行政院國家科學委員會專題研究計畫　成果報告

子計畫三：以閱聽人資訊為基礎的 Web 新聞系統之設計與實作(I)

計畫類別： 整合型計畫
計畫編號： NSC91-2213-E-004-011-
執行期間： 91 年 08 月 01 日至 92 年 07 月 31 日
執行單位： 國立政治大學資訊科學系

計畫主持人： 沈錳坤
共同主持人： 陳百齡

報告類型： 精簡報告
報告附件： 出席國際會議研究心得報告及發表論文
處理方式： 本計畫可公開查詢

中　華　民　國 92 年 11 月 1 日

行政院國家科學委員會專題研究計畫成果報告
以閱聽人資訊為基礎的 Web 新聞系統之設計與實作(I)
Design and Implement Web News Systems based on Readers' Information (I)

計畫編號：NSC 91-2213-E -004-011
執行期限：91 年 8 月 1 日至 92 年 7 月 31 日
主持人：沈錳坤　政治大學資訊科學系
共同主持人：陳百齡 政治大學新聞系

一、中文摘要

在 Web 為基礎的媒體服務與運作平台上，分析閱聽人在 Web 新聞系統的閱聽行為，在外部將可提供閱聽人個人化新聞與社群推薦的功能，在內部則可提供新聞企畫人員有關分眾閱聽行為分析、新聞事件分析等功能。本計畫第一年主要在研究新聞網站上閱聽人的查詢行為。閱聽人在查詢新聞時，都是依據存在腦海裡的已知知識來下達查詢，而這些已知知識，可以看成是一個 Ontology。Ontology 可用來組織、管理與分享知識。本計畫分析閱聽人在新聞網站下關鍵字查詢時的行為，並結合與查詢關鍵字有關的新聞資訊，以輔助建構 Ontology。Ontology 由閱聽人所查詢的關鍵詞組成，並依此建構出關鍵詞間彼此的關係，關係包括上下義、同義，與其他相關性。我們主要利用查詢關鍵詞間的先後順序關係與邏輯關係，搭配關鍵詞的屬性，來分析關鍵詞間的關係。所建立的 Ontology 將可應用在新聞網站的分類索引，也有助於分析專家與生手在媒體概念上的異同。

關鍵詞：Web 新聞系統、資料探勘、本體論學習、查詢記錄檔、查詢行為

Abstract

The searching behavior of readers is according on their knowledge. And the knowledge can be considered as Ontology. Ontology can be used to organize, manage and share knowledge. But it's usually a time-consuming and error-prone task. Therefore, using computer to assist in constructing Ontology becomes an important issue. In this way, we implement a Web news system to construct a general news ontology from readers' searching behavior. Utilizing methods of Knowledge Discovery to help Ontology Engineering is called Ontology Learning. In this project, we do Ontology Learning by using those pages related query terms and analyzing the querying behavior of readers on news sites. The Ontology is made up of terms queried by users, and there are some relations among these terms. These relations are hyperonomy, synonymy and the "relative" relation we define. Thus, our goal of this project is to learn the correct relations among these query terms. The system can automatically collect logs, extract and analyze query keywords, and output the final Ontology.

Keywords: Web News System、Data Mining、Semantic Web、Ontology、Query Log、Query Behavior

二、緣由與目的

每個人在做每件事情，都是依據存在腦海裡的知識去行動，例如上網閱讀新聞，一定會先從他最感興趣的新聞開始閱讀，這就是依據他的喜好所表現出來的行為，而喜好也是屬於知識的一部份，所以一旦能夠得知閱聽人的知識，自然也就可以明白閱聽人的喜好。

要表達知識並不是一件簡單的事情，尤其是現今資訊化的時代，由於資訊氾濫，許多的資料不容易整合，也因此不容易利用。Ontology 的出現，即為了解決資訊交換的問題，只要將知識建構成 Ontology，即可達到知識的互相交換，以及再利用的功能。一般在建構 Ontology 時因為相當費時，因此可能會利用 Knowledge Discovery 等方法來輔助建構，即為 Ontology learning。

本計畫的重心，即為從新聞網站的查詢紀錄中，以 Ontology learning 的方法建構 Ontology，目前針對查詢紀錄的研究中，還沒有人將閱聽人在查詢時，下關鍵字查詢的先後順序考慮進來。例如，觀察閱聽人下查詢時的行為，通常的狀況是閱聽人下了某個查詢，結果新聞網站回傳了太多資訊，閱聽人接著會重新查詢比前一組查詢字串更具有特定意義的關鍵字，來過濾太多且複雜的資訊，此種閱聽人的行為帶來的訊息為，後一組查詢字串在概念上位於前一組查詢字串的下層，也就是說，前者可能是後者的上義詞。

本計畫最主要的目的，即在分析閱聽人下關鍵字查詢資訊時的行為，利用這個資訊建構出能代表閱聽人的知識的 Ontology。

三、結果與討論

1 原理

我們的目的為從閱聽人的查詢資訊裡面去自動建構出新聞的 Ontology。主要可以分為幾個步

驟，首先利用 Web Log Preprocessing 和 User Session Identification 從 Log 裡面找出閱聽人的查詢記錄，之後再利用 Query Session Identification 找出閱聽人從開始查詢一則新聞到找到該則新聞所下的查詢關鍵詞。接著利用 Phrase Relation Identification 去判斷關鍵詞之間的關係。最後就可以得到一般閱聽人的 Ontology。

1.1 Ontology 定義

Ontology 是一種描述事實的方法，可以有各種各樣的形式，以較細部的角度來看，Ontology 最基本的需要包含：以定義好的字彙來描述已經存在的實體，應用一些規格來表示出這些實體間的關係與存在的意義，而形成在某個領域裡面中可以解釋其知識的架構。以 Computer Science 知識分享的角度，Ontology 是描述存在某個代理人裡面的概念(concept)和關係(relation)，簡單的說，可以看成是一個概念和關係的集合。而 Ontology 可用來組織、管理與分享知識，使溝通更容易。

本計畫中 learning 的 Ontology，整合了查詢關鍵字彼此之間的關係，我們所尋找的關係為上下義詞、同義詞，以及最基本的有關。

```
Entry 1.
210.70.195.237 - - [13/Oct/1999:11:50:40 +0800]
"GET /query.cgi?term=教育部" 200 65536
"http://udndata.com/query.cgi"
"Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)"
Entry 2.
210.242.31.70 - - [13/Oct/1999:11:51:01 +0800]
"GET /query.cgi ?term=評鑑指標" 200 2418
"http://udndata.com/query.cgi "
        "Mozilla/4.0 (compatible; MSIE 4.01; Windows
```

圖 1.1

1.2 Web Log Preprocessing

閱聽人在新聞網站上的行為，如存取網頁與點選網頁的 link 時，所有的動作都會被記錄在 web log 中，因此，藉由分析 web log 的內容，便能夠得知閱聽人的瀏覽行為。圖 1.1 為 web log 的範例，內容為閱聽人對網站發出 request 後的狀態記錄，其中記載了下列欄位：

1.IP address：閱聽人的來源，例如 210.70.195.237。

2.Timestamp：閱聽人發出 request 的時間，例如 1999 年 10 月 30 日中原標準時間 11 時 50 分 40 秒。

3.Response：在 Request 之後是一組三位數字，其代表回應的狀態，例如 200 代表正確的回傳網頁。

4.Size：在狀態碼之後的是傳送出的檔案大小，例如 65536 個 byte。

5.Referer：指向 request 的網頁，亦即此 request 的前一個 request。

6.User-Agent：記錄閱聽人的作業系統及使用的瀏覽器，例如例子中為 windows98，瀏覽器為 IE4.01。

如果當時 Request 的欄位是如 CGI 等程式，除了記錄閱聽人瀏覽的網頁外，也會同時記錄閱聽人查詢的字串，如圖 1.1 中的第一筆資料，記錄了閱聽人查詢 query.cgi 時，用了"教育部"這個查詢字串。

由上面的介紹我們可以了解在 web log 中包含了許多的資訊，然而 log 內常有一些不必要的東西，像是圖檔、音效檔等，此類檔案的下載可能不是閱聽人對該檔感到興趣，而是由於瀏覽器在載入網頁的同時一併下載的。另外，有些閱聽人可能會使用像 teleport 之類的工具大量抓取網頁，但此動作與他真正的瀏覽行為並無太大關係，因此必需將這些雜訊排除在外。

1.3 User Session Identification

在 Web log 裡面有一筆一筆的閱聽人查詢紀錄，一個 User Session 即為一個閱聽人的查詢記錄。我們先定義：

1. U: User session。
2. Q: Query Session。
3. r: Web log 裡面的查詢記錄。
4. id: Web log 裡面的 ip address。
5. t: Web log 裡面的 Timestamp。
6. q: 每一筆查詢記錄裡面的查詢詞。

所以對一個 User Session U_i 來說：

1. $id_1 = id_2 = id_3 = \ldots = id_n$。
2. for all r_i，$t_i <= t_{i+1}$。

1.4 Query Session Identification

針對每一個 User Session，可以利用幾個方法去切出 Query Session，方法如下：

1. 利用兩筆查詢記錄間的時間差，如果時間相隔太久，則定義為不同的兩個 Query Session。
2. 利用前後查詢詞之間的相似度，如果低於某個值，則定義為不同的兩個 Query Session。
3. 利用閱聽人是否點選網頁作為判斷，如果有點選可能代表有找到要找的資料，因此前後兩筆查詢記錄則分屬於不同的 Query Session。

由以上我們可得出一個 Query Session Q_i：

For all $r_i \in Q_i$:

1. $ti+1 - ti <$ time threshold。
2. sim(qi, qi+1) > similarity threshold or no click between r_i, r_{i+1}。

1.5 Phrase Relation Identification

在這節裡，我們介紹如何從前面找出的 query session 中，實際探勘 phrase 之間的 relation。首先，我們利用 phrase 先後查詢的關係，分析其可能的 relation，此 relation 為 phrase 間的 candidate relation，接著，再計算以 phrase 的 feature 為標準的 relation，綜合二者的資訊後，即為我們最後認定的 phrase 的 feature。

Candidate Discovery of Phrase Relation

我們所要分析的是，當閱聽人做出了先後查詢的行為，而產生 query session 時，session 中各別的先後 phrase pair，其 phrase 間的關係是什麼。既然我們知道，除了查詢結果是空的以外，閱聽人的連續查詢我們認定是為了減少回傳的資訊，來趨近查詢的目標，因此，閱聽人的查詢行為，結合 boolean logic 後，可以轉換成如表 3.1 中的集合關係，其中左側圖示的實心部分代表閱聽人欲查詢的結果所對應之集合，右側的結果表示當閱聽人下某種查詢行為時，他想查詢的結果可不可能符合左側的集合，例如 A→A+B 這個查詢，我們可以發現它可能

是表 1.1 中的(a)或(c)的集合關係，表示當閱聽人先下了查詢字串 A，再下 A+B 時，B 對應的集合可能是 A 的 subset，或 A 與 B 有部分交集，或 AB 意義相近。

表 1.1

	A→B	A→A+B	A→A/B	A→A-B
(a) 	V	V		
(b) 				V
(c) 		V		
(d) 				V
(e) 				
(f) 				

實際觀察閱聽人所下的查詢，我們發現幾乎所有的閱聽人都不會使用 boolean logic 來搜尋目標，大多不是完全置換關鍵詞(如 A→B)，便是將原來的查詢新增其他關鍵詞(如 A→B+C)，因此，下面我們的討論便主要針對這兩種查詢來探討。如表 1.2 所示，query session 中完全置換或新增關鍵詞的查詢，總共可分為四種情形，分別是 single phrase→single phrase、single phrase→multi phrase、multi phrase→single phrase 與 multi phrase→multi phrase，以下分別就這四種 query type 來討論 phrase 的 relation。

表 1.2

Query Type	Query String
Single → Single	A→B
Single → Multi	A→A+B
	A→B+C
Multi → Single	A+B→C
Multi → Multi	A+B→A+C
	A+B→C+D

<1>single phrase → single phrase:此種 case 閱聽人只會做 A→B 這種查詢，而閱聽人在查詢了一組 phrase 後，會下另一次的查詢的原因，可能是因為第一次的查詢結果是空的，或者是閱聽人認為找錯了，回傳的結果不是他要的，再不然便是回傳結果太多了。因此，這些線索便成了我們判斷 phrase 之間關係的重要依據，首先，我們先檢查 A 的回傳結果，如果是空的，表示此次查詢是因為閱聽人查不到東西，此時通常閱聽人會下另一個同義詞或上義詞來重新查詢；如果回傳結果不是空的，則我們檢查閱聽人是不是有翻頁或點選瀏覽的動作，如果沒有，可能是找錯了，此時他可能重新下一個類似的同義詞來查看看，但如果有這些動作，則應該歸屬於回傳結果太多，此時，我們利用新聞網站的回傳筆數，來粗略地估計 A 與 B 的關係，假設 A 的回傳筆數大於 B 的話，則 A 可能為 B 的上義詞，反之 B 為 A 的上義詞，兩者筆數約略相等時，則可能是同義詞。

<2>single phrase → multi phrase:此種 case 又分為兩種情形，一種是後面的查詢包含前面的 phrase，如 A→A+B，另一種則是不包含，以第一種情形來看，A 與 B 任何關係都可能，同樣我們交由搜尋引擎的回傳筆數來預估兩者關係，另一種情形為 A→B+C，跟 A→A+B 不一樣的地方在於一般第一次的查詢沒有回傳東西的話，閱聽人不會直接下多個 phrase 來查詢，而會直接用兩個 phrase 來取代前者的查詢，通常後者會比前者更具有特定的意義，因此我們判定為 A 可能是 B 與 C 的上義詞。

<3>multi phrase → single phrase: 此情形如 A+B→C，C 通常具有比 A 與 B 更特定意義，因此判定為 A 與 B 為 C 的上義詞。

<4>multi phrase → multi phrase:此種 case 的查詢算是較特殊的一種，出現次數較少，A+B→A+C 的情形，由於次數少且關係不定，因此我們跳過此種情形，而 A+B→C+D 則多半為 phrase identification 沒有發現的 Single → Single case，我們將其轉為"A B"→"C D"來處理。

Phrase Feature Extraction

我們的系統提供了兩種 phrase 的 feature，分別為 keyword_based 與 document_based，keyword_based 的 phrase feature 做法為對 phrase 的查詢回傳新聞，利用字典對其做 keyword extraction，此時，phrase 的 keyword_based feature 即代表此 phrase 包含了哪些字典中的詞。假設字典中有 n 個詞，則 phrase 的 keyword_based feature 可能為(k1 k2 … kn)的 keyword vector，假設 phrase 的搜尋回傳結果與相關新聞網頁中只包含字典中第一與第三個詞的話，則他的 keyword vector 為(101 … 0)。

第二種 document_based feature 做法為比對 news database，看 phrase 出現在哪些新聞中，以此當做 phrase 的 feature。假設 news database 中有 m 個網頁，而第一、第二、及第四個網頁包含 phrase 這個詞的話，此 phrase 的 document vector 則為(120 1 … 0)。

Final Relation Validation

由 Candidate Discovery of Phrase Relation 找出的 relation 並不一定正確，仍需進一步由 phrase feature 來判斷其正確性，至於從 feature 計算 phrase 之間的關係，我們採用了[Lawrie, 2000]中，計算 phrase 間彼此互相 subsume 的情形，來判斷 relation 的方法。其中

X subsume Y 的值為：

$$\frac{|X \cap Y|}{|Y|} \quad (1)$$

Y subsume X 的值為：

$$\frac{|X \cap Y|}{|X|} \quad (2)$$

給定一 subsumption_threshold 值，如果 X 與 Y 互相 subsume 值皆大於此 threshold，則 X 與 Y 為同義詞，否則，X 對 Y 的 subsumption 值大於此 threshold 時，X 為 Y 的上義詞，反之 Y 對 X 大於此值的話，Y 為 X 的上義詞，皆不符合的話則 X 與 Y 沒有關係。

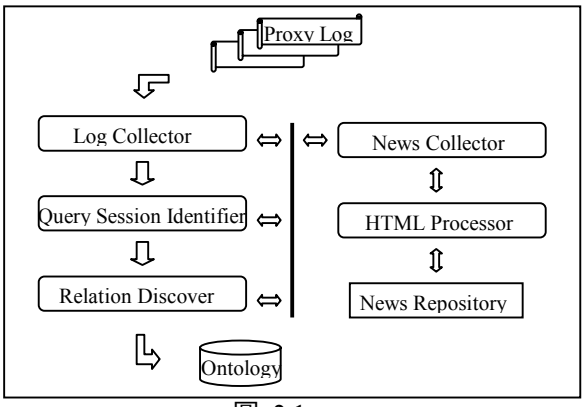

圖 2.1

2 系統實做

圖 2.1 為根據系統流程所實作的系統架構，底下我們就實作上的一些議題分別探討。

2.1 Log 檔的選擇

分析閱聽人在新聞網站上的行為，在資料來源方面，需取得該新聞網站的 web log 才行，但此 log 通常為商業機密，並不容易取得，以及新聞網站並不一定記錄了閱聽人對於查詢結果的點選情形，甚至對 web log 做 user session identification 時，還需考慮閱聽人設定了 proxy server 時帶來的影響，因此，取用網站上面的 web log 當作分析的資料來源並不十分恰當。

除了直接取用新聞網站的 web log，另一個選擇是 proxy server log，當閱聽人設定好瀏覽器的 proxy server 後，所做的網頁瀏覽動作都將透過 proxy server 送出，閱聽人的所有行為，全都無所遺露的記錄在 proxy server 的 log 中，因此，本研究選擇的資料來源，即為 proxy server 上面的 web log。

2.2 News Repository 的建立與網頁資訊處理

在 Query Session Identification、Candidate Discovery of Phrase Relation，以及 Phrase Feature Extraction 這些階段，往往需要向新聞網站重新查詢當初閱聽人的 query，而 extract phrase 的 keyword_based feature 時，更是需要將原本 query 的回傳結果中的延伸連結都抓取回來，因此，這些重新向新聞網站要回的 news，即形成了 news repository，功用為當需要重新檢示某個 link 的內容時，可以直接向 news 要資料，減少對新聞網站的負擔，此 repository 亦成了 document_based feature extraction 的資料來源。

重新抓取資料時，我們使用 w3m 工具向搜尋引擎 sequential 的 fetch，避免瞬間送出大量的要求，而一個 news 抓取後，必需做資訊處理的動作，如回傳結果的 result size，以了解與此 news 相關的資訊量約有多少；包含一則新聞的網頁中除了 html tag 需刪除外，新聞網站原本使用的 template 也必需去除，以免 extract keyword_based feature 時產生誤差。

3 實驗評估

3.1 實驗環境與資料來源

我們實驗的環境為 P4 2.0G，2GB 記憶體，跑 FreeBSD 5.1-Current 作業系統，相關程式則用 perl coding。實驗資料來源部分為政大計中四台 proxy server 的 log，由於 query log 資訊容易牽涉到個人隱私部分，因此我們取得的資料為經過計中執行過 log preprocessing 與 user session identification 的結果，且只取在查詢中 query term 出現過三次以上的 query，以盡量避免涉及個人隱私權的問題。

最後實驗的資料，經過計中 preprocessing 過後，我們取 9/1~9/20 來做，所有 log 檔合併後的大小為 432MB，共有 4689630 筆資料，18669 個 user session。

上述 log 檔經過 query session identification 後，共計有 2.2MB，21673 筆資料，4084 個 query session；經過 candidate relation discovery 產生了 2395 個 phrase pair 的 relation，最後，再與 feature extraction 的做法合併比較，產生 Ontology，與 keyword_based 的 feature 合併計算後，共計有 285 個 phrase pair relation，與 document_based 的 feature 合併計算 Ontology，產生了 271 個 phrase pair relation，如表 3.1 所示。

表 3.1

File Size after Preprocessing	432MB
Log Size after Preprocessing	4689630
Query Sessions after Preprocessing	18669
File Size after Query Session Identification	2.2MB
Log Size	21673
Query Sessions	4084
Relations number after candidate relation discover	2395
Relations number after feature extraction	285
Relations number with document_based feature	271

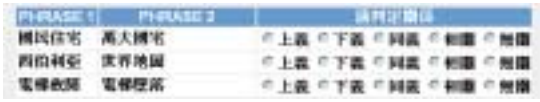

圖 3.1

3.2 實驗評估

最後的實驗，我們將 learning 出來的 Ontology relation，以圖 3.1 中的 webpage 表示，請人來判斷兩者間的關係，左邊顯示了 learning 出來具有某種 relation 的 phrase pair，右邊則請閱聽人判斷 phrase1 是 phrase2 的何種關係，關係的選擇有"上義"、"下義"、"同義"、"相關"與"無關"五種，之後用來和我們做出來的相比較來計算準確率。

準確率的測量上，我們計算實驗結果的 precision 與 learning accuracy(LA)：

$$\text{precision: } = \frac{N_{output\_correct}}{N_{output}} \qquad (3)$$

有些關係的存在，並不是單純只有有無二分法，用程度上的差別會比較適當。而以我們定義的 Ontology relation 來看，上下義、同義或有關的關係，並不是完全獨立或互斥，如兩個 phrase 是同義詞時，兩者之間實際上具有一定的上義或下義關係，因此，我們定義了表 3.2 中的 learning accuracy 分數計算法，公式如下：

$$LA := \sum_{i \in \{1...n\}} \frac{LAi}{n} \quad \text{with}$$

$$LAi := Score(Relation1, Relation2) \quad (4)$$

表 3.2

Relation1	Relation2	Score
上義	上義	1
上義	下義	0
上義	同義	0.3
上義	有關	0.5
上義	無關	0
下義	下義	1
下義	同義	0.3
下義	有關	0.5
下義	無關	0
同義	同義	1
同義	有關	0.5
同義	無關	0
有關	有關	1
有關	無關	0.5

3.3 實驗結果

我們的實驗，請五個人分別判斷 document_based 與 keyword_based feature，與 query session 兩兩產生的 Ontology，並測量其正確率，結果如表 3.3 所示。

表 3.3

	Doc_based	Keyword_based.
Precesion1	0.24	0.23
LA1	0.39	0.42
Precesion2	0.34	0.29
LA2	0.55	0.47
Precesion3	0.16	0.21
LA3	0.36	0.42
Precesion4	0.23	0.48
LA4	0.64	0.52
Precesion5	0.17	0.32
LA5	0.53	0.42

四、計畫成果自評

Ontology 的組成為閱聽人查詢的關鍵字，當新聞網站上面的查詢紀錄餵進我們的系統後，即會經過 log preprocessing、user session identification、query session identification、phrase extraction 與 relation discovery 等動作，最後分析出關鍵字彼此間的關係，自動完成 Ontology 的建立。在關鍵字關係的分析上，我們考慮的特徵有閱聽人下查詢時的先後行為、下完查詢後的點選行為、瀏覽行為，加上新聞網站對於該次查詢所回傳的新聞資訊，如回傳的新聞多寡、以及相關連結的內容等等。

在未來的研究方面，除了尋找更多閱聽人在查詢時，可供利用的特徵之外，較需值得探討的地方如區分不同的閱聽人，例如專家或一般閱聽人。如果可以做到的話，將會增進建構出的 Ontology 的準確度及可利用性。

五、參考文獻

[1] Lawrie, D. & Croft, W.B. (2000). Discovering and Comparing Topic Hierarchies. *Proc. of RIAO 2000 Conference*.

[2] Alfonseca, E. & Manandhar, S. (2002). Improving an Ontology Refinement Method with Hyponymy Patterns. *Proc. of International Conference on Language Resources and Evaluation* LREC'02.

[3] Beeferman, D. & Berger, A. (2000). Agglomerative Clustering of a Search Engine Query Log. *Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[4] Berendt, B., Mobasher, B., Spiliopoulou, M. & Wiltshire, J. (2001). Measuring the Accuracy of Sessionizers for Web Usage Analysis. *Proc. of Workshop on Web mining, SIAM Conference on Data Mining*.

[5] Byrd, R. J. & Ravin, Y. (1999). Identifying and Extracting Relations in Text. *Proc. of International Conference on Applications of Natural Language to Information Systems* NLDB'99.