編碼(one-hot encoding)使用高維度、離散、稀疏的向量表示詞
- 維度為詞列表大小，表示法中所有維度只有一位為1，其餘為0
  - 馬鈴薯：[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, …]
  - 好：[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, …]
  - 吃：[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, …]

缺點

- 嚴重的資料稀疏問題
- 無法處理「多詞一義」的現象

詞的分佈語意假設

- 分佈語意假設（Distributional semantic hypothesis）
- 詞的意思可由其上下文詞的分佈來表示
  - You shall know a word by the company it keeps -- Firth J.R. 1957

詞的分佈（Distributional）表示

› 分佈詞向量

	shinning	bright	trees	dark	look
moon	38	45	2	27	12

he curtains open and the moon shining in on the barely
ars and the cold , close moon " . And neither of the w
rough the night with the moon shining so brightly , it
made in the light of the moon . It all boils down , wr
surely under a crescent moon , thrilled by ice-white
sun , the seasons of the moon ? Home , alone , Jay pla
m is dazzling snow , the moon has risen full and cold
un and the temple of the moon , driving out of the hug
in the dark and now the moon rises , full and amber a
bird on the shape of the moon over the trees in front
But I could n't see the moon or the stars , only the
rning , with a sliver of moon hanging among the stars
they love the sun , the moon and the stars . None of
the light of an enormous moon . The plash of flowing w
man 's first step on the moon ; various exhibits , aer
the inevitable piece of moon rock . Housing The Airsh
oud obscured part of the moon . The Allied guns behind

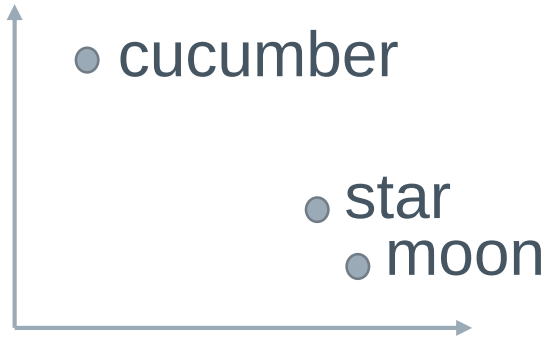
語言模型

- 語言模型（Language Model，LM）
  - 描述一段自然語言的概率或給定上文時下一個詞出現的概率
    - $P(w_1, \cdots w_l)$，$P(w_{l+1}|w_1, \cdots w_l)$
  - 以上兩種定義等價

$$P(w_1 w_2 \cdots w_l) = P(w_1)P(w_2|w_1)P(w_3|w_1 w_2)\cdots P(w_l|w_1 w_2 \cdots w_{l-1})$$

$$= \prod_{i=1}^{l} P(w_i|w_{1:i-1})$$

- 語意相似度透過計算向量相似度獲得



斷詞/分詞

- 詞（Word）
  - 最小具有獨立意義的單位
- 中文分詞是將中文字序列切分成一個個單獨的詞
- 例如以下句子:「我的興趣是看電影和讀書」
  - 「我的興\趣是看電\影和讀\書」像這樣分組是不符合現實世界的意義
  - 「我\的\興趣\是\看\電影\和\讀書」透過斷詞技術就可以取得這樣的詞彙。

斷詞/分詞演算法

- **基於設定好的單位切分**：將整個字符串以固定單位來切分，例如N-Gram
- **基於詞典的分詞法**：將待匹配的字符串和一個已建立好的詞典中的詞進行匹配，通常會採用雙向匹配的方法，但這方法的能力有限，例如像是新發明的詞就無法進行匹配
- **統計的機器學習算法**：如HMM，CRF (Conditional Random Field)，常見中文斷詞Jieba套件，對於不存在於字典的字詞就是用統計的方法來處理的

N-Gram 斷詞模型

- N-gram 模型是一種基於統計機率的自然語言處理模型，用於對文本進行建模和預測。它基於一個簡單的假設，即在一個句子或文本中，下一個詞的出現只與前面的N-1 個詞有關，與整個文本的上下文無關。
- N-gram 模型將文本拆分為一系列的N 個詞的序列，這些序列被稱為N-gram。
- 假設N=2，文本為“ChatGPT is a language model”，則會被拆分成:(ChatGPT, is) (is,a) (a,language) (language,model)

N-Gram 斷詞模型

- **我的興趣是看電影和讀書**
- Uni-Gram
  - 「我\的\興\趣\是\看\電\影\和\讀\書」
- Bi-Gram
  - 「我的\的興\興趣\趣是\是看\看電\電影\影和\和讀\讀書」
- Tri-Gram
  - 「我的興\的興趣\興趣是\趣是看\是看電\看電影\電影和\影和讀\和讀書」

N-Gram 斷詞模型

› Bi-Gram

$$P(w_n|w_{n-1}) = \frac{count(w_{n-1}w_n)}{count(w_{n-1})}$$

› Tri-Gram

$$P(w_n|w_{n-2}, w_{n-1}) = \frac{count(w_{n-2}w_{n-1}w_n)}{count(w_{n-2}w_{n-1})}$$

基於詞典的分詞法

- **我的興趣是看電影和讀書**
  - 「我\的\興趣\是\看電影\和\讀\書」

詞典
我
興趣
電影
讀
看
書
看電影

中文斷詞

- 常遇到的難題
  - 歧異詞
    - 「我們 / 在野 / 生動 / 物 / 園 / 玩」
    - 「我們 / 在 / 野生 / 動物園 / 玩」
  - 新詞識別
    - 特有名詞(ptt梗、溫拿)
    - 人名地名

中文斷詞

- 正向最大批配法
  - 會有一個詞典，將句子在詞典中由前向後比對，一一比對最長詞的匹配結果
  - 「我們 / 在野 / 生動 / 物 / 園 / 玩」
- 逆向最大批配法
  - 會有一個詞典，將句子在詞典中由後向前比對，一一比對最長詞的匹配結果
  - 「我們 / 在 / 野生 / 動物園 / 玩」
- 正反都做完之後取最長詞
  - 「我們 / 在 / 野生 / 動物園 / 玩」
  - 「我們 / 在野 / 生 / 動物園 / 玩」

中文斷詞

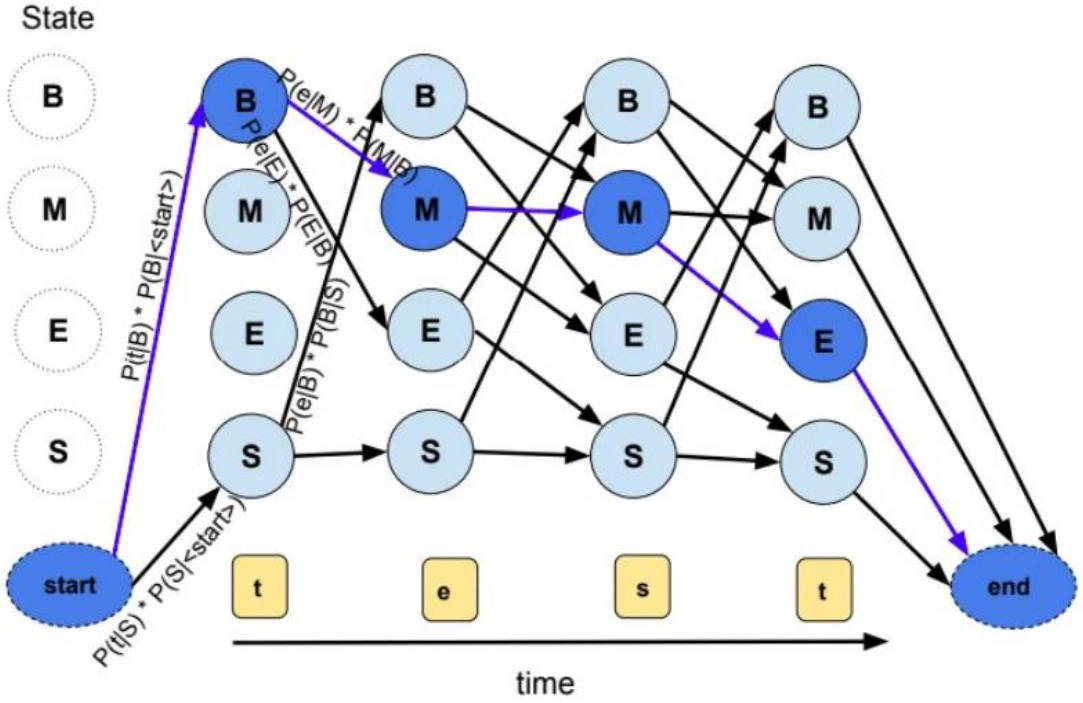
- 基於詞典的斷詞方式，只要詞典中沒有收錄句子中的詞，那可能效果會非常差
- 大部份比較好的斷詞系統都是使用全切分方法，切分出與詞庫匹配的所有可能，然後再運用統計模型決定最好的切分結果

統計的機器學習算法

› HMM(隱馬可夫模型Hidden Markov Model)
  – 運用大量文本去統計"詞與詞"
    之間的關聯性來模擬機率

B→Begin(詞開頭)
M→Middle(詞中間)
E→End(詞結尾)
S→Single(單獨)

在野生動物園→SBEBME



斷詞/分詞

- 以英語為代表的印歐語系語言，是否需要進行分詞？
- 這些語言詞形變化複雜如：computer、computers、computing等
  - 僅用空格切分的問題
    - 數據稀疏詞表過大，降低處理速度
- 子詞切分
  - 將一個單詞切分為若干連續的片段（子詞）方法眾多，基本原理相似
    - 使用盡量長且出現頻率高的子詞對單詞進行切分

BPE子詞切分演算法(Byte Pair Encoding)

› 建立子詞表 {“old”: 7, “older”: 3, “finest”: 9, “lowest”: 4}

Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	16
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13
11	t	13
12	w	4

BPE子詞切分演算法(Byte Pair Encoding)

› 合併es

Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	16-13=3
6	r	3
7	f	9
8	i	9
9	n	9
~~10~~	~~s~~	~~13-13~~
11	t	13
12	w	4
13	es	13

BPE子詞切分演算法(Byte Pair Encoding)

› 合併est

Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	3
6	r	3
7	f	9
8	i	9
9	n	9
~~10~~	~~t~~	~~13-13~~
11	w	4
~~12~~	~~es~~	~~13-13~~
13	est	13

BPE子詞切分演算法(Byte Pair Encoding)

› 合併est</w>

Number	Token	Frequency
1	</w>	23-13
2	o	14
3	l	14
4	d	10
5	e	3
6	r	3
7	f	9
8	i	9
9	n	9
10	w	4
~~11~~	~~est~~	~~13-13~~
12	est</w>	13

BPE子詞切分演算法(Byte Pair Encoding)

› 合併ol

Number	Token	Frequency
1	</w>	10
2	o	14-10=4
3	l	14-10=4
4	d	10
5	e	3
6	r	3
7	f	9
8	i	9
9	n	9
10	w	4
11	est</w>	13
12	ol	14

BPE子詞切分演算法(Byte Pair Encoding)

› 合併old

Number	Token	Frequency
1	</w>	10
2	o	4
3	l	4
~~4~~	~~d~~	~~10-10~~
5	e	3
6	r	3
7	f	9
8	i	9
9	n	9
10	w	4
11	est</w>	13
~~12~~	~~ol~~	~~10-10~~
13	old	10

BPE子詞切分演算法(Byte Pair Encoding)

› 子詞詞表

Number	Token	Frequency
1	</w>	10
2	o	4
3	l	4
4	e	3
5	r	3
6	f	9
7	i	9
8	n	9
9	w	4
10	est</w>	13
11	old	10

BPE子詞切分演算法(Byte Pair Encoding)

- 將子詞詞表依照子詞的長度由大到小進行排序
- 從前向後遍歷子詞詞表，依序判斷一個子詞
- 是否為單字的子字串如果是則將該單字切分，然後繼續向後遍歷子詞詞表
- 如果子詞詞表全部遍歷結束，單詞中仍然有子串沒有被切分，那麼這些子串一定為低頻串，則使用統一的標記，如'<UNK>'進行替換

BPE子詞切分演算法(Byte Pair Encoding)

- the oldest man in USA
- [“the</w>”, “old”, “est</w>”, “man</w>”, “in</w>”, “USA</w>”]

→[1, 2, 5, 3, 4, <unk>]

Number	Token
1	the</w>
2	old
3	man</w>
4	in</w>
5	est</w>

NLTK (Natural Language Tool Kit)

- import nltk
- nltk.download('punkt')
- 斷詞
  - nltk.word_tokenize(string)
- 斷句
  - nltk.sent_tokenize(paragragh)

NLTK正規表達式客製斷詞

- from nltk.tokenize import RegexpTokenizer
- RegexpTokenizer() 的參數跟用法：
  - 第一個參數會是你希望它留下來的東西。也就是說，你要告訴它每次遇到"非什麼條件的東西"就要停下來分割字串
  - 第二個參數gaps告訴它我們需不需要保留第一個參數指涉的東西，也就是遇到的分割符號(false: 不需要保留)

NLTK正規表達式客製斷詞

- 自定義規則去除標點符號
  - tokenizer = RegexpTokenizer(r'\w+', gaps = False)
  - clean_sent = tokenizer.tokenize(string)
- 保留不是標點符號的符號I’m
  - tokenizer2 = RegexpTokenizer(r'\w+|\'\w+', gaps = False)
  - clean_sent2 = tokenizer2.tokenize(string)
- 保留切割用的字符串
  - tokenizer3 = RegexpTokenizer(r'\w+|\'\w+', gaps = True)
  - clean_sent3 = tokenizer3.tokenize(string)

NLTK正規表達式客製斷詞

- 去除stopword
  - from nltk.corpus import stopwords
  - nltk.download('stopwords')
  - stopword = stopwords.words('english')

文字正規化

- 詞幹提取(Stemming)
  - 詞幹提取是去除詞綴得到詞根的過程
  - 較偏向基於規則(rule-based)的方式去拆解單詞
  - 後綴去除法
    - 如果詞的結尾是「ed」,則去掉「ed」
    - 如果詞的結尾是「ing」,則去掉「ing」
    - 如果詞的結尾是「ly」,則去掉「ly」

文字正規化

- 詞形還原(Lemmatization)
  - 首先確定詞彙的發音部分，然後根據發音的部分確定詞彙的根，停頓詞規則隨著單詞的發聲部分的改變而改變
  - 動詞形式、形容詞形式、名詞形式、名詞複數、過去分詞...將不同形式的字還原成同一個字
  - 降低文本詞彙的複雜度
- Example
  - play, played, playing, plays
  - baby, babies
  - feet, foot

文字正規化

- Stemming
  - 計算速度快
  - 容易over stemming
  - 在處理特殊的詞上效果較差(run/ran)
- Lemmatization
  - 相較於 Stemming 會更精準
  - 處理的時間較長

NLTK stemming

- 載入模組
  - from nltk.stem.porter import PorterStemmer
- 創建 PorterStemmer 物件
  - ps=PorterStemmer()
- 使用已創的物件去做詞幹提取
  - ps.stem(words)

NLTK lemmatizer

- 載入模組
  - from nltk.stem import WordNetLemmatizer
  - nltk.download('omw-1.4')
- 創建WordNetLemmatizer物件
  - wnl = WordNetLemmatizer()
- 使用已創的物件去做詞形還原
  - wnl.lemmatize(words)

NLTK詞頻統計

- 載入模組
  - from nltk.probability import FreqDist
- 建立字典
  - fdist = FreqDist(result)
- 詞頻統計函式
  - fdist.most_common()
  - fdist.most_common(number)

NLTK語料庫

- Gutenberg
  - 是第一個提供免費的網路電子書平台，根據官方網站說明，project gutenberg 已經有超過 57,000本免費的電子書，NLTK 的 package 僅納入部分語料
- Brown
  - brown 語料庫是第一個百萬等級的電子語料庫(英文)， 1961 年由 Brown University 所整理，這個語料庫包含的字詞來自 500 個資料源，並參考資料源的種類做分類，例如：adventure 、news、reviews...等。
- Reuters
  - reuters 是路透社語料庫，涵蓋 10,788 個新聞文本，共有 90 個分類，例如：housing、income、tea...等。
- Inaugural
  - inaugural 是歷屆美國總統就職演說的語料庫，文本的命名方式是『年份+人名』，共有 56 個文本，最新一筆收錄的是 2009 年 Obama 的演說稿。

NLTK語料庫

- 載入語料庫(以gutenberg為例)
  - from nltk.corpus import gutenberg
- 查找語料庫當中的文本 id
  - corpus.fileids()
- 原始內容、單詞列表、句子列表
  - corpus.raw(fileids)
  - corpus.words(fileids)
  - corpus.sents(fileids)
- 語料庫內文本的分類屬性
  - corpus.categories(fileids)

練習

- 基於NLTK套件建立一個英文文本詞頻統計程式
  - 下載NLTK語料庫內的任意一篇文章當作要處理的文件
  - 資料清理
    - 全部改成小寫
    - 去除非標點符號(但是要保留一般符號)
    - 去除stop words
  - 斷詞
  - 文字正規化
  - 詞頻統計並輸出