同位詞夾子:主題式分類詞庫萃取演算法

Appositional Term Clip: A Subject Oriented Appositional Term Extraction Algorithm

謝育平*

Yuh-Pyng Shieh

arping@gmail.com

* 銘傳大學資訊工程學系

Department of Computer Science and Information Engineering,

Ming Chuan University

摘要

在資訊檢索、自然語言處理、數位典藏等眾多領域之中，文字處理始終是研究者面臨的第一個課題。對中文資料處理來說，自動斷詞、命名實體萃取、詞彙自動分類是前置工作的重點，傳統研究著重於各項自動化工作的精準率與召回率。本文提出半自動主題式詞庫萃取演算法，命名為「同位詞夾子」，主要利用人工來保證精準率，利用機器速度來補足召回率，以達到極高的準確率與儘量高的召回率。

同類的詞彙具有很高的同位性；例如「台北」與「高雄」就有很高的同位性；所謂同位性係指在文件中所有出現「台北」的地方，幾乎都可以使用「高雄」來替代，且替代後文句仍是非常通順，所以我們稱「台北」與「高雄」互為同位詞。「同位詞夾子」是由五個部件所組成（前文、前綴、中綴、後綴、後文），主要描述一個詞彙在文件某處的特徵，用以在文件中萃取該詞彙的同位詞。

本文演算法事先要求使用者提供該類詞彙的種子範例，演算法利用種子範例在文件中掃描以產生該類詞彙的詞夾子，再利用產生的詞夾子在文件中掃描並夾出該類詞彙的候選詞，依照候選詞數值化後的「同位性分數」排序供使用者人工決定是否符合該分類；再依人工幫助擴充種子範例、重啟演算法，如此互動循環到滿意為止。

本文成功萃取台灣歷史數位圖書館中的人名、地名、官職名、事件名等，也成功在中國古典小說中萃取三國演義的武器名、西遊記的法術名、紅樓夢的衣飾名、金瓶梅的小吃名等非傳統命名實體研究的詞彙分類。平均而言，一個分類詞庫的萃取可以在兩個小時內完成。

關鍵字：同位詞、詞夾子、命名實體、詞彙萃取、文字探勘

Abstract

On Chinese text processing, automated term extraction, named entity recognition, and term classification are important. Traditional researches focus on recall and precision of these automated works. This paper provides a semi-automated subject-oriented term-extraction algorithm called "Appositional Term Clip". We utilize "human-aid" to ensure extremely high precision and utilize "machine-aid" to improve recall as higher as possible.

Terms in the same category have high appositional similarity, for example, Taipei and Kaohsiung. Appositional similarity describes that almost all occurrences of "Taipei" can be substituted by "Kaohsiung" with high readability and fluency. Taipei and Kaohsiung are called appositional terms. An appositional term clip comprises 5 parts （preamble, prefix, infix, suffix, postamble） to describe a feature of some term's occurrence.

Applying the algorithm, users are asked to provide some terms of some category as seeds. Appositional term clips are produced by scanning all occurrences of seeds. Candidate （ appositional ） terms are produced by scanning all occurrences of the produced clips. An ordered list of candidate terms is provided for users to check whether a candidate is in the category. According to the users' work, seeds are enriched and the algorithm is applied again. The semi-automated procedure is applied again and again until the result is satisfied.

We successfully extract some categories such as person names, places, official positions and events from Taiwan History Digital Library, and some special ones such as weapons, magic, clothing, and snacks from Chinese ancient archives. In average, a category can be completed in two hours depending on the size of corpus.

Keyword: Appositional Term, Term Clip, Named Entity, Term Extraction, Text Mining.

壹. 簡介

在資訊檢索、自然語言處理、數位典藏等眾多領域之中，文字處理始終是研究學者面臨的第一個課題。處理原始文字，有助於後續的文章搜尋，資料統計，文件導讀等工作。對中文資料處理來說，詞彙辨識（自動斷詞）、詞彙萃取（命名實體萃取）、詞彙分群、詞彙分類是前置工作的重點。

1.詞彙標識（自動斷詞）係指在文章中將詞彙組合情形標識（標誌及辨識）出來。自動斷詞主要分為兩種，一是線性斷詞，一是樹狀斷詞，比如說：「同位詞夾子」的線性斷詞應為「（同位詞）（夾子）」，而其樹狀斷詞則為「（（（同位）詞）（夾子））」。線性斷詞是詞彙的直線排列，可以從中閱讀散狀的語意。而樹狀斷詞可稱為句構分析，其代表詞句語意之正確性。不同的線性斷詞或樹狀斷詞會導致不同的語意，比如說「（（（同位）詞）（夾子））」是指「同位詞」的「夾子」；而「（（同位）（詞（夾子）））」是指「同位」的「詞夾子」；這兩種解釋在語意上是不相同的。在詞彙標識中，除了將詞句的結構分析出來以外，還可以為每一個樹狀子結構標識其在語言學上的屬性，例如名詞、形容詞、動詞等。

2.**詞彙萃取（命名實體萃取）**係指在文章中將詞彙萃取出來，在此可分兩種，一種是萃取出所有的詞彙但不分類；另一種是根據某主題，萃取出屬於該主題的詞彙。例如給定主題「地名」，要求在文庫中萃取所有地名。命名實體萃取大部分是指人名、地名、組織名等實體，是屬於主題導向的詞彙萃取。

3.**詞彙分群**係指在一個既定的詞彙集（可動態產生）中，利用關連性及分群演算法分成若干群， 幫助加工再利

用。一個可能的功用是將搜尋後結果的文章集及詞彙集自動分群、以詞彙集分群提示讀者分群瀏覽，或是靜態使用人工做文庫分群分類使用。

4.**詞彙分類**係指在一個既定的詞彙集（可動態產生）及暨有分類別中，利用關連性及分類演算法對詞彙進行分類。例如將詞彙「台北」分入類別「地名」中。此處類似詞典的建立，可應用的範圍相當地多。

以上**四個工作**是互相影響的，須看應用的領域來選擇處理的問題。

1. 「詞彙萃取」幫助「詞彙分類」：為了完成「詞彙分類」，可以針對暨有的不同類別進行「詞彙萃取」，然後將萃取結果與既有詞彙集比對來完成部份工作。
2. 「詞彙分群」幫助「詞彙分類」：在「詞彙分群」後，可以人工或自動的方式針對各個分群進行命名即可完成部份工作。
3. 「詞彙分類」幫助「詞彙萃取」：利用「詞彙分類」後的分類詞典與給定的主題進行相似性分析，然後利用詞典中的相似分類再文章的過濾進行過濾，即可完成部分工作。
4. 「詞彙分類」幫助「詞彙標識」:「詞彙分類」後的分類詞典有助於計算斷詞結構的機率及斷詞後的詞性標誌。
5. 「詞彙標識」幫助「詞彙萃取」：詞彙標識後的詞彙集可依詞性對萃取主題的關係，有助於排除或認同某詞彙是否符合該主題等等。

本文所要處理的主要問題是「主題式之詞彙萃取」並應用在「高精準率要求」的領域上，屬於以上四種應用之「詞彙萃取」。所謂「主題式詞彙萃取」係指在給定主題之下，從文庫中搜尋該主題分類下的詞彙，並在高精準率要求之下，儘量提高召回率。例如從文庫中找出「地名」，越多越

好，但不能錯認地名。

傳統研究著重於各項自動化工作的精準率與召回率。本文提出半自動主題式詞庫萃取演算法，命名為「同位詞夾子」，主要利用人工來保證精準率，利用機器速度來補足召回率，以達到極高的準確率與儘量高的召回率。同類的詞彙具有很高的「同位性」；例如「台北」與「高雄」就有很高的同位性；所謂「同位性」係指在文件中所有出現「台北」的地方，幾乎都可以使用「高雄」來替代，且替代後文句仍是非常通順；「台北」與「高雄」可以在相同的位置上替代，所以我們稱「台北」與「高雄」互為同位詞。「同位詞夾子」(簡稱「詞夾子」)是由五的部件所組成（前文、前綴、中綴、後綴、後文），主要描述一個詞彙在文件某處的特徵，用以在文件中萃取該詞彙的同位詞。

「同位詞夾子」演算法事先要求使用者提供該類詞彙的種子範例，演算法利用種子範例在文件中掃描以產生該類詞彙的同位詞夾子，再利用產生的詞夾子在文件中掃描並夾出該類詞彙的候選詞，依照候選詞數值化後的同位性排序供使用者人工決定是否符合該分類；再依人工幫助擴充種子範例、重啟演算法，如此互動循環到滿意為止。本文成功萃取台灣歷史數位圖書館（Taiwan History Digital Library）（http://thdl.ntu.edu.tw）中的人名、地名、官職名、事件名等，也成功在中國古典小說中萃取三國演義的武器名、西遊記的法術名、紅樓夢的衣飾名、金瓶梅的小吃名等非傳統命名實體研究的詞彙分類。平均而言，一個分類詞庫的萃取可以在兩個小時內完成。

「同位詞夾子」演算法的歷史源於 2005 年，項潔、謝育平、張尚斌合作萃取明清檔案中的分類詞彙所設計之演算法，並由謝育平命名為「詞夾子」，而後成為張尚斌之畢業

論文(張尚斌，2006)。後因在文章上未多著墨，導致詞夾子雖然廣為使用，卻極少人知道其源於作者。

本文將闡述詞夾子的精神、設計理念及操作技巧，並提供程式下載供學術研究。本文後續章節組織如下：第貳章進行現有技術的分類及探討、第參章講述同位詞夾子演算法，第肆章討論精準率與召回率、第伍章講述同位詞夾字在幾個計劃中的成果，第陸章為結論與未來發展，第柒章則是參考文獻。

貳. 現有技術探討

「命名實體萃取」（Named Entities Recognition），或是「詞彙萃取」（Terms Extraction）源自於 6th Message Understanding Conference （MUC-6），（Grishman & Sundheim 1996），屬於資訊萃取的一個子工程。萃取工作分為兩種，一種是純粹從文庫中萃取詞彙程一集合，另一種是在文庫中辨識詞彙並標示詞彙屬性。以 Mikheev et al.（1999）論文為例，他定義了五種預先定義好的類別, 分別是時間、數字、人物、組織, 以及區域。並對文庫中的文章進行標識工作。例如：

```
<ENAMEX TYPE='PERSON' >Flavel
Donne</ENAMEX> is an analyst with <ENAMEX
TYPE= ' ORGANIZATION ' >General Trends
</ENAMEX>, which has been based in <ENAMEX
TYPE='LOCATION'>Little Spring</ENAMEX> since
<TIMEX TYPE='DATE' >July 1998</TIMEX>.
```

對於命名實體萃取方法來說，在 Nadeau, D. & Sekine, S（2007）關於詞彙萃取的分類依照系統學習的方法分為三大類：**指導性學習**（Supervised Learning）、**半指導性學習**（Semi-supervised Learning）以及**不需指導性學習**（Unsupervised Learning）。

指導性學習：在學習階段給予系統大量的註解過的文庫、經過整理分析後建立出判別的規則來進行詞彙萃取。這類的方法包含下列幾種馬可尼可夫鍊（Hidden Markov Models）（D. Bikel et al. 1997）、決策樹（Decision Trees）（S. Sekine 1998）、最大熵值法 Maximum Entropy Models （A. Borthwick 1998）,支持向量機 （Support Vector Machines）（M. Asahara & Matsumoto 2003）, and 條件隨機域（Conditional Random Fields）（A. McCallum & Li 2003）。

半指導性學習：以一些種子出發，尋找種子詞彙在文庫中的規則，利用規則尋找同類詞彙，再將找到之詞彙擴充成新種子，再重複演算法擴充種子集以成詞彙萃取。依此類推，舉一個例子說明：Richard C. Wang and William W. Cohen（2007） 利用 html 的標籤（tag）在 Google 裡搜尋出排在 Google 前幾頁的高度搜尋然後分析出相同文法的同樣元素，例如本田汽車（Honda）作為種子，然後搜出本田汽車網頁內的位置利用相同的標籤去臆測出候選詞,
<tr>或是<td>然後找出相同位置之後選詞回到 Google 內繼續當種子查詢，最後利用 Random Walk 的方式將候選詞排出一個分數，以分數高低決定命名實體。另外如 Tianhao Wu & William M. Pottenger （2005）以及 S. Brin （1998）也採類似的方法作為命名實體的研究，這兩篇論文不約而同的利用正規表示法作為搜尋相似上下文的框架。例如 S.Brin 利用下面的正規表示法[A-Z][A-Za-z .,&]5,30[A-Za-z.]去搜尋和{Isaac Asimov, The Robots of Dawn}有同樣正規表示法的書名及作者。因此找出了 The Robots of Dawn, by Isaac Asimov

（Paperback）和 The Ants, by Bernard Werber （Paperback）等相似的命名實體。

不需指導性學習：是利用分群的方法達成，有些利用相似位置的詞彙作為分群的條件，另外有一些則用詞彙的資源如 WordNet 或是中研院平衡語料庫去分類出同一群的詞彙。例如簡立峰（1997），先將內文建立 PAT-tree，在輔以詞頻和規則將詞彙萃取出來。

本文所提「詞夾子演算法」是屬於半指導性學習的演算法。Nadeau, David （2007）對半指導性學習有更進一步的調查與分類。本文參考並補充而成以下討論。

在萃取出詞彙之後，詞彙的意義就很重要。可能面對一詞多義與多詞同義的問題。在去除命名實體的歧異性上，Mann & Yarowski （2003） 尋找人名所對應的正確資訊。在 Mann 和 Yarowski （2003）的論文中就用來將在自傳中的人物名字做統整和分析並且消除某些特定或是非特定的歧異。而諸如 Zoominfo（http://www.zoominfo.com）和 Intelius（http://search.intelius.com）兩個網站皆對網路上的人名作分析也是此類的應用。Einat Minkov et al. （2005）也針對在電子郵件中的人名進行判別和萃取判別和萃取。而在醫學名詞或是生物化學名詞的判別上也多有建樹，如 Seokhwan Kim et al. （2006）則以 Maximum Marginal Relevance 技術提出萃取生物訊息相關詞彙的論文。而在詞彙語義的多詞同義上， Nadeau & Turney （2005）進行同義詞的處理統做。例如「IBM」和「International Business Machines」在同一文件中是相同的屬意。而對於同義詞的辨別在製作所謂參照的時候也相當的有用，在探勘資料的時候可以增進其召回率。

在解決命名實體的意義之後，跨語言的翻譯工作就能進行。例如：Fung （1995）和 Huang （2005）就針對命名實體進行翻譯，是將命名實體從一種語言翻譯到其他語言的相關研究。命名實體在翻譯上的應用也演變成一個重要的機器翻譯技術。Vilar et al. （2006）的研究也指出，雖然有大於10%的翻譯錯誤率但對於人工翻譯的速度仍可以加快不少。Y. C. Wang （2009）也針對韓文和中文進行機器翻譯的相關研究。

另外、命名實體跳脫詞意之上，重要的是所指何物。為了達成這個目的。Charniak （2001）的研究是針對人名的結構進行分析，舉例來說 Doctor Paul R. Smith 則包含了幾個部份；人的職稱、第一名字、中間名字和姓。這可用來參照該職稱是對應到哪一個人物，因此利用此方法分析是否為同一人。如「John F. Kennedy」和「President Kennedy」指的是同一人，而 「John F. Kennedy」與 「Caroline Kennedy」則分屬兩人。Dimitrov （2002）進一步判斷句子中的代名詞所指的是哪一個人。「Rabi finished reading the book and he replaced the library」 這裡的代名詞他（he）指的就是 Rabi 本人。也可以利用這樣的照應方式設計出問題-答案式（Question Answering）的問答機器，如誰把書放回圖書館，答案就是 Rabi。

最後在是各資料庫命名實體的串連。Cohen & Richman（2001）的研究是在兩個不同的資料庫中去找尋相同的命名實體。Cohen & Sarawagi （2004）的研究也輔以分群和字串比對的方式來對資料庫中的命名實體進行整理和探勘。

參. 同位詞夾子演算法

台灣歷史數位圖書館的「明清檔案」是明朝到清朝間有關台灣的歷史資料，計三萬七千件全文資料。計畫在資料中萃取所有的人、事、時、地、物。對於人與地來說、是這五類中較具特徵的。首先面臨的問題是「何謂地名?」。為了能從三萬七千件全文資料中萃取所有的地名，我們得先對地名進行一番了解及定義。在教育部頒布之「重編國語辭典修訂本」(http://dict.revised.moe.edu.tw/)中，並無地名之定義。贛南師範學院客家研究院院長羅勇在其文章「罗勇：从地名中探寻文化之踪——评《赣州地名溯源》」中提到所謂地名就是「人們賦予某一特定空間位置上自然或人文地理實體的專有名稱」。但這樣的精準的定義，對於讓人理解何謂地名已然足夠，但是對自動萃取工作來說並沒有幫助。人的心中都知道何謂地名，但卻寫不出計算機可接受的定義來。

我們觀察「我今天要到東桃園上班」，我們知道「桃園」是一個地名。又比如說「我今天要到園桃來上班」，雖然「園桃」是我杜撰的新詞，可是我們就覺得它是一個地名。 所以人們判別地名的關鍵在於前後文，如此我們對於地名的定義大致上可以這樣說：

1. 那些前方可以接「到」、後方可以接「來」的字串。
   例如：到豐原來、到桃園來等。
2. 那些前方可以接「在」、後方可以接「上班」的字串。
   例如：在台北上班、在桃園上班。

但是這樣描述仍無法成為對地名的完整描述。

綜合以上觀察，地名有很高的同位性，「我今天要到桃園來上班」中的「桃園」用任何一個地名來替代，我們都不覺得奇怪。例如：台北、高雄、紐約、東京等。因為這些地名可以互相替代而不覺得奇怪，我們稱之為同位詞。所以對於地名這個主題來說，其所代表的是符合某些位置特徵的同位詞集。

為了進行地名的萃取，接下來的面臨兩大問題：一個是這些**規則該如何表示**；另一個是這些**規則該如何收集**來定義何謂地名。

既然這些規則與前後文有關，我們就使用（前文、後文）來當作一個規則，例如：（到、來）表示當前文是「到」、後文是「來」時就符合這個規則了；其表達的語意是當我們遇到了「到 OO 來」時，那個「OO」八成是地名了。隨著對地名的深度觀察，我們發現不僅僅是前後文會帶給我們地名的感覺，詞彙本身有什就可以帶給我們地名的感覺。例如：「台灣省」、「台北縣」、「中山區」等，其最後一個字（後綴）強烈表達了該詞彙為一個地名。又「台灣」、「台中」、「台北」、「台南」、「台東」、「台西」中，其最前一個字（前綴）則表達了該詞彙可能與台灣有關，應該是一個地名。所以我們又可以增加一些描述地名的方法。

3. 以「省」、「縣」、「區」結尾的字串。
4. 以「台」帶頭的字串。

所以在這樣的形容下，我們應該使用（前文、前綴、後綴、後文）來記錄一個規則，我們稱這樣的一個規則叫做「詞夾子」，因為我們可以使用這規則到文章中去夾出地名。只用到前後文的詞夾子（前文、""、""、後文），我們稱之為「**外夾**」，因為該類詞夾子是從外方來夾住候選詞。而只用到前後綴的詞夾子（""、前綴、後綴、""），我們稱之為「**內夾**」，因為這類詞夾子是從候選詞內部向外撐夾來夾出候選詞。除了內外夾外，單一詞夾子也可以同時擁有內外夾的特性。又因為「**詞夾子**」主要夾出同位詞，所以我們稱之為「**同位詞夾子**」。

前後文、前後綴，充分表現候選詞的特徵，以地名為例，前後文與**後綴**就相當重要，前綴表現就沒有那麼強；以人名

為例，前後文與**前綴**就相當重要，因為前綴是姓氏，強烈表達該候選詞可能是一個人名，而後綴表現就沒有那麼強。又各類主題詞庫有不同的特性，記錄規則（詞夾子）時，就會特別設計。比如在化學名稱上，我們觀察「氫氧化鈉」、「氫氧化鉀」、「硫酸鈉」、「碳酸鈣」、「甲基苯」、「乙基酯」等，可以觀察得到中綴「化」、「酸」、「基」是化學名稱的重要特徵。所以目前「同位詞夾子」是由五的部件所組成（前文、前綴、中綴、後綴、後文），主要描述一個詞彙在文件某處的前文、前綴、中綴、後綴、後文等特徵，用以在文件中萃取該詞彙的同位詞。

在使用**「同位詞夾子」（前文、前綴、中綴、後綴、後文）**來處理好**規則該如何表示**的問題後，我們要面對的是這些**規則該如何收集**的問題，也就是「同位詞夾子」該如何取得的問題。對於地名，我們可以觀察地名規則並撰寫詞夾子如下：

1. （到、""、""、""、來）：夾前接「到」、後接「來」的詞。
2. （在、""、""、""、上班）：夾前「在」、後「上班」的詞。
3. （""、""、""、縣、""）：夾取那些以「縣」結尾的詞。
4. （""、台、""、""、""）：夾取那些以「台」帶頭的詞。
5. （在、台、""、""、上班）：夾取那些前方接「在」、後方接「上班」且「台」帶頭的詞。

在此我們觀察第 5 個詞夾子，那是一個內夾與外夾混用的夾子，而且可以注意的是詞夾子的五個部件長度不一定是 1，也可以是長度為 2 的字串。在寫了這些詞夾子後，我們可以使用程式利用這些詞夾子到文庫中去夾取地名，在夾取時，被越多詞夾子夾住的候選詞，我們認為其越有可能是地名，我們將候選詞排序，使用人工來檢驗其正確性，挑選正確的詞彙進入地名集合中。但是我們知道這五個規則並不是

地名的完整定義，完整的規則量大而雜並不容易真確地整理，所以詞夾子不應該使用人工來撰寫，應該要自動產生。

於是我們自動化以上的流程，也就是從一群已知是地名的詞彙（稱之為種子集，例如：{苗栗、花蓮}），觀察這些種字在文庫中被哪些詞夾子夾住，收集該些詞夾子（例如上方 5 個詞夾子），然後再利用這些詞夾子在文庫中夾取候選詞，此時因為被種子集找出的詞夾子會很多（數十、數百到數千），且夾出的候選詞也會數以千計，所以賦予各詞夾子及各候選詞分數來標示其對原始主題（地名）的符合程度就非常重要，我們認為對一個詞夾子來說，

1.夾中越多的種子，分數越高；

2.夾中的詞彙中，種子純度越高，其分數越高。

對一個候選詞來說，

1.夾中該候選詞的詞夾子越多，其分數越高；

2. 候選詞被分數高的夾子夾中，其分數應加重。

如此我們可以依分數（地名符合度）來排序候選詞。分數越高的候選詞表示其越有可能是地名。然後我們使用人工來依序檢驗候選詞的正確性，將人工確認是地名的詞彙放入種子集中，然後再來一次。如此，一遍又一遍地，「種子集經文庫產生詞夾子，詞夾子經文庫產排序候選詞，人工介入挑選正確的詞彙擴充種子集」；「種子集經文庫產生詞夾子，‥‥」；‥‥，一直到對種子集的內容感到滿意為止。

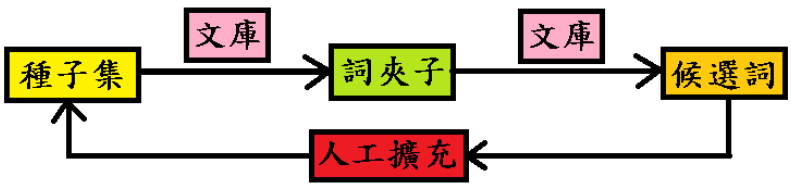

圖 1：詞夾子演算法的基本流程

圖 1 是以上流程的圖式化說明，在這樣的過程中，我們

可以注意到的是一開始使用種子集{苗栗、花蓮}所找出來的詞夾子，其實並非完整代表地名詞彙的行為，那只是苗栗和花蓮兩詞彙在文庫中的行為。又因為機器不懂何謂地名，所以「地名」兩字只存在操作者心中，機器不知操作者想要哪一個集合，因為{苗栗、花蓮}可能是地名的子集，但也可能是人名的子集，那是兩位姑娘的名字，一位姓苗名栗，一位姓花名蓮。所以在人工介入擴充種子集時，就會主導種子集更趨近於操作者心中所設定的主題，如果主題是地名，則由於人工的介入，種子集將越來越像地名集合；又如果主題是人名，則種子集將越來越像人名集合。當種子集被擴充了，產生的詞夾子就不只是{苗栗、花蓮}兩個詞彙的行為，而是整個種子集中詞彙（包含後來新增的詞彙）的行為，這時詞夾子夾到的候選詞其實是基於整個擴充後的種子集。也就是機器不知道操作者心中設定的主題，所以只能儘量夾出與種子集有類似行為的詞彙來猜測操作者想要的東西；而所謂類似的行為就是所謂的同位性。

同位詞夾子模具

一個詞夾子為一個（前文、前綴、中綴、後綴、後文）的組合，我們說一個詞彙在文庫中被哪些夾子夾住，這是一個不好回答的問題。例如：詞彙「桃園」在文章「我今天要到桃園來上班」中被哪些夾子夾住?因為未規範各部件的長度，所以

1. 前文長度可以從零到五計六種變化、
2. 前綴長度可以從零到二計三種變化、
3. 中綴長度可以從零到二，但因中綴可挪前挪後，所以計四種變化、
4. 後綴有三種變化、
5. 後文有三種變化。

總共有 648 個詞夾子，例如：

1. （今天要到、""、""、""、來）
2. （要到、桃、""、""、來）
3. （""、桃、桃、""、來）
4. …

但是 648 個詞夾子中跟地名有強烈直覺的只有一部分，所以我們需要對詞夾子進行分群，我們設計「詞夾子模具」用以描述詞夾子的特徵。所謂「詞夾子模具」是五個字串或數字的組合 （前文或前文長、前綴或前綴長、中綴或中綴長、後綴或後綴長、後文或後文長）。例如：

1. （1、0、0、0、1）：表示前後文長度為 1 且前中後綴長度為 0 的詞夾子。
2. （1、0、0、縣、1）：表示前後文長度為 1 且後綴一定要是縣但前中綴不管。
3. （2、0、0、0、1）：表示前文長度為 2、後文長度為 1 且前中後綴長度為 0 的詞夾子。
4. （1、0、0、1、1）：表示關心前後文及後綴各一個字，適合用來夾取「地名」。
5. （1、1、0、0、1）：表示關心前後文及前綴各一個字，適合用來夾取「人名」。

在程式開始時、需要事先設定好詞夾子模具，則種子集產生詞夾子時，只會產生符合詞夾子模具的詞夾子。此時（1、0、0、1、1）模具是很適合來夾取「地名」的而（1、1、0、0、1）是很適合來夾取「人名」的。

同位詞夾子演算法

在萃取的過程中，有兩個面相輪流重複出現，一個是機器處理、一個是人工介入。根據實驗，機器處理的時間極快，人工介入的時間則需較長。在種子集找出的候選詞對操作者主題的精準率會因種子集的大小有很大的關係，例如主題是人名時，以{苗栗、花蓮}為種子，找出的候選詞就大部分都不是人名，人工介入就會很辛苦，因會要剔除很多不是者。初期付出的人工成本，主要的因素是該種子集數量太少不足以代表該主題，所以在初期快速或自動取得一定量的種子集是重要的。根據實驗，我們發現在排序後的候選詞列表中，前面幾個詞彙具有超高的準確率，幾乎都是該主題下的詞彙，所以我們將人工介入的時間延後，直接認定前 k 個詞彙（k=3~5，稱 k 為保送名額）是符合主題的詞彙並將該些詞彙直接放入種子集中，此時種子集中將分兩區，一區是符合區：人工確認的符合主題的詞彙，另一區是待驗區：機器認為的符合主題的詞彙，我們依照該些詞彙進入待驗區的先後排序之。此時詞夾子演算法將如下：

1. 設定種子集及詞夾子模具，
2. 種子集依詞夾子模具經文庫產生詞夾子，詞夾子經文庫產生候選詞，機器自動將前 k 個（例如:前 3 個）候選詞擴充至種子集待驗區
3. 如果種子集待驗區達到某一數量（例如:2000 個），前往步驟 4。否則回到步驟 2，繼續使用機器自動擴充種子集。
4. 使用人工介入從種子集待驗區中挑選符合主題的詞彙進入種子集符合區，通常挑選數百個到一千多個，挑到覺得煩，覺得累了就不挑了。如果種子集符合區中的數量及內容已經滿意，則停止演算法，輸出種子集符合區；如果還不滿意，則重設種子集為種子集符合區，回到步驟 1，此時還可以人工加修種子集及詞夾子模具。

自動控制與手動操作

根據上段的討論，種子集是在變化的，我們可以記錄其變化的狀況，種子集一個狀態到下一個狀態可分兩種影響，一個是機器自動擴充，一個是人工介入改變（非擴充，因為待驗區中可能有些精確認不符合而剔除）。種子集的變化都是基於種子集上一個狀態，這種感覺就像是種子集緩緩慢逐步的在變化。在這個想法下，種子集的變化是受到人工及機器的控制，人工介入是手動操作，而機器擴充則是自動控制；自動控制會緩慢地改變種子集，引導種子集的發展。錯誤的引導當然會將種子集帶往錯誤的方向，導致後續的詞彙都是不相符的。此時，手動操作是需要的。手動操作則是大幅度修正種子集的發展，快速調整種子集的發展方向。

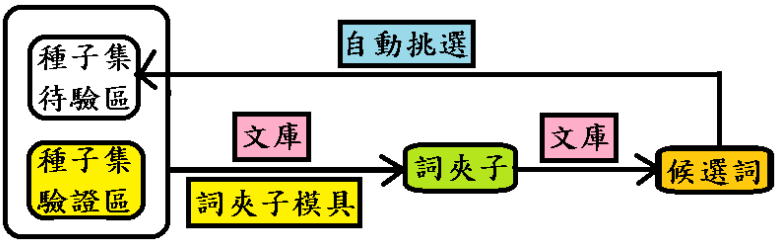

圖 2：詞夾子演算法之自動控制模式

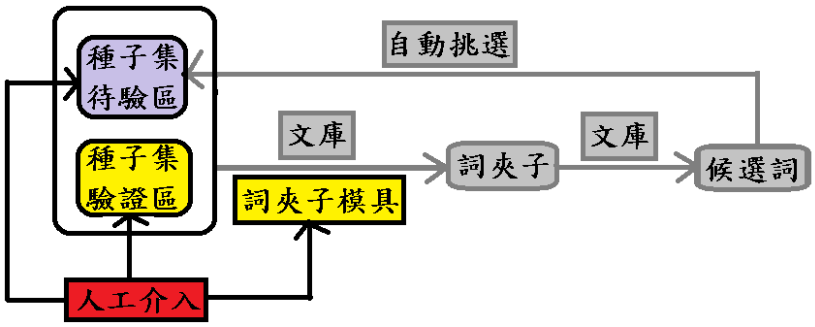

圖 3：詞夾子演算法之手動操作模式

同位詞夾子演算法之虛擬碼

為了能夠讓讀者可以實作詞夾子演算法，以下探討「同位詞夾子演算法」之程式化虛擬碼及範例說明。

1 事前工作：
  1.1 設定「文庫 D」。
  1.1 設定「種子集符合區 SM」。
  1.2 設定「負詞集 N」。
  1.3 設定「詞夾子模具集 M」。
  1.4 設定「候選詞長度集 R」。
  1.5 設定「保送名額 k」。
  1.6 設定「種子集待驗區飽和數 SVN」。
  1.7 設定「種子集待驗區 SV」為空集合。
2 主流程：
  2.1 將「文庫 D」中文章依標點符號分割成「文句集 T」。
  2.2 設定「種子集 S」為「種子集符合區 SM」與「種子集待驗區 SV」之聯集，並計算「種子詞長度集 SR」。
  2.3 「種子集 S」依「詞夾子模具集 M」及「負詞集 N」的要求，經「文句集 T」產生「詞夾集 C」。
  2.4 「詞夾集 C」依「候選詞長度集 R」及「負詞集 N」的要求，經「文句集 T」產生「候選詞集 H」。
  2.5 對「詞夾集 C」中之每一詞夾子賦予同位分數。此分數是根據該詞夾子與「種子集 S」及「候選詞集 H」之關係而得。
  2.6 對候「選詞集 H」中之候選詞賦予同位分數。此分數是根據該候選詞與「詞夾集 C」的關係而得。
  2.7 將排序後的「候選詞集 H」中，前 k 個候選詞擴充至「種子集待驗區 SV」中，其中 k 為設定之「保送名額 k」。
  2.8 如果「種子集待驗區 SV」中的詞彙個數比「種子集待驗區飽和數 SVN」小前往步驟 2.2。
3 暫停程式，等待人工手動操作。人工觀看「種子集待驗區 SV」將符合主題的詞彙放入「種子集符合區 SM」，不符合者放入「負詞集 N」中，操作者如果對「種子集符合區 SM」仍不滿意，則回到步驟 1.1 繼續執行。

在以上演算法中，步驟 2.1 及步驟 2.3 到步驟 2.6 較為

複雜，文章後方將再進行討論。以下針對上方演算法進行範例說明。

1 事前工作：
- 1.1 設定「文庫 D」。
  例如：「文庫 D」為數個文件之集合。
- 1.2 設定「種子集符合區 SM」。
  例如：{苗栗、花蓮市}
- 1.3 設定「負詞集 N」。
  例如：{一日三市}
- 1.4 設定「詞夾子模具集 M」。
  例如：{(1、0、0、市、0)、(1、0、0、0、1)}。
- 1.5 設定「候選詞長度集 R」。
  例如：{2、3、4、5}
- 1.6 設定「保送名額 k」。
  例如：3
- 1.7 設定「種子集待驗區飽和數 SVN」。
  例如：2000
- 1.8 設定「種子集待驗區 SV」為空集合。

2 主流程：
- 2.1 將「文庫 D」中文章依標點符號分割成「文句集 T」。
  例如：{
  桃園市位於中壢市之北、
  他要到花蓮來口試、
  你要到苗栗來找工作嗎、
  到花蓮市來出差
  那他今天有到花蓮市政府來上班、
  你今天有到新竹市去嗎、
  到一般政府機關去送公文、
  就是要到一日三市的黃金店面、
  我到桃園來的目的是逛街、
  到園桃市來送貨、
  到蘭宜來是為了開拓一片天地
  }
- 2.2 設定「種子集 S」為「種子集符合區 SM」與「種子集待驗區 SV」之聯集，並計算「種子詞長度集 SR」。。
  此時：「種子集 S」為{苗栗、花蓮市}，「種子詞長度集 SR」={2、3}。
- 2.3 「種子集 S」依「詞夾子模具集 M」及「負詞集 N」的要求，經「文句集 T」產生「詞夾集 C」。

此時：「詞夾集 C」為{
(到、""、""、""、來)
(到、""、""、""、政)
(到、""、""、市、"")
}。令以上三詞夾分別為詞夾甲、詞夾乙與詞夾丙。
詞夾甲夾中的種子有{苗栗、花蓮市}
詞夾乙夾中的種子有{花蓮市}。
詞夾丙夾中的種子有{花蓮市}。

2.4 「詞夾集 C」依「候選詞長度集 R」及「負詞集 N」的要求經「文句集 T」產生「候選詞集 H」。
此時：以下畫底線的為「種子集 S」中的詞彙，畫刪除線的為「負詞集 N」中的詞彙。剩餘的則為夾中之候選詞。
詞夾甲夾中{花蓮、苗栗、花蓮市、花蓮市政府、桃園、園桃市、蘭宜}
詞夾乙夾中{花蓮市、一般}
詞夾丙夾中{花蓮市、新竹市、~~一日三市~~、園桃市}
所以「候選詞集 H」為{花蓮、花蓮市政府、桃園、園桃市、蘭宜、一般、新竹市}。

2.5 對「詞夾集 C」中之每一詞夾子賦予同位分數。此分數是根據該詞夾子與「種子集 S」及「候選詞集 H」之關係而得。
假設分數是該詞夾子夾中種子的個數的平方除以夾中的詞彙數。則此時：「詞夾集 C」中各詞夾的分數如下：
詞夾甲：(到、""、""、""、來)：分數為 2*2/7=0.57
詞夾乙：(到、""、""、""、政)：分數為 1*1/2=0.50
詞夾丙：(到、""、""、市、"")：分數為 1*1/4=0.25

2.6 對候「選詞集 H」中之候選詞賦予同位分數。此分數是根據該候選詞與「詞夾集 C」的關係而得。
假設分數是夾中該候選詞的所有詞夾子分數之和。則此時：「候選詞集 H」中各詞分數如下：
園桃市：0.57+0.25=0.82
花蓮：0.57
花蓮市政府：0.57
桃園：0.57
蘭宜：0.57
一般：0.50
新竹市：0.25

2.7 將排序後的「候選詞集 H」中，前 k 個候選詞擴充至「種子集待驗區 SV」中，其中 k 為設定之「保送名額 k」。

此時將前 3 名詞彙：{園桃市、花蓮、花蓮市政府}擴充至「種子集待驗區 SV」中。

2.8 如果「種子集待驗區 SV」中的詞彙個數比「種子集待驗區飽和數 SVN」小前往步驟 2.1。

此時因為「種子集待驗區 SV」中只有三個詞彙，個數遠小於「種子集待驗區飽和數 SVN」(2000)，所以前往步驟 2.1，繼續自動控制。直到「種子集待驗區 SV」內個數達到 2000 才停止。

3 暫停程式，等待人工手動操作。人工觀看「種子集待驗區 SV」將符合主題的詞彙放入「種子集符合區 SM」，不符合者放入「負詞集 N」中，操作者如果對「種子集符合區 SM」仍不滿意，則回到步驟 1.1 繼續執行。

在詞夾子演算法中，步驟 2.1 及步驟 2.3 到步驟 2.6 較為複雜，以下分項進行詳細說明。

2 主流程：

2.1 將「文庫 D」中文章依標點符號分割成「文句集 T」。

此處須注意，如果候選詞位於文句開頭則不容易夾取，例如「桃園市位於中壢市之北」，則前方的「桃園市」無法被(1、0、0、0、1)夾夾住，因為其無前文，此時為每一個文句前方加一個特別符號^表示句頭，另一個特別符號$表示句尾。則演算法在處理時將可順利夾取句首與句尾之候選詞。例如：

{
^桃園市位於中壢市之北$、
^他要到花蓮來口試$、
^你要到苗栗來找工作嗎$、
^到花蓮市來出差$
^那他今天有到花蓮市政府來上班$、
^你今天有到新竹市去嗎$、
^到一般政府機關去送公文$、
^就是要到一日三市的黃進店面$、
^我到桃園來的目的是逛街$、
^到園桃市來送貨$、
^到蘭宜來是為了開拓一片天地$
}

2.3 「種子集 S」依「詞夾子模具集 M」及「負詞集 N」的要求，經「文句集 T」產生「詞夾集 C」。

此時需注意迴圈繞行的順序，不然會產生記憶體換頁錯誤，記憶體會不斷地換出換入，造成效能低

落。比較有效能的做法是：
對於「文句集 T」中的文句 t{
　假設文句 t 的開頭地址為 tb，結尾地址為 te。
　對於「詞夾子模具集 M」中的模具 m{
　　假設模具 m 五部件長度各為 m1、m2 到 m5
　　對於「種子詞長度集 SR」中的長度 sr {
　　　對於文句 t 中候選詞 h 開頭指標 b，tb≦b<te{
　　　　設定候選詞 h 結尾指標 e 為 b+sr。
　　　　計算候選詞 h 為地址 b 到 e 之字串。
　　　　如果候選詞 h 非種字詞集 S 中之詞彙，
　　　　　則繼續下一個 b。
　　　　如果 tb≦b-m1 不滿足，則繼續下一個 b。
　　　　計算前文為地址 b-m1 到 b 之字串。
　　　　如果 b+m2≦e 不滿足，則繼續下一個 b。
　　　　計算前綴為地址 b 到 b+m2 之字串。
　　　　如果 b≦e-m4 不滿足，則繼續下一個 b。
　　　　計算後綴為地址 e-m4 到 e 之字串。
　　　　如果 e+m5≦te 不滿足，則繼續下一個 b。
　　　　計算後文為地址 e 到 e+m5 之字串。
　　　　對於可能的中綴開頭 d，b≦d<e{
　　　　　如果 d+m3≦e 不滿足則繼續下一個 d。
　　　　　計算中綴為地址 d 到 d+m3 之字串。
　　　　　如果詞夾 5 部件有不滿足模具 m 之
　　　　　　文字部件要求，則繼續下一個 d。
　　　　　建立詞夾子 c 為此 5 部件。
　　　　　設定詞夾子 c 與種子詞 h 之夾住關係。
　　　　}
　　}
}

2.4　「詞夾集 C」依「候選詞長度集 R」及「負詞集 N」的要求經「文句集 T」產生「候選詞集 H」。
與 2.3 類似，需注意迴圈繞行的順序，不然會造成效能低落。比較有效能的做法是：
對於「文句集 T」中的文句 t{
　假設文句 t 的開頭地址為 tb，結尾地址為 te。
　對於「詞夾子模具集 M」中的模具 m{
　　假設模具 m 五部件長度各為 m1、m2 到 m5
　　對於「候選詞長度集 R」中的長度 r {
　　　對於文句 t 中候選詞 h 開頭指標 b，tb≦b<te{
　　　　設定候選詞 h 結尾指標 e 為 b+r。
　　　　計算候選詞 h 為地址 b 到 e 之字串。
　　　　如果該詞為負詞集 N 或種子集中之詞彙，
　　　　　則繼續下一個 b。

如果 tb≦b-m1 不滿足，則繼續下一個 b。
計算前文為地址 b-m1 到 b 之字串。
如果 b+m2≦e 不滿足，則繼續下一個 b。
計算前綴為地址 b 到 b+m2 之字串。
如果 b≦e-m4 不滿足，則繼續下一個 b。
計算後綴為地址 e-m4 到 e 之字串。
如果 e+m5≦te 不滿足，則繼續下一個 b。
計算後文為地址 e 到 e+m5 之字串。
對於可能的中綴開頭 d，b≦d<e{
如果 d+m3≦e 不滿足則繼續下一個 d。
計算中綴為地址 d 到 d+m3 之字串。
如果詞夾 5 部件有不滿足模具 m 之
文字部件要求，則繼續下一個 d。
建立詞夾子 c 為此 5 部件，
設定詞夾子 c 與候選詞 h 之夾住關係。
}
}
}

2.5 對「詞夾集 C」中之每一詞夾子賦予同位分數。此分數是根據該詞夾子與「種子集 S」及「候選詞集 H」之關係而得。

此處分數的設計，可以有很多變化，只要合乎直覺都可以有不錯的表現。對於一個詞夾來說，其夾中率為夾中種子的個數除以夾中所有詞彙的個數。一個詞夾夾中的種子數目越多，分數越高；一個詞夾的夾中率越高，分數越高。所以系統設定一個詞夾的分數為夾中的種子數目乘上其夾中率。

2.6 對候「選詞集 H」中之候選詞賦予同位分數。此分數是根據該候選詞與「詞夾集 C」的關係而得。

此處分數的設計，可以有很多變化，只要合乎直覺都可以有不錯的表現。對於一個候選詞來說，越多的詞夾夾中，表示其與主題越相符，但是詞夾有強弱性，被具代表性的詞夾夾中，應該有較高的分數。所以系統設定候選詞的分數為夾中該詞的所有詞夾子分數之和。

懸涯現象

根據實驗，自動控制收集的待驗區詞彙依照其進入該區的順序有一個明顯的懸涯現象，在某一個時間之前進入的詞

彙幾乎都是符合的，而該時間之後進入的詞彙幾乎都是不符合的，所以人工介入的時候，發現懸涯現象的時候，就可以停止了，將手動操作換回自動控制。懸涯現象表示，種子集的變化受種子集本身的影響非常嚴重，當種子集中進入了一些模擬兩可的詞彙時，或是同屬於兩三個主題的詞彙時，種子集的發展就會受到影響。

六十七問題

從明清檔案中萃取人名時，有一個懸涯現象發生在種子集收錄了一個詞彙「六十七」之後，接著 所有的數字就陸陸續續都收進來了，我們當然覺得「六十七」這個詞彙應該是夾錯了，可是經過查證後，發現在乾隆年間真的一位巡臺御史名叫「六十七」，但是因為「六十七」在文庫中同時展現人名的行為及數字的行為，機器無法分辨何時「六十七」是人名，何時「六十七」是數字，更何況機器也不懂何謂人名、何謂數字。以下是李光濤（1959）在《明清檔案存真選輯(初集)》中，收錄關於「六十七」的奏摺。

巡視臺灣戶科給事中紀錄三次臣六十七謹奏，為恭謝天恩事。

本年正月十五日，接准都察院劄，開乾隆十年十一月初六日，奉旨稽察吉林烏拉事務著塔坦去巡視臺灣；六十七著再留任二年。欽此。欽遵。劄行前來。臣恭設香案望闕叩頭謝恩訖。

伏念臣賦性凡庸至微極陋，由內閣中書歷任刑部主事，因北路軍前行走議敘，仰荷皇上天恩，陞授禮部員外郎。更蒙簡擢監察御史，隨擢戶科給事中，於乾隆八年奉命巡視臺灣。二年以來寸長未効。茲復奉旨留任二年。臣聞命之下感激無地，惟有益加奮勉，務竭駑駘，以圖仰報皇恩高厚于萬一耳。所有微臣感激下忱，理合繕疏恭謝天恩。為此，謹具奏聞。

文庫特性

在明清檔案中萃取人名，觀察其詞夾子可以發現，人名在明清檔案與其他文庫中的用法很不相同，例如（臣、""、""、""、謹奏）是當中一個很重要的詞夾子，但是其他文庫中並不會有如此的行為，所以主題行為是因文庫而異，無法對一主題在沒有確定文庫下，就有辦法完全定義的。也就是說何謂「地名」、何謂「人名」，是無法被定義的。所以命名實體的萃取將因文庫不同、主題不同而具有獨特性。

實務操作

詞夾子程式是一個非常好用的工具，但是使用詞夾子演算法萃取主題式分類詞庫所需的時間與操作者本身的經驗有非常大的關係。這就像 Google 的搜尋引擎是很好的搜尋工具，但是在相同目標下，所需的搜尋時間與操作者本身的經驗有非常大的關係。詞夾子演算法的的手動操作部份需要相當的經驗。

在文庫中夾取主題詞彙（例如：地名），務求快速擁有一定數量的地名，來方便自動控制不會走錯方向。此時、我們可以觀察各詞夾模具的精準狀況，例如（2,1,0,1,2）模具所描述的詞夾子具有相當強烈的力道，兩個詞同時被此種夾子夾到，就表示這兩詞非常的相似，因為其有兩字前文、兩字後文、一字前綴、一字後綴，這要求非常強烈，這兩詞幾乎必屬同類。例如：（前往、台、""、縣、定居）夾到的一定是「台中縣」、 「台北縣」、「台南縣」、「台東縣」等詞。詞夾子的力道越強烈、種子集的變化越小、同質性越高、符合主題的希望越大，在擁有一定數量後、再依序慢慢放寬使用（2、0、0、1、2）模具、（2、0、0、1、1）模具、（1、0、

0、1、2）模具、（1、0、0、1、1）模具、最後在幾乎再也找不到地名時，改用（1、0、0、0、1）模具來放寬搜尋。另外也可使用更強烈的文字模具來快速取得對的詞。例如：先使用 （1、0、0、縣、1）、（1、0、0、市、1）、（1、0、0、鄉、1）、（1、0、0、州、1）、（1、0、0、府、1）等模具，等待數量夠多之後再使用（1、0、0、1、1）模具，然後在最後收尾的時候才使用（1、0、0、0、1）模具和（0、0、0、1、0）模具。詞夾子程式在實務經驗上，也並非都是一開始時微調，然後逐步放寬。交錯使用強烈模具與寬鬆模具是常見的，完全依賴操作者的智慧。

肆. 精準率與召回率討論

「同位詞夾子」演算法，主要利用人工來保證精準率，利用機器速度來補足召回率，以達到極高的準確率與儘量高的召回率。是一個不需要詞庫就可以找尋詞集的方法，是透過半監督學習的方式來建立詞集。在此特性之下，非常難與現有方法進行比較，因為「詞夾子演算法」強調人工的介入確保極高精準率，所以在精準率上就無法與其他演算法進行比較。又強調人工介入，所以種子集是經過人工調整的，傳統演算法是不經人工調整的，所以比較召回率就會跟時間有關，對「詞夾子演算法」來說，越久的操作將有越高的召回率，但是傳統演算法大都是執行完畢即有固定的召回率，所以在召回率上是難跟其他演算法比較的。另外在時間上，「詞夾子演算法」的流程都是以數小時、數天為單位，速度上無法與其他演算法相比，而且起始種子集的不同將嚴重影響種子集成長的狀況。所以在精準率、召回率、執行時間上與其他演算法無法比較的情況下，我們唯有討論「詞夾子演算法」自己的一些數據狀況及在幾個計劃中的成果。另外在半監督式學習的子領域上，文庫的影響非常大，相關研究有自己專用的文庫，文庫屬性及操作者經驗將影響執行的狀況，在這種情況下，也無法與半監督學習演算法進行比較。

在領域應用方面，也確實存在需要極高精準率與儘量高召回率的領域。這與 Google 搜尋引擎相當類似，搜尋結果的精準率遠遠重要於召回率，搜尋結果只需要在有限結果內，展現絕佳的精準率即可。在中文詞彙應用領域上，我們舉三個例子，分析精準度與召回率的重要性。

1.文庫詞彙索引表輔助文庫導讀係指將文庫中具代表性及對文庫帶有特殊意義之詞彙收集供讀者點擊查閱，類似書本後方的索引表（Index）。在此應用之下，因為需要人工介入判斷詞彙是否對文庫帶有特殊意義，所以精準率是「普通重要」，不是問題，但是因為索引表本來就是要用來幫助讀者閱讀的，文庫作者最害怕的其實是索引表的不完善，所以在此考量下、召回率是「**極為重要**」的。

2.文庫搜尋引擎索引檔建立係指增加詞彙在索引檔中的強度。在全文檢索時，有很多非詞彙也會在索引列中，特別是使用 N-gram 演算法時容易造成此種現象，這時增加已知詞彙的權重，有助於搜尋結果品質的提升，也可降低非詞彙的權重或是去除來縮減索引檔的大小。因為是搜尋引擎，其重要之處在於搜尋結果的相關性排序，中文詞庫只是站在輔助的角色而已，所以此應用對詞彙的精準率與召回率的要求並不高，僅是「普通重要」而已。

3.文庫搜尋結果的詞類分析導讀係指針對文庫搜尋引擎的搜尋結果進行詞彙分析，以使用詞彙的分類詞庫將該子集在各分類上的特性解析，例如在搜尋「賈寶玉」的搜尋結果中，分析其出現的人名、地名、服裝、飲食等等，在此分析下，因為分類詞庫表達了分類的概念且外顯於讀者，所以精準率是「**極為重要**」的，而召回率則是「普通重要」。

表 1：精準率與召回率分析

	精準率	召回率
1.文庫索引表導讀	「普通重要」	「**極為重要**」
2.搜尋引擎索引檔建立	「普通重要」	「普通重要」
3.文庫子集詞類分析導讀	「**極為重要**」	「普通重要」

精準率與召回率，並非同時都很重要；其中詞庫使否外顯於讀者是一個很重要的因素，文庫管理者不希望詞彙分類出現錯誤讓讀者覺得該文庫並不專業，所以 80%的精準率是不堪使用。這就像光學文字辨識（Optical Character Recognition（OCR））可以幫忙建立影像文章的索引檔一樣，因為是將影像文章給讀者觀看，其文字辨識結果未外顯於讀者，所以精準率越高越好但不苛求，但如果將文字辨識結果供讀者閱讀，那麼極高的精準率是基本要求。

對於精準度「**極為重要**」的要求，人工是免不了的，主因是仍然存有太多連人也會有不同意見的地方。例如究竟「同位詞夾子」是指「同位詞」的「夾子」，還是指「同位」的「詞夾子」。又比如說，何謂地名，「台北」是個地名，「大樹下」是不是一個地名。對於自然語言的處理，我們相信，「人的問題須要人來幫助」，即使是 Google 的 Page Rank，也是靠觀察極大量人類使用鍵結狀態來對目標網頁進行評分，所以我們也可以說是所有網頁的建構者在幫忙蒐尋引擎進行相關性排名。

伍. 同位詞夾子的實務應用

在 2005，項潔、謝育平與張尚斌（2006）合作萃取明清檔案中的分類詞彙並成功設計及開發詞夾子程式。在明清

檔案三萬七千件全文檔中成功萃取 1373 個人名。為了測試精準率與召回率，使用自動控制只得到 47.3%的精準率與78.4%的召回率，同時著名的「六十七問題」，也是在這時候發現的。此時台灣大學數位典藏與自動推論實驗室正在建設台灣歷史數位圖書館，此工具從此大量使用在數位人文相關議題上，直到今日，實驗室內各項中文研究都直接或間接使用此工具。例如：廖傋凡（2010）使用詞夾子萃取儒林外史中的人名及稱謂進行加值研究。

在 2006 年，謝育平擔任某家論文資料庫公司顧問時，製作醫學論文摘要分類萃取詞彙，使用詞夾子萃取領域名（如表 2）計 1,188 詞、疾病名計 1,703 詞、症狀名計 1,059 詞、癌症名計 473 詞如下表，並實際應用於搜尋引擎資料後分析呈現（如圖 5）的使用上。另外從圖 4 可以看到領域名稱在萃取實的狀況，相當穩健。

表 2：醫學論文摘要分類萃取詞彙

詞類	詞彙
領域名	公羊學、生物力學、口腔細胞學、土壤化學、中西哲學、丹麥兒童文學、內視鏡學、公共行政學、分子生態學、心臟病學等 1188 詞。
病名	十字花科蔬菜黑斑病、下呼吸道感染病、口啼疫病、子宮外孕病、川崎氏病、弓形蟲病、中耳炎病、中華肝吸虫感染病、內分泌系統疾病、巴金森氏病等 1703 詞。
症狀名	β 受體功能亢進症 二氣化碳血症 子宮腺肌症　中風後遺症　分娩併發症　心肌梗塞後併發症　心絞痛症　支氣管哮喘氣道炎症 牙科畏懼症 出血症等 1059 詞。
癌症名	支氣管肺癌 水平膀胱癌 外陰癌　白斑癌 皮脂腺樣癌 舌鱗癌 舌鱗癌　尿道癌 肝细胞癌 乳管癌等 473 詞。

0	53933	1359	39	1506	35	光學	Search	13100000	0	0	0	0%
1	606436	15552	38	16762	36	化學	Search	52900000	0	0	0	0%
2	281987	7223	39	7424	37	力學	Search	7420000	1546	155	9	2%
3	320336	8023	39	8678	36	理學	Search	9140000	243	53	4	0%
4	248770	6918	35	7050	35	科學	Search	136000000	225	61	3	0%
5	1026370	11564	88	28460	36	醫學	Search	51100000	574	99	5	0%
6	199785	7405	26	5481	36	動力學	Search	4070000	2546	262	9	4%
7	236485	2014	117	6793	34	大學	Search	165000000	1055	160	6	2%
8	223739	10221	21	6185	36	生物學	Search	6350000	2568	294	8	4%
9	231746	6935	33	5933	39	統計學	Search	2950000	889	115	7	1%
10	206042	4969	41	5202	39	物學	Search	143000	2214	209	10	4%
11	148029	891	166	4437	33	的學	Search	2370000	554	91	6	2%
12	147376	12416	11	4110	35	組織學	Search	988000	2008	303	6	7%
13	160175	452	354	3754	42	計學	Search	51200	375	29	12	0%
14	111682	10278	10	3078	36	影像學	Search	1280000	4464	411	10	13%
15	129202	3906	33	3356	38	態學	Search	26900	3899	230	16	6%
16	130430	8448	15	3751	34	流行病學	Search	995000	4092	363	11	9%
17	128826	15514	8	3291	39	病理學	Search	98200	6769	533	12	16%
18	82972	1007	82	2304	36	小學	Search	51900000	951	100	9	4%
19	100965	11681	8	2744	36	形態學	Search	539000	9340	522	17	19%
20	117899	1572	74	2986	39	病學	Search	3220000	1307	100	13	3%
21	112738	707	159	2882	39	行病學	Search	15500	604	51	11	1%

```
<drDoc>d:\clip\TextBank</drD
<fnDoc></fnDoc>
<fnOut>subject.htm</fnOut>
<fnSearchNumber>subject.tgn<
<Examples>
化學 光學
</Examples>
<NegativeTerms>
中學 教學 產學
</NegativeTerms>
<Iteration>       -1     </It
<EliteNumber>      4     </El
<CandicateNumber> 500    </Ca
<tmClipCoverage>  0.1    </tm
<tmTermOccurs>    3      </tm
<Clips>
(1,0,學,1) [2-10]
(2,0,學,1) [2-10]
(1,0,學,2) [2-10]
(2,0,學,2) [2-10]
(3,0,學,2) [2-10]
(2,0,學,3) [2-10]
(3,0,學,3) [2-10]
(3,0,學,1) [2-10]
(1,0,學,3) [2-10]
</Clips>
<ClipStopWords>
？ 、 。 、 ； ： ＼ ！ ？ 「 『 （ 【 】
＄ ￥ € ℃ mmcmkmkgcc ″ 》 ： — （ 「
</ClipStopWords>
<NegtiveClips></NegtiveClips
<TermStopWords></TermStopWor
<TermStopPhrase></TermStopPh
```

圖 4：使用詞夾子萃取領域名稱時的候選詞及詞夾子模具等

圖 5：應用詞夾子演算法之資料庫搜尋介面

2009 年，謝育平等（2009）使用詞夾子建立中文典籍分析加值服務，選用 432 部中文典籍，實作搜尋引擎及自動閱讀分析代理程式，供中文領域研究學者使用，並作為中文典籍領域的研究基礎建設。針對每一部典籍決定有意義的詞庫類別，然後將維基百科中該部典籍該類詞庫目前的記載當成樣本，使用人工半自動詞夾子程式進行詞類萃取以擴充詞庫。例如針對封神演義中的人名詞庫，維基百科中記載 55 個人名，使用人工半自動詞夾子萃取程式後，新增了 344 個人名，其增加的幅度不難發現該研究對封神演義的人物研

究有其貢獻。根據實驗，使用人工半自動詞夾子萃取程式處理一個分類詞庫平均約需 14 分鐘的機器時間及 35 分鐘的人工時間，平均新增 64 個詞。這表示建立典籍針對性分類詞庫並不是一件難事。研究共處理 96 部典籍，139 個分類詞庫，計 21538 個詞。在分類上，共取 13 個類別：人物、女姓、分類、地名、年號、衣物、武器、法術、神怪、動物、國家、飲食、寶物。如表 3 所見，值得注意的是紅樓夢的人名萃取，從維基百科中取得 22 人，經操作後可以擴充增加 716 人，即使是水滸傳已有 108 人，也可增加 121 人，甚至是紅樓夢的飲食，這個不常見的萃取類別，也可找到 157 個詞彙，這些都說明詞夾子是實用之工具。

表 3：萃取詞類及新增加的詞彙

典籍	分類	維基百科樣本	增加
三國演義	人物	1174	0
封神演義	人物	55	344
水滸傳	人物	108	121
紅樓夢	人物	22	716
西遊記	神怪	0	178
紅樓夢	飲食	0	157
東周列國志	年號	0	108
後與西遊記與補	地名	0	186
後與西遊記與補	武器	0	66
後與西遊記與補	法術	0	54
紅樓夢	地名	0	149
紅樓夢	衣服	0	199
紅樓夢	飲食	0	157

2010 年，謝育平等（2010）應用詞夾子演算法於廣域性搜尋引擎 Google 搜尋結果的後分析。搜尋引擎在日常生

活上已被廣泛的使用，可是依舊存在許多 Google 無法在一般使用者容忍的時間內找到資訊需求的搜尋問題，我們稱之為 Google 難題（Google-Hard Problem）。為解決 Google 難題，我們採取精度推薦與廣度推薦並以分類推薦進行輔助，精度方面採傳統學習演算法、廣度方面則採詞夾子演算法，總後統整成分類展現，讓使用這再使用搜尋引擎時，可以取得搜尋結果的後分析、並請受到分析結果的導引。

圖六：使用詞夾子之搜尋引擎助理程式

陸. 結論與未來發展

「詞夾子演算法」是一個半自動的主題式詞庫萃取工具，屬於半監督式學習工具。主要先設定一個主題，給定少量種子集，定設定詞夾子模具，使用種子集及模具從文庫中提取詞夾子，再使用詞夾子經文庫夾出有序候選詞，最終反覆使用自動駕駛與手動駕駛來引導種子集的成長狀況以達操作者心中所設定的主題。

「詞夾子演算法」主要利用同類詞在文庫中具有同位性的概念來來萃取主題詞庫。其所萃取之詞並不具詞意，也無主題之意，純粹是「同位詞」，這是操作者須謹記在心的，須小心設定詞夾模具及種子集和負詞集(確定不屬於主題詞庫，但有具長有高分之詞，放入負詞集以排出之）來手動及自動駕駛操控種子集發展。

「詞夾子演算法」基於「同位詞」，還有很多可應用之處，例如將之應用於文字考古學，辨識未知甲骨文之同位詞，因為同位詞具有同類性，所以如果某一詞為已知之器具，則其同位詞應該也是器具，可以協助考古學家來辨識未知文字。

又「詞夾子演算法」可以根據同位性計算各詞彙之同位候選詞，這表示自動對仗是有可能的，「千江有水」對「萬里無雲」這是因為「千」與「萬」是同位詞、「有」與「無」」是同位詞、「水」與「雲」是同位詞等等，我們可以根據上聯，讓電腦自動對出下聯。

又「詞夾子演算法」可以表彰文庫作者之習慣，不同的文庫其找出的詞夾子不同，並可依詞夾子在文章中的作用設定字句間的流暢度，來辨別一篇未知作者之文章為何人所做。

柒. 參考文獻

1. Asahara, Masayuki; Matsumoto, Y. （2003）. Japanese Named Entity Extraction with Redundant Morphological Analysis. In Proc. Human Language Technology conference - North American chapter of the Association for Computational Linguistics.
2. Bikel, D.; Miller, S.; Schwartz, R. and R. Weischedel. （1997）. Nymble: a high-performance learning name-finder. In Proceed-ings of the Fifth Conference on Applied Natural Language Processing, pages 194–201.
3. Borthwick, Andrew; Sterling, J.; Agichtein, E.; Grishman, R. （1998）. NYU: Description of the MENE Named Entity System as used in MUC-7. In Proc. Seventh Message Understanding Conference.
4. Brin, Sergey. 1998. Extracting Patterns and Relations from the World Wide Web. In Proc. Conference of Extending Database Technology. Workshop on the Web and Databases.
5. Charniak, Eugene. （2001） Unsupervised Learning of Name Structure from Coreference Data. Proc. Meeting of the North American Chapter of the Association for Computational Linguistics.
6. Chien, Lee-Feng （1997）, PAT-Tree Based Keyword Extraction for Chinese Information Retrieval, ACM SIGIR.
7. Cohen, William and Richman, J. （2001） Learning to Match and Cluster Entity Names. Proc. International ACM SIGIR Conference on Research and Development in Information Retrieval. Mathematical/Formal Methods in IR （workshop）.
8. Cohen, William W. and Sarawagi, S. （2004） Exploiting

Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. Proc. Conference on Knowledge Discovery in Data.

9. Dimitrov, Marin; Bontcheva, K.; Cunningham H and Maynard, D. （2002） A Light-weight Approach to Coreference Resolution for Named Entities in Text. Proc. Discourse Anaphora and Anaphor Resolution Colloquium.
10. Grishman, Ralph; Sundheim, B. （ 1996）. Message Understanding Conference - 6: A Brief History. In Proc. International Conference on Computational Linguistics.
11. Fung, Pascale （1995）. A Pattern Matching Method for Finding Noun and Proper Noun Translations from Noisy Parallel Corpora. Proc. Association for Computational Linguistics.
12. Huang, Fei （ 2005 ）. Multilingual Named Entity Extraction and Translation from Text and Speech. Ph.D. Thesis. Carnegie Mellon University.
13. Kim, Seokhwan; Song , Yu; Kim, Kyungduk; Cha , Jeong-won and Lee, Gary Geunbae（2006）. MMR-based active machine learning for bio named entity recognition, In Proceedings of the Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting （HLT-NAACL06.）
14. Mann, Gideon S. and Yarowsky, D. （ 2003 ） Unsupervised Personal Name Disambiguation. Proc. Conference on Computational Natural Language Learning.
15. McCallum, Andrew; Li, W. （2003）. Early Results for Named Entity Recognition with Conditional Random Fields, Features Induction and Web-Enhanced Lexicons. In Proc. Conference on Computational Natural Language

Learning.

16. Mikheev, A.; Moens, M.; Grover, C. （1999）. Named Entity Recognition without Gazetteers. In Proc. Conference of European Chapter of the Association for Computational Linguistics.
17. Minkov, Einat; Wang, R. and Cohen, W. （2005） Extracting Personal Names from Email: Applying Named Entity Recognition to Informal Text. Proc. Human Language Technology and Conference Conference on Empirical Methods in Natural Language Processing.
18. Nadeau, David （2007）, Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision, Ottawa-Carleton Institute for Computer Science, School of Information Technology and Engineering, University of Ottawa, Canada.
19. Nadeau, David and Sekine, Satoshi （2007）. A survey of named entity recognition and classification. Linguisticae Investigationes, 30（1）:3–26.
20. Nadeau, David and Turney, P. （2005） A Supervised Learning Approach to Acronym Identification. Proc. Canadian Conference on Artificial Intelligence.
21. Sekine, Satoshi. （1998）. Nyu: Description of the Japanese NE System Used For Met-2. In Proc. Message Understanding Conference.
22. Vilar, David; Xu, J.; D'Haro, L. F. and Ney, H. （2006） Error Analysis of Statistical Machine Translation Output. Proc. Language Resources and Evaluation conference.
23. Wang, R. C., and Cohen, W. W. （2007）. Languageindependent set expansion of named entities sing the web. In Proceedings of ICDM.
24. Wang, Yu-Chun; Tsai, Richard Tzong-Han; and Hsu, Wen-Lian. Web-based pattern learning for named entity translation in Korean-Chinese cross-language information

retrieval. Expert Systems with Applications.

25. Wu , T. and Pottenger, W. M..（2005）. A semi-supervised active learning algorithm for information extraction from textual data. Journal of the American Society for Information Science and Technology, pp. 258-271.
26. 台灣歷史數位圖書館，http://thdl.ntu.edu.tw。
27. 李光濤，《明清檔案存真選輯(初集)》（台北市：中央研究院歷史語言研究所，1959 年，平裝本），頁 162-163。
28. 張尚斌（2006），〈詞夾子演算法在專有名詞辨識上的應用：以歷史文件為例〉，台北：台灣大學電機資訊學院資訊工程研究所碩士論文。
29. 廖儁凡（2010），〈中國古典白話小說中的社會網路關係：以儒林外史為例〉，台北：台灣大學電機資訊學院資訊網路與多媒體研究所碩士論文。
30. 謝育平、詹登淵、郭子文（2010），〈協助解決 Google 難題的資訊萃取機制〉，《銘傳大學 2010 國際學術研討會》，97-106 頁。
31. 謝育平、楊龍廉、趙建宏、黃銘立、古馮文、林郁智（2009），〈使用詞夾子建立中文典籍分析加值服務〉，《銘傳大學 2009 資訊科技與實務研討會》，78-91 頁。
32. 羅勇：從地名中探尋文化之蹤——評《贛州地名溯源》，http://www.ganyoo.com/thread-2369-1-1.html。