# 一個識別特定主題深網查詢介面的分類器


| 周清江 | 張珮慈 | 鄭又誠 | 石永瑜 |
|---|---|---|---|
| 淡江大學資訊管理學系 | 淡江大學資訊管理學系 | 淡江大學資訊管理學系 | 淡江大學資訊管理學系 |
| 副教授 | 碩士班研究生 | 碩士班研究生 | 碩士班研究生 |
| cjou@mail.tku.edu.tw | coldbows@gmail.com | ccchenny@gmail.com | yungyu.shih@msa.hinet.net |



## 摘要

根據研究估算，深層網路(Deep Web)的規模大約為表層網路(Surface Web)的400~550 倍，為了擷取深網資料庫的內容，首先必須找出資料庫的入口，即深網查詢表單。此外，由於深網內容通常屬於某個特定主題，為了從眾多該特定主題的網頁表單中識別出深網查詢表單，本研究提出一個兩階段的分析方法，結合提交查詢前之表單分析以及提交查詢後之表單分析，發展一個自動化深網查詢介面識別技術。實驗結果顯示，我們提出的方法可以得到高精確度(precision)，不僅可過濾搜尋引擎這類的非深網查詢表單，更可自動偵測及過濾連結失效的查詢表單。



**關鍵詞**：深層網路、查詢介面、搜尋引擎


## 1.前言

隨著網際網路的快速發展，其中蘊藏了豐富的資訊。傳統的搜尋引擎僅能針對靜態網頁爬行，透過網頁上的超連結索引資料，索引到的頁面集合又稱為表層網路(surface web)。

相對於表層網路，網路上另外存在深層網路(Deep Web, 本研究簡稱為深網)，深網指的是不能被傳統搜尋引擎索引到的內容，這些內容只有在被查詢時才會經由網站伺服器動態產生頁面，將結果回傳給使用者，因此並沒有超連結指向這些頁面。

欲存取深網中的內容，首要步驟是必須找出連接線上資料庫的查詢介面。目前連接線上資料庫的查詢介面大多都是以HTML(HyperText Markup Language, 超文件標示語言)表單的形式呈現，然而，並非所有 HTML 表單都是深網的查詢介面，例如：用戶註冊表單、留言表單等等。

針對尋找深網查詢介面這個問題，目前有 completeplanet[1]等相關的深網目錄網站針對查詢介面進行收集與整理，其是採用人工的方式判斷表單是否為深網的查詢介面，之後再依後端資料庫的內容對查詢介面進行分類。然而，這個方法主要有以下幾個缺點：

1. 無法即時反映現況。由於網站內的查詢介面都是以人工方式進行收集與整理，因此當連結失效時，必須等待管理者定期維護之後，才能符合現況。
2. 涵蓋範圍與規模仍有所限制。根據 Chang 等人[6]的研究指出，最大的深網目錄網站 completeplanet 覆蓋率僅有 15.6%，覆蓋率明顯偏低。

為了有效辨識特定主題深網查詢介面，本研究提出一個兩階段的分析方法，結合提交查詢前之表單分析以及提交查詢後之表單分析，發展一個自動化深網查詢介面識別技術。不同於其他研究，本研究不僅能識別出查詢表單，更能進一步過濾搜尋引擎、站內搜尋這類只對靜態網頁進行索引的非深網查詢表單。

在前置準備階段，我們會建立非查詢表單欄位特徵字，並透過大量爬行特定主題查詢表單以擷取出該主題常見欄位語意。我們的分類系統，在提交查詢前之表單分析這個階段，使用非查詢表單欄位特徵字優先過濾常見的非查詢表單，以降低提交查詢的時間成本。在參考提交查詢結

[1] http：//www.completeplanet.com/

果之表單分析這個階段，我們利用常見欄位語意對表單自動填值以實際對表單自動提交查詢，並根據查詢回傳的結果進一步分析，以判定表單是否為特定主題的深網查詢介面。實驗結果顯示，我們提出的方法可以得到高精確度，不僅可過濾搜尋引擎這類的非深網查詢表單，更可自動偵測及過濾連結失效的查詢表單。

## 2.背景介紹

廣義而言，深網(Deep Web, 又稱 invisible Web、hidden Web)，指的是不能被傳統搜尋引擎索引到的內容，主要包含以下四個部分：

1. 透過對後端資料庫提交查詢而回傳取得的動態頁面。
2. 由於缺乏指向自己的超連結而沒有被搜尋引擎索引到的內容。
3. 具備某些存取限制的內容。(如：需要註冊或付費的內容)
4. 搜尋引擎無法處理的內容。(如：多媒體檔案裡的文字內容)

在實際應用中會特別關注第一個部分的內容，主要是因為這種由提交查詢取得的動態頁面通常包含結構化資料，對於後續的資料整合會更有意義。

網際網路中分佈著各個主題領域的查詢介面，其中大部分都是以 HTML 表單的形式呈現，然而，並非所有的 HTML 表單都是深網的查詢介面(即深網資料庫的入口)。

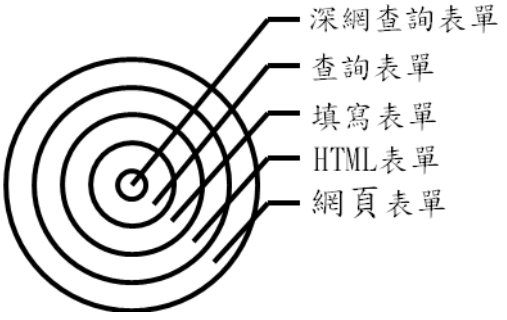


**圖 1 網頁表單分類**

如圖 1 所示，所有的網頁表單中，存在少部分非 HTML 表單；所有 HTML 表單中也存在部分非填寫表單；所有填寫表單中，另外存在非查詢表單；而所有查詢表單也並非都是深網查詢表單。如何從所有 HTML 表單中自動識別出深網查詢介面是本研究最主要的目標。

## 3.文獻探討

我們將於本章討論深網查詢介面識別的相關研究，於 3.1 節簡介網頁爬行器，說明為擷取表單，在網際網路空間中爬行可使用的四種策略，3.2 節和 3.3 節則討論在識別深網查詢介面這個議題上，以往學者提出的兩個主要的研究方向，分別是未提交查詢之表單分析(pre-query)以及參考提交查詢結果之表單分析(post-query)。

### 3.1 網頁爬行器

為了擷取深網資料庫中的內容，我們必須先取得深網資料庫的入口，即深網查詢表單。使用網頁爬行器(web crawler)自動在網頁之間爬行，以擷取出查詢表單，之後，必須對擷取到的表單進行分類，以驗證是否為深網查詢表單，因此分類的正確與否。現有的網頁爬行器大致可分為以下四類：通用類、特定主題類、查詢表單物件類、特定主題及查詢表單物件類。

通用爬行器(General Crawler)是以網頁為基礎爬行(page-based)，藉由遞迴巡訪網頁中的每個超連結，來達到收集所有網頁的目標。Brin 與 Page[3]提出具備 PageRank 以及 Anchor Text 這兩個特點的搜尋引擎系統。PageRank 的概念是，當越多連結指向某個網頁，該網頁的重要性越高。系統另外也針對 Anchor Text 進行了特殊處理，將 Anchor Text 和它所導向的

網頁進行關聯，通常 Anchor Text 會比網頁本身能更精確地描述該網頁。這種方式對於搜尋特定物件的獲益率[2] (harvest) 是很低的。

特定主題爬行器 (Topic-Focused Crawler)的爬行方式類似通用爬行器，但僅針對特定主題領域中的網頁爬行。Chakrabarti 等人[5] 提出的作法是事先定義主題類別，在系統初始時，使用者必須先匯入樣本資料作為訓練網頁分類器的基礎，網頁分類器負責判斷爬行到的網頁與特定主題的相關性，以決定是否要加入網頁中的連結。特定主題爬行器可大幅提升爬行特定主題網頁的效率，而且由於僅針對特定主題，相較於一般的爬行器，特定主題爬行器資料更新的頻率可以比較高，能更加敏銳地察覺主題範圍內網頁的變化。然而，主題導向爬行器並不適用於爬行難以區分領域、主題中立的網頁。

不同於一般爬行器皆以網頁為基礎，He 等人[8]提出一個以網站為基礎的查詢表單物件爬行器(Object-Focused Crawler)。其概念是先蒐集查詢表單物件在各個網站的分佈特性與結構特徵，將整個網際網路的網頁依所屬網站為基本單位，針對每個網站皆爬行到深度 3 的位置，尋找符合上述特性之表單。He 等人提出的做法主要有兩個優點: 第一點，爬行器是以結構資訊來進行探測，因此可處理所有主題領域，不受內容影響。第二點，可取得穩定的獲益率與覆蓋率[3](coverage)，並進一步預測爬行器執行的效益。

特定主題及查詢表單物件爬行器(Topic & Object-Focused Crawler)結合特定主題爬行器以及查詢表單物件爬行器，其是以網站為基礎進行爬行，並藉由使用內容分類器識別目標物件，適用於針對特定主題領域的爬行。

[2]在目前所有爬行的網頁中，確實找到目標物件的比率。

[3]所有目標物件中實際被找到的比率。

### 3.2 未提交查詢之表單分析

未提交查詢之表單分析(Pre-Query)指的是藉由擷取及分析表單的相關特徵來判斷表單是否為深網查詢介面，其步驟為:

1. 擷取及解析表單特徵。
2. 未提交查詢之表單分類。

Nguyen 等人[11]提出一個學習分類器 LABELEX 根據表單排版樣式(form layout pattern)識別出表單元素的標籤。其中，LABELEX 的對應產生元件會負責針對元素鄰近的各個可能的標籤進行配對。He 等人[9]提出一個架構，使用介面綱要模型(interface schema model)擷取出查詢介面所呈現的語義資訊，將查詢介面轉換成機器可以理解的形式。並發展以排版表示式為基礎(layout-expression-based)的擷取技術，其概念是對於 IEXP 中的每一個表單控制項元素(e)，LEX 會在鄰近的列中依據相關的衡量標準找出最可能為欄位標籤的字串，並且根據這個字串對表單控制項元素進行聚集，以找出控制項之間的關係。

Cope 等人[7] 提出一個擷取表單特徵，並以 C4.5 演算法建立決策樹的方法。控制項的型別與控制項的數量也是可以擷取的特徵。在擷取表單特徵之後，接著使用 C4.5 演算法建立決策樹，以作為分類 HTML 表單的依據。

### 3.3 參考提交查詢結果之表單分析

參考提交查詢結果之表單分析(post-query)指的是實際對表單提交查詢，並根據回傳的結果頁面判斷表單是否為深網查詢介面。

實際提交查詢時，由於每個介面提供的查詢功能都不相同，必須了解表單介面

各欄位的語意，以針對所有的表單欄位，都能給定一個該表單所屬網站伺服器能夠處理的值。

Wu 等人[13]提出 AVG (Attribute-Value-based Graph) model，概念是將每個深網資料庫視為一個獨立的無向圖，每個不同的屬性值會使用一個頂點來表示，而當某兩個屬性值同時存在於一筆資料紀錄中時，表示屬性值彼此間的兩個點會被連結起來。因此爬行深網資料庫的問題就可以被轉換為是一個巡訪圖形的問題。

Shu 等人[12]提出一個查詢功能模型(querying capability model)，其基本概念是，將組成有效查詢的最小屬性集合視為一個基礎查詢(atomic query)，並且藉由識別每個查詢介面的基礎查詢來自動建立查詢功能模型。另外使用一個類似 BNF (Backus-Naur Form)的模型來描述查詢介面的功能，建構出該功能模型。

Barbosa 與 Freire [1]針對以關鍵字為基礎的查詢介面(keyword-based interface)提出一個自動產生具代表性的關鍵字集合的方式，並建立高覆蓋率的查詢以擷取隱藏於後端的資料。他們發展一個以抽樣為基礎的演算法(sampling-based algorithm)，能自動找到可取得高召回率(recall)的關鍵字，並使用這些關鍵字建立查詢，擷取資料庫中可用的資料。

深網查詢介面提交查詢後動態產生的回傳結果，通常包含許多豐富的資訊，例如：查詢結果、廣告及導覽。

Caverlee 等人[4]提出 QA-Pagelet 的概念，指一個動態頁面中包含查詢結果的內容區域。也實作 THOR 的探勘系統，可自動偵測並擷取出 QA-Pagelet。共分為三階段：第一，必須提交查詢以收集深網網站回傳的查詢結果頁面；第二，必須在動態產生的結果頁面中識別出 QA-Pagelet；第三，則繼續針對上個階段產生的子樹進行分割，將包含的查詢結果項目分別擷取出來。

Hedley 等人[10]提出的方法是針對回傳頁面中的文字內容及鄰近的標記結構進行分析，以擷取出查詢結果。將回傳頁面轉換為標記區段(tag segment)及文字區段(text segment)，將文字區段以向量來表示，並根據特定字詞在區段中出現的頻率給定權重值。藉由不同回傳頁面中，所產生的文字區段間的相似度，將會擷取相似度最低的文字區段，即可能為查詢結果。

Bergholz 等人[2] 實作一個深網爬行器，系統會準備欲爬行網站的集合，接著使用爬行器依設定的爬行深度與巡訪的網頁數量，以廣度優先搜尋的方式識別候選網頁，並記錄每個包含 HTML 表單的網頁。接著使用表單分析器分析收集的 HTML 表單，判斷表單是否為查詢表單，對於查詢表單則會繼續實際提交查詢，並分析回傳結果的有效性。

## 4. 特定主題深網查詢介面分類器

我們將於本章說明深網查詢介面分類器的實作內容，於 4.1 節說明整體系統架構以及各個系統元件的功能，於 4.2 節說明分類器在前置準備子系統所執行的各項工作，於 4.3 節說明此分類器的兩階段分類方法─提交查詢前之表單分析以及提交查詢後之表單分析。

### 4.1 系統架構與元件說明

本系統整體架構如圖 2，依照其功能與執行順序可分為兩個子系統，左邊的前置準備子系統以及右邊的分類子系統。在分類子系統，我們採取兩階段的表單分類策略: 提交查詢前之表單分析以及提交查詢後之表單分析。

在提交查詢前之表單分析這個階段，我們會先使用表單擷取器擷取出表單的結構特徵以及表單控制項的相關特徵以便進行後續的分析；為避免在提交查詢

後之表單分析階段對一個非查詢表單提交查詢，導致增加不必要的成本，我們會在提交查詢前之表單分析這個階段直接過濾特徵明顯的非查詢表單，(例如:使用者登入表單會包含密碼控制項)。

在提交查詢後之表單分析這個階段，主要是針對通過提交查詢前之表單分析階段的查詢表單進一步探測，確認是否為深網查詢表單。這個階段我們會對表單實際提交查詢，並從回傳結果判斷表單是否為深網查詢介面。為了對表單欄位填入適當的資料值以取得回傳結果，我們將表單欄位分成以下三類：

1. 關鍵字欄位允許輸入任意值。
2. 封閉式欄位則要求精確的值。
3. 區間欄位表示的是一段範圍區間的資料值，通常由兩個控制項所組成，分別表示起始資料值以及終止資料值。

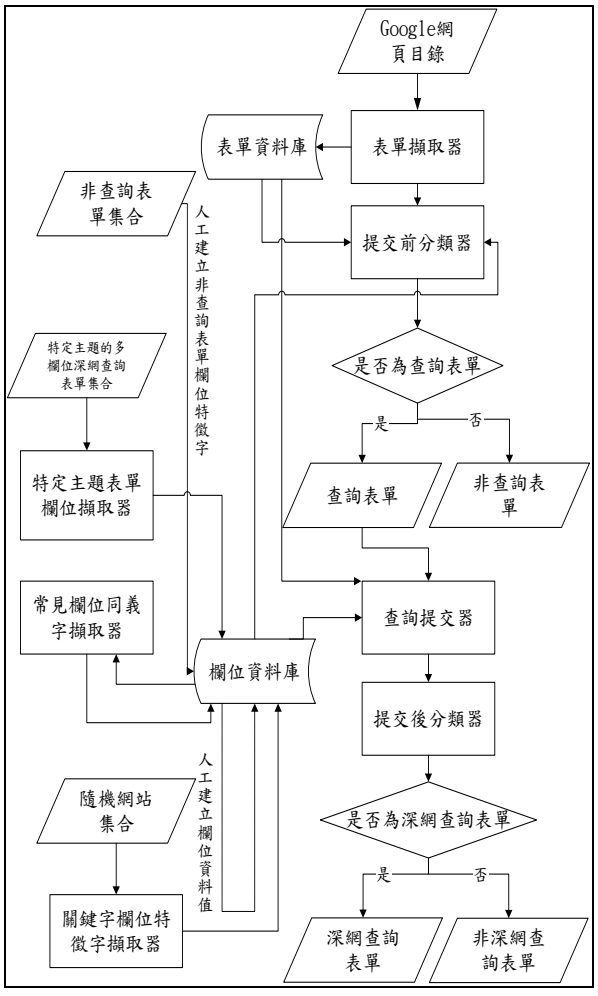


**圖 2 分類器系統架構圖**

各系統元件說明如下：

1. 特定主題常見欄位擷取器：負責從一個特定主題的深網查詢表單集合中，依據給定的門檻值 θ，找出出現頻率大於或等於 θ 的表單欄位，即為該主題的深網查詢介面中經常會出現的常見欄位。
2. 常見欄位同義字擷取器：負責至 Google 線上辭典查詢常見欄位的同義字資料。
3. 關鍵字欄位特徵字擷取器：負責大量爬行以關鍵字查詢為基礎的表單，並從表單中唯一的文字方塊控制項擷取 name 屬性作為特徵字，以提供其他元件識別關鍵字欄位時的依據。
4. 表單擷取器：負責從網頁中擷取出 HTML 表單，並分析、儲存表單的各項特徵，以及表單中所有控制項的相關特徵。
5. 提交前分類器：負責依表單與控制項的相關特徵將表單分為"查詢表單"與"非查詢表單"。
6. 查詢提交器：負責對 HTML 表單提交查詢，並儲存回傳結果。
7. 提交後分類器：負責分析查詢提交器取得的回傳結果，並判斷表單是否為深網查詢介面。
8. 表單資料庫：內容包括表單相關的結構特徵及表單控制項的相關特徵。
9. 欄位資料庫：內容包括關鍵字欄位特徵字、非查詢表單欄位特徵字、特定主題深網查詢表單欄位資料及特定主題常見欄位同義字資料。

## 4.2 前置準備子系統

### 4.2.1 建立非查詢表單欄位特徵字

為了在提交前查詢這個階段過濾非

查詢表單，除了使用表單本身的結構特徵進行判斷之外，我們更進一步使用非查詢表單特徵字對表單控制項執行細部的檢查。我們使用隨機網站產生器實際觀察上百個網站，從中收集非查詢表單。透過觀察，我們從這幾種常見的非查詢表單中擷取經常會出現的特徵字串作為非查詢表單特徵字，以供提交前分類器進行表單分類時的參考依據。

### 4.2.2 建立特定主題深網查詢表單欄位資料

對表單提交查詢時，經常會需要該表單所屬主題領域的相關知識，意即必須了解表單中各個欄位之語意。表單提交查詢的目的是為了協助我們進一步判斷表單是否為深網查詢介面，只須確保提交的查詢能順利取得回傳結果。

為了找出這些常見欄位，我們以人工方式收集特定主題的多欄位深網查詢表單，本研究在實驗的部分使用的是"Book"、"Job"這兩個主題，因此我們從Google 網頁目錄[4]分別輸入"bookstore"、"job"進行搜尋，最後，針對這兩個主題我們各別收集了 50 個表單。

接著，我們必須擷取出表單的標籤(label)。這裡我們將問題定義如下[11]。設 $L = \{l_1, l_2, \ldots, l_m\}$為表單 F 的標籤集合，且 F 包含控制項集合 $E = \{e_1, e_2, \ldots, e_n\}$，則標籤擷取的問題即是要找出每個控制項 $e_i$ 與標籤 $l_i$ 之間的對應關係集合 M。

對於相同主題的表單集合，我們分別將表單轉為 DOM tree (Document Object Model Tree, 文件物件模型樹)的結構，並以深度優先爬行的方式，依序將每個節點轉為(type, level, name, label)這樣的形式進行分析。其中，type 記錄的是節點的標記(tag)，以<tr>…</tr>這段原始碼為例，type 會記錄"tr"，但若該節點的標記為"input" (意即該節點為控制項)，則 type 會記錄控制項的 type 屬性，name 則記錄控制項的 name 屬性。另外，level 記錄的是該節點在整棵 DOM tree 中的階層，label 則是該節點的原始碼經過處理後所過濾出來的字串。完成轉換後，我們以 type 作為依據，擷取出表示文字方塊控制項以及下拉式選單控制項的節點，依序由這些節點往前搜尋包含 label 但尚未配對的非控制項節點進行對應，並以(name, label)的形式儲存找出的對應關係。以圖 3 為例，表單擷取出的對應結果如圖 4。

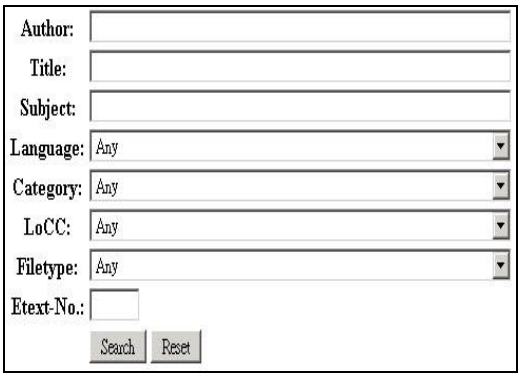


**圖 3 多欄位表單範例**


**圖 4 標籤對應結果**

將每個表單的標籤(label)擷取出來之後，我們會分別計算每個標籤在特定主題表單集合中出現的頻率，若頻率大於或等於我們設定的門檻值 θ，則該欄位即被視為是常見欄位。針對 Book 主題我們給定的門檻值 θ 為 0.5，取得的常見欄位為" title"、" author"、" isbn"；針對 Job 主題我們給定的門檻值 θ 為 0.3，取得的常見欄位為"category"、"country"、"location"、"job"、"type"。擷取出特定主題的常見欄位集合之後，我們以人工方式解讀欄位代

[4] http：//www.google.com.tw/dirhp

表的語義以及欄位之間的關係，並針對這些欄位分別建立資料，我們實際會建立多筆可能的欄位值提供查詢提交器參考，以表 1 中 id 為 3 的資料列為例，我們會實際填入一本書的 ISBN 碼。

擷取出特定主題的重要欄位集合之後，我們以人工方式解讀欄位代表的語義以及欄位之間的關係，並針對這些欄位分別建立資料。以"Book"這個主題為例，我們給定的門檻值 θ 為 0.8，意即 10 個表單中，若 8 個表單皆出現某個欄位，則該欄位即為"Book"主題的重要欄位。我們使用這個方式得到的重要欄位為 "title","author","isbn"，根據這個結果，即可分別針對這些欄位於資料庫中建立資料，我們實際會建立多筆可能的欄位值提供查詢提交器參考，以表 2 中 id 為 3 的資料列為例，我們會實際填入一本書的 ISBN 碼。

**表 1 特定主題深網查詢表單欄位資料表**

| id | 主題 | 標籤 | 資料值 | 父親 | 優先權 |
|---|---|---|---|---|---|
| 1 | Book | title | computer | -1 | 1 |
| 2 | Book | author | peter | -1 | 2 |
| 3 | Book | isbn | 0545139708 | -1 | 3 |
| 4 | Job | country | United States | -1 | 5 |
| 5 | Job | state | Illinois | 4 | 4 |
| 6 | Job | city | Chicago | 5 | 3 |
| 7 | Job | location | New York | 5 | 3 |
| 8 | Job | category | accounting | -1 | 2 |
| 9 | Job | job | designer | -1 | 1 |
| 10 | Job | type | contract | -1 | 6 |

### 4.2.3 擷取特定主題常見欄位同義字

在不同表單中，表單設計者有可能使用不同的標籤來表示相同語意的欄位，舉例來說，書籍作者這個欄位的標籤可能是"author"或"writer"等等，因此在欄位語意的識別上，我們特別考慮到同義字的問題。為了降低提交查詢的時間成本，對於每個常見欄位，我們事先使用常見欄位同義字擷取器對 Google 線上辭典填入表 1 中的標籤欄位值以提交查詢，並分析回傳結果，從中擷取出詞性為名詞的同義字儲存至同義字資料表，以便後續判斷欄位是否為特定主題之特定封閉式欄位。

### 4.2.4 擷取開放式欄位特徵字

在實際對表單提交查詢時，我們會優先對關鍵字欄位填值，以避免因有些欄位輸入的條件過於嚴苛，或是對具有特殊語義的欄位填入一個不適當的值而導致提交失敗(請參考 4.3.2.1 節)。然而，表單中經常存在多個文字方塊控制項提供使用者填寫，為了能夠從中識別出關鍵字欄位，我們設計了一個關鍵字欄位特徵字擷取器。為了使用關鍵字欄位特徵字擷取器抓取關鍵字查詢表單，以便能夠從關鍵字查詢表單擷取出關鍵字欄位特徵字，我們實際觀察關鍵字查詢表單所具備的結構特徵，根據這些特徵，我們設定的條件如下：

1. 下拉式選單控制項須小於或為 1 個。
2. 單選按鈕控制項須小於或為 1 個。
3. 多行文字方塊控制項須為 0 個。
4. 文字方塊控制項的數量須等於 1 個，且 size 屬性須大於 10。
5. 提交控制項的數量須等於 1 個，且 value 屬性為 "search"。
6. 複選按鈕控制項數量須等於 0 個。
7. 按鈕控制項須等於 0 個。
8. 密碼控制項須等於 0 個。
9. 重設控制項須等於 0 個。
10. 檔案控制項須等於 0 個。

我們由隨機網站集合中實際擷取出 10,000 個符合條件的表單，並人工檢視其中的 100 個表單，驗證是否為關鍵字查詢表單，結果顯示，100 個表單皆為關鍵字查詢表單。由於取得的關鍵字查詢表單僅有一個文字方塊控制項，而且該控制項必

為可輸入任意關鍵字的控制項，即我們所指的開放式欄位，因此我們擷取表單中唯一一個文字方塊控制項的 name 屬性作為開放式欄位特徵字，並統計特徵字串出現的次數，以便了解程式設計人員在開放式欄位的命名上是否傾向於使用某些特定的字詞。

透過爬行 10,000 個表單，我們一共取得 376 個關鍵字欄位特徵字，其中，出現次數前 10 名的統計結果如表 2。

**表 2 開放式欄位特徵字出現次數統計表**

| 出現次數排名 | 關鍵字欄位特徵字 | 出現次數 | 出現次數百分比 |
|---|---|---|---|
| 1 | q | 4527 | 45% |
| 2 | querytext | 1057 | 11% |
| 3 | query | 628 | 6% |
| 4 | s | 614 | 6% |
| 5 | p | 502 | 5% |
| 6 | search | 322 | 3% |
| 7 | keyword | 163 | 2% |
| 8 | search_block_form | 147 | 1% |
| 9 | searchword | 132 | 1% |
| 10 | keywords | 119 | 1% |
| ... | ... | ... | ... |
| 376 | searchinput | 1 | 0.01% |

## 4.3 分類子系統

在分類子系統，我們提出一個兩階段的分類架構，整合提交查詢前之表單分析(pre-query)與參考提交查詢結果之表單分析(post-query)，分成兩個階段過濾 HTML 表單。首先根據表單相關的結構特徵過濾掉非查詢表單，以避免對一個非查詢表單提交查詢，增加不必要的成本。接著再對所有的查詢表單實際提交查詢，並依回傳結果判斷其是否為深網查詢介面。

### 4.3.1 提交查詢前之表單分析

為了避免對非查詢表單提交查詢，增加不必要的成本，因此在提交查詢前之表單分析這個階段，我們會先根據擷取出的表單以及控制項特徵對表單進行分析，過濾出非查詢表單，剩下的查詢表單則繼續進入參考提交查詢結果之表單分析這個階段進行探測。

我們設計了一個表單擷取器，它會負責從網頁原始碼中擷取出表示表單的原始碼區段<form>…</form>，並且擷取表單以及控制項的相關特徵。

取得的特徵，除了可以在未提交查詢之表單分析這個階段提供提交前分類器作為分類上的依據，在參考提交查詢結果之表單分析這個階段，有許多特徵也是查詢提交器必須使用到的資訊。

我們實際觀察幾種常見的非查詢表單，發現部分的非查詢表單可以輕易由表單結構特徵識別出來。然而，有另外一部分的非查詢表單無法單純從表單結構特徵識別出來，必須進一步檢查表單控制項的相關屬性進行分析。因此，除了使用表單結構特徵進行判斷之外，我們也另外分析表單控制項的相關屬性是否包含非查詢表單欄位特徵字，藉此推測表單是否為一個非查詢表單。

在未提交查詢之表單分析這個階段，我們判斷的規則如下：

1. 若表單中密碼控制項的數量大於 0，則為非查詢表單。
2. 若表單中檔案控制項的數量大於 0，則為非查詢表單。
3. 若表單中多行文字方塊控制項的數量大於 0，則為非查詢表單。
4. 若表單中文字方塊控制項的數量等於 0，則為非查詢表單。
5. 對於每個表單控制項，我們分別擷取控制項的 id 屬性、name 屬性、value 屬性以及控制項的標籤，並檢查擷取

出的屬性值是否包含非查詢表單欄位特徵字，若是，則為非查詢表單，否則繼續進入參考提交查詢結果之表單分析階段執行探測。我們判斷某控制項屬性值 $C_{attr}$ 是否包含非查詢表單欄位特徵字的方法如下：

令 S 為非查詢表單欄位特徵字集合，$l$ 為字串長度變數(我們參考識別關鍵字欄位的作法將 $l$ 設為 3，詳情請參考 4.3.2.1 節)，若存在一個特徵字 $S_j \in S$ 滿足以下其中一個條件，則該控制項屬性值 $C_{attr}$ 即包含非查詢表單欄位特徵字。

1. $C_{attr}$ 字串長度大於等於 $l$，而且 $C_{attr}$ 是 $S_j$ 的子字串。
2. $S_j$ 字串長度大於等於 $l$，而且 $S_j$ 是 $C_{attr}$ 的子字串。
3. $C_{attr}$ 字串長度小於 $l$，而且 $C_{attr}$ 等於 $S_j$。

### 4.3.2 參考提交查詢結果之表單分析

在參考提交查詢結果之表單分析這個階段，我們實際對表單提交查詢，並從回傳頁面中擷取出查詢結果進行分析。

#### 4.3.2.1 提交查詢之處理原則

我們發現在填入表單欄位值時，無法直接從 HTML 原始碼看出表單控制項之間存在的聯結關係，必須依靠控制項的相關屬性以及控制項的位置來推測。

以圖 5 為例，Make 欄位表示汽車的廠牌，Model 欄位表示汽車的型號，兩者之間呈現一對多的邏輯對應關係，廠牌只會對應到某些特定的汽車型號，特定的汽車型號也僅會出自於特定的廠牌，意即廠牌欄位可被視為型號欄位的父親。填入這類欄位的欄位值時，我們必須注意填欄位值的先後順序並特別處理欄位之間的父子關係，因此，我們會檢查該欄位是否有對應到的父欄位，如果有，則必須檢查父欄位目前所填寫的資料值，並且對目前的欄位(即子欄位)填入與其父欄位相對應的值。

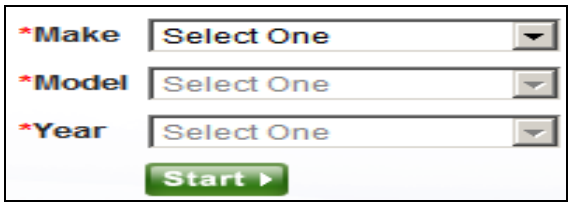


**圖 5 具備父子關係的控制項範例**

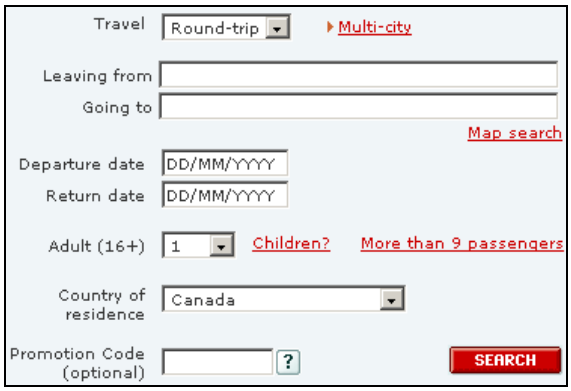


**圖 6 深網查詢介面範例**

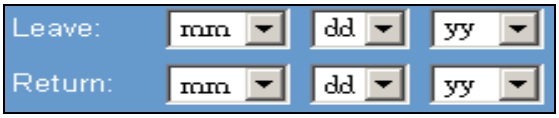


**圖 7 以下拉式選單呈現的區間欄位**

另外，以圖 6 中的 Departure Date 以及 Return Date 為例，這兩個欄位實際上是表示一段時間範圍的值，且 Departure Date 的時間值須小於 Return Date 的值，這類表示一段範圍資料值的區間欄位在實際填入欄位值時，須特別注意控制項前後的位置關係。若前面曾經出現零或偶數個這種日期時間型態的控制項，目前的控制項極可能須填入一個起始的時間值；反之，若出現奇數個這種日期時間型態的控制項，則目前的控制項極可能須填入一個終止的時間值。其他如最大、最小值等等，在填寫時也是使用類似的技巧。這類區間欄位也經常以下拉式選單的形式出現(如圖 8)，而成對的下拉式選單，其候選項目經常出現相似的文字內容，因此我們除了

以控制項的位置進行推測，也可透過比對欄位之間各候選資料項目中文字內容的相似度來推測是否為成對的下拉式選單。

#### 4.3.2.2 各控制項之填值策略

在填寫表單時，我們希望僅須填入能構成查詢的最少欄位所成集合，以避免太多不必要的欄位限制，造成無回傳任何查詢結果，影響後續的判斷。對一般使用者而言，部分控制項即使不填入值仍能順利提交查詢，然而本系統以程式自動提交查詢時，我們仍必須對表單中的每個欄位設值，以符合網站伺服器預期能夠處理的資料格式。

(1) 文字方塊控制項:

步驟 1: 優先尋找文字方塊控制項集合中是否存在關鍵字欄位。對於一個非關鍵字欄位而言，欄位本身通常包含標籤及其資料值兩個部份。以圖 8 中，標籤為 Title 的文字方塊控制項為例，這個封閉式欄位可以轉換成如圖 9 所示的兩個控制項，其中文字方塊控制項的值域受到下拉式選單控制項的限制。因此，在選擇填寫的欄位時，我們會優先選擇為關鍵字欄位的文字方塊控制項。

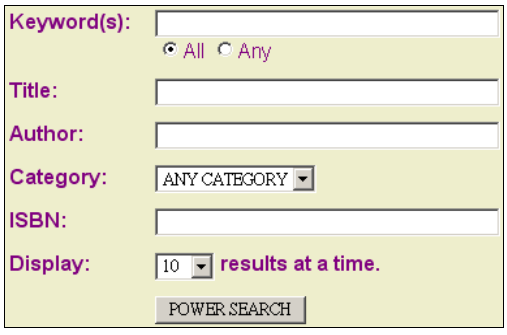


**圖 8 深網查詢介面範例**

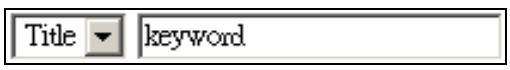


**圖 9 查詢條件轉換範例**

步驟 2: 若無任何關鍵字欄位，則繼續尋找文字方塊控制項集合中是否存在封閉式欄位，若是，則依前置準備階段所找到的特定主題深網查詢表單的常見欄位及其欄位資料值，按照欄位優先權選取封閉式欄位填寫。

(2) 下拉式選單控制項:

步驟 1: 觀察候選資料項目中的文字內容是否包含 "all", "any" 等字串，若是，則優先選取使得限制降到最低，否則，繼續執行後續步驟。

步驟 2: 若控制項為封閉式欄位，則從欄位資料表中選取資料值，比對與候選資料項目字串之間的相似度，並選取相似度最高的項目。另外，選取過程中須注意該欄位是否有對應的父欄位，以便選取一個適當的值。

步驟 3: 若控制項為區間欄位，則依控制項的位置判斷目前的欄位為起始欄位或終止欄位，並給予適當的值。一般來說，我們會盡量選取候選資料項中資料值最小的項目作為起始值，候選資料項中資料值最大的項目作為終止值，以求將限制降到最低。

步驟 4: 若上述三個步驟皆無法對此下拉式選單填值，則隨機選取一個不為空的值填入。

(3) 單選按鈕、複選按鈕控制項:

觀察候選資料項目中的文字內容是否包含 "all", "any" 等字串，若是，則優先選取使得限制降到最低，若否，則直接選取表單預設的項目資料值。

(4) 隱藏控制項:

直接使用隱藏控制項的預設資料值。

另外，為避免因表單欄位或是資料值選取不當，造成無任何回傳結果，對同一個表單，我們會提交多次查詢(目前預設為 3 次)，並儲存回傳結果，提供提交後分類器進行分析。

#### 4.3.2.3 擷取查詢結果

對表單提交查詢後，會取得內容為

HTML 原始碼的回傳結果。由於頁面中經常包含廣告、導覽列等相關的雜訊，因此，我們必須先將與查詢結果相關的部分從 HTML 原始碼中擷取出來。

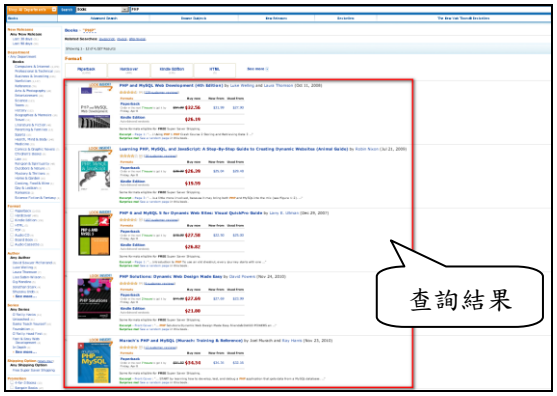


**圖 10 查詢回傳頁面**

我們發現查詢回傳頁面有以下幾點特性：

(1) 頁面經常反覆出現某些特徵樣式(pattern)。以圖 10 為例，由於頁面顯示 10 筆查詢結果，若將頁面轉為 DOM tree 的結構，會發現相同的子樹(subtree)重複出現 10 次。類似的情況也經常發生在導覽列或廣告連結列。

(2) 對表單進行多次提交查詢，其回傳的結果通常具備相似的頁面結構，其中不同的部分，主要是查詢結果或動態廣告的雜訊所造成的差異。

(3) 查詢結果區塊在開始或結束的位置經常會出現查詢結果訊息以及分頁資料列(如圖 11)。


**圖 11 查詢結果訊息及分頁資料列範例**

若要從散亂的 HTML 原始碼中直接看出哪一個範圍區段是與查詢結果相關的部分是非常困難的，因此，我們將回傳頁面解析成 DOM tree 的結構，使用結構化的方式進行分析。以圖 12 的 HTML 原始碼為例，我們會將原始碼解析成圖 13 中的 DOM tree。

```
<html>
    <head>
        <title>HTML DOM</title>
    </head>
    <body>
        <table>
            <tr>
                <td>test</td>
                <td><a href="#">link</a></td>
            </tr>
        </table>
    </body>
</html>
```

**圖 12 HTML 原始碼範例**

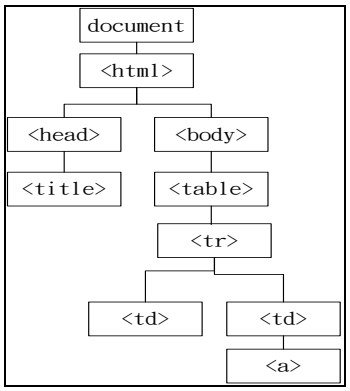


**圖 13 DOM tree 範例**

實際執行中，我們會以深度優先搜尋的方式從網頁原始碼中 body 這個位置開始爬行，並將每個節點儲存為(tag, level)這樣的結構，其中 tag 為節點標記，level 為整個 DOM tree 中節點所在的階層。以圖 12 為例，最後儲存的結果為如下陣列: ((body, 0), (table, 1), (tr, 2), (td, 3), (td, 3), (a, 4))。

### 4.3.2.4 分析查詢結果

運用經驗法則分析，以判斷是否為深網查詢表單回傳之查詢結果的流程如圖 14。

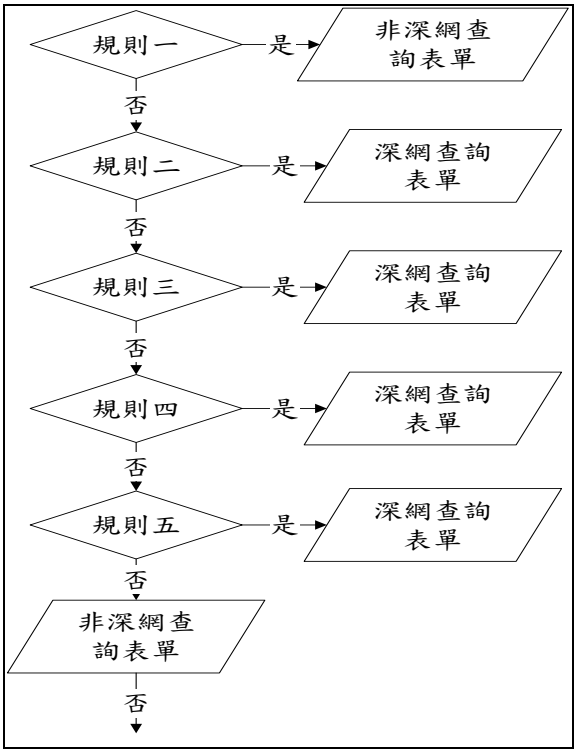


**圖 14：查詢結果判斷流程圖**

**規則一：**查詢結果中超連結的網域數量大於 5 個，則該查詢表單很可能為搜尋引擎，因此我們會直接判定為非深網查詢表單。

**規則二：**查詢結果包含圖片、文字方塊、按鈕等結構較複雜的控制項特徵。要注意的是，這些控制項出現的次數，必須大於或等於查詢結果的資料筆數，而且控制項出現在整棵 DOM tree 中的階層位置必須一致。由於我們無法直接從原始碼的結構判斷目前查詢結果顯示的資料筆數，因此我們的做法是從如圖 12 中的查詢結果訊息進行推測。

**規則三：**查詢結果包含 2 個(含)以上之資料欄位。我們檢查查詢結果中以"："結束之字串，並且分別計算這些字串的出現次數，若同時有 2 個(含)以上的字串其出現次數皆相同，則這些字串很有可能即為資料欄位名稱字串。

**規則四：**查詢結果包含多欄位資料表格。要判斷是否包含多欄位資料表格，必須先計算出資料行以及資料列的數量。表示資料列的原始碼區段可能為 … 或<tr>…</tr>，因此我們先從查詢結果中擷取出所有節點標記為<td>的節點，對於每個<td>節點分別往父節點爬行，並計算先爬行到 以及<tr>節點的數量，若先爬行到 節點的數量大於先爬行到<tr>節點的數量，則表示資料列是以 … 表達，反之則是以<tr>…</tr>表達。接著，繼續從查詢結果中擷取出所有表示資料列的節點，並計算每個表示資料列的節點包含<td>節點的數量，找出多數資料列包含的<td>數量，即為資料表可能的欄位數。對於具備相同欄位數的資料列，我們逐一分析其中的<td>節點是否包含欄位資料值，一般而言，欄位資料的字串數量通常不會太大，。

**規則五：**查詢結果包含搜尋引擎無法索引到的超連結集合。對於查詢結果中的超連結，我們先擷取超連結的網域、階層數、副檔名以及超連結所帶的參數名稱串列等資料。我們將查詢結果中之超連結集合，依照上述四項特徵之值是否完全相同，切割成數個子集合，接著使用最大子集合中的各個超連結即時對 Google 搜尋引擎提交查詢，若取得查詢結果，則表示 Google 已建立該超連結的索引資料，反之則否。若該子集合裡半數以上的超連結未被 Google 索引，則判定此表單為深網查詢表單，反之，則判定為非深網查詢表單。

總結上述說明，在這個階段，有別於以往研究僅是以回傳結果的資料長度進行判斷，我們實作的提交後分類器藉由分析查詢結果中的各項細部特徵，能以更精確的方式推測出表單是否為深網查詢介面。

## 5.實驗與討論

本實驗以"Book"與 "Job"這兩個主題領域為例，分別測試提交前分類器與提交後分類器對於識別一個 HTML 表單是否為深網查詢介面的精確度。實驗

環境如表 3：

**表 3 實驗環境**

| | |
|---|---|
| 作業系統 | Windows XP Professional |
| 中央處理器 | AMD Athlon(tm) 64 X2 Dual Core Processor 5200+ 2.71GHz |
| 記憶體 | 1GB |
| 網頁伺服器 | Apache 2.2 |
| 資料庫 | MySQL 5.0 |
| 程式語言 | PHP 5 |

## 5.1 前置工作

在過去的研究中，Chang 等人[6]將深網分為 8 個主題領域，並收集相關的查詢表單整理出一個資料集，但由於該資料集已有許多表單連結失效，因此我們自行收集表單作為測試資料集，方法如下: 我們進入 dmoz[5]網頁目錄下"Shopping "類別中的 "Books "分類，在該目錄以"book "作為關鍵字進行搜尋，另外進入 dmoz 網頁目錄下"Business"類別中的 " Jobs"分類，在該目錄以"job "作為關鍵字進行搜尋，最後分別依查詢結果收集取得 Book 主題以及 Job 主題的網站集合。接著，我們使用自行設計的網頁爬行器依序開始爬行，為了有效率地爬行，避免耗費太多時間成本在同一個網站，我們設計了兩個參數來控制爬行器的行為，level 參數負責控制爬行的深度，limit 參數負責控制於同一個網站中爬行的最大網頁數量，本研究將 level 參數設為 3，limit 參數設為 100。另外，要注意的是，我們僅爬行和網站相同網域的頁面，因此可避免受到廣告連結的誤導。爬行過程中若發生以下其中一種情況，則爬行器將會停止於該網站中繼續爬行：

1. 爬行器已擷取到 HTML 表單。
2. 爬行器已巡訪完所有深度為 level 層以內的頁面，但仍未找到 HTML 表單。
3. 爬行器巡訪網頁的數量已達 limit，但仍未找到 HTML 表單。

[5] http://www.dmoz.org/

使用上述方法，我們分別收集到 100 個 Book 主題以及 100 個 Job 主題的 HTML 表單作為測試資料集。

## 5.2 實驗結果

在提交前分類器的部分，我們將測試資料集分為三個群組，定義如下：

第一組：使用者可輸入的控制項數量小於 3 個。

第二組：使用者可輸入的控制項數量介於 3~5 個之間。

第三組：使用者可輸入的控制項數量大於等於 5 個。

**表 4 提交前分類器執行時間**

| 群組 | 表單數量 | 平均執行時間 |
|---|---|---|
| 第一組 | 57 | 2.2 msec |
| 第二組 | 68 | 3.3 msec |
| 第三組 | 75 | 2.8 msec |

在提交後分類器的部分，由於執行時間與查詢回傳結果檔的大小有關，此外，分析過程中我們也可能會對 Google 即時提交查詢，因此網路連線速度也可能影響執行時間。基於上述原因，我們僅取平均執行時間，並未分組測試。結果顯示，查詢回傳結果檔最小為 0KB，最大為 272KB，平均執行時間約為 7.75 秒。

本研究結合未提交查詢之表單分析與參考提交查詢結果之表單分析，在未提交查詢之表單分析階段，我們使用提交前分類器對表單進行分類；在參考提交查詢結果之表單分析階段，我們則使用提交後分類器對表單進行分類。

**表 5 提交查詢前分類結果(Book)**

| 未提交查詢之表單分析(主題：Book) | | |
|---|---|---|
| | 深網查詢介面 | 非深網查詢介面 |
| 須執行參考提交查詢結果之表單分析 | 31 | 12 |
| 判定為非深網查詢介面 | 0 | 57 |

**表 6 提交查詢前分類結果(Job)**

| 未提交查詢之表單分析(主題：Job) | | |
|---|---|---|
| | 深網查詢介面 | 非深網查詢介面 |
| 須執行參考提交查詢結果之表單分析 | 30 | 20 |
| 判定為非深網查詢介面 | 2 | 48 |

**表 7 提交查詢後分類結果(Book)**

| 參考提交查詢結果之表單分析(主題：Book) | | |
|---|---|---|
| | 深網查詢介面 | 非深網查詢介面 |
| 判定為深網查詢介面 | 27 | 1 |
| 判定為非深網查詢介面 | 4 | 11 |

**表 8 提交查詢後分類結果(Job)**

| 參考提交查詢結果之表單分析(主題：Job) | | |
|---|---|---|
| | 深網查詢介面 | 非深網查詢介面 |
| 判定為深網查詢介面 | 26 | 2 |
| 判定為非深網查詢介面 | 4 | 18 |

**表 9 表單分類結果(Book)**

| 主題：Book |
|---|
| Accuracy = 95 / 100 = 0.95 |
| Recall = 27 / 31 = 0.87 |
| Precision = 27 / 28 = 0.96 |
| Specificity = 68 / 69 = 0.99 |
| F-measure = 1.6704 / 1.83 = 0.91 |

**表 10 表單分類結果(Job)**

| 主題：Job |
|---|
| Accuracy = 92 / 100 = 0.92 |
| Recall = 26 / 32 = 0.81 |
| Precision = 26 / 28 = 0.93 |
| Specificity = 66 / 68 = 0.97 |
| F-measure = 1.5066 / 1.74 = 0.87 |

**表 11 搜尋引擎、站內搜尋及連結失效表單分類結果(Book)**

| | 搜尋引擎、站內搜尋表單數量 | 連結失效表單數量 |
|---|---|---|
| 判定為深網查詢介面 | 0 | 1 |
| 判定為非深網查詢介面 | 9 | 1 |

**表 12 搜尋引擎、站內搜尋及連結失效表單分類結果(Job)**

| | 搜尋引擎、站內搜尋表單數量 | 連結失效表單數量 |
|---|---|---|
| 判定為深網查詢介面 | 1 | 1 |
| 判定為非深網查詢介面 | 13 | 5 |

## 5.3 討論

在提交前分類器執行時間的部分，由於表單的相關特徵已由表單擷取器事先擷取出來，我們可直接使用表單相關的特徵資訊進行比對，並未有太多複雜的運算動作，因此執行時間明顯較短。而在提交後分類器執行時間的部分，由於我們必須在兩棵 DOM tree 之間爬行，對節點逐一進行比對，以擷取出可能包含查詢結果的原始碼區段，另外，還必須逐一分析頁面出現的字串，以推測查詢結果的回傳筆數，並找出頁面是否包含可能為資料欄位的字串，因此，若回傳頁面的資料內容越多或是回傳頁面的版面結構越複雜，則我們解析出來的 DOM tree 也會越大，在龐大的 DOM tree 之間爬行，可能導致整體的執行時間越長。此外，分析過程中若有必要也會對 Google 即時提交查詢，這中間的等待時間也是造成提交後分類器執行時間明顯較長。

觀察實驗結果，在未提交查詢之表單分析這個階段，我們發現若程式設計人員對表單控制項相關屬性給定疑似非查詢表單特徵字的字串，則我們會傾向於判定表單為非深網查詢介面，進而造成誤判。原因主要有以下四點：查詢回傳頁面包含過多的動態廣告雜訊、查詢結果資料筆數

過少、查詢提交器並未成功提交查詢、和查詢結果的資料欄位語意未明確標示。

這裡我們分別與 Cope 等人[7]以及 Bergholz 等人[2]提出的方法進行比較。

**表 13 未提交查詢之表單分析方法之比較**

| 特性編號 | 說明 | Cope 等人 | Bergholz 等人 | 本研究 |
|---|---|---|---|---|
| 1 | 是否可有效利用擷取出的表單特徵? | 否 | 是 | 是 |
| 2 | 是否可過濾非查詢表單? | 是 | 是 | 是 |
| 3 | 是否對表單實際提交查詢? | 否 | 是 | 是 |
| 4 | 是否對多欄位表單實際提交查詢? | N/A | 否 | 是 |
| 5 | 是否可過濾連結失效的查詢表單? | 否 | 是 | 是 |
| 6 | 是否可過濾搜尋引擎這類的非深網查詢表單? | 否 | 否 | 是 |

## 6.結論與未來展望

本研究提出一個兩階段的分類架構，結合未提交查詢之表單分析與參考提交查詢結果之表單分析，針對特定主題，發展一個自動化的深網查詢介面識別技術。在未提交查詢之表單分析這個階段，我們根據表單的結構特徵以及對表單控制項的相關屬性進行分析，可優先過濾出非查詢表單，以避免後續對一個非查詢表單提交查詢，增加不必要的成本。在參考提交查詢結果之表單分析這個階段，我們實際針對特定主題的查詢表單提交查詢，並從回傳頁面中擷取出可能包含查詢結果的原始碼區段，以進行各項特徵的細部分析，最後判定表單是否為深網查詢介面。實驗結果顯示，我們使用在兩棵 DOM tree 之間爬行進行比對的策略，可有效過濾大部分的雜訊，擷取出包含查詢結果的原始碼區段，對擷取出的原始碼區段進行更細部的特徵分析，除了可提升分類時的準確性，也可幫助我們進一步過濾搜尋引擎以及僅連結至靜態網頁的站內搜尋引擎，另外，對於連結失效的查詢表單也可進一步過濾。

本研究以自動識別特定主題 HTML 表單為主，對於少部分的非 HTML 表單(如：Java Applet, Flash) 以及客戶端程式碼(如：Javascript)尚未加以處理。此外，對於少數具備特定商業邏輯的表單(少於整體的 0.1%)[2]，也無法自動探測。

實驗的部分，由於我們須以人工方式判斷 HTML 表單實際是否為深網查詢介面，以驗證分類的正確性，因此，實驗的樣本數會有所限制。

本研究目前在特定主題表單控制項的欄位意義識別上，仍仰賴控制項的標籤，未來可以參考識別關鍵字欄位的方式，針對特定主題領域的查詢表單進行大量爬行，並觀察其中具備特定標籤的控制項，程式設計人員在對 name 屬性的設定上是否也傾向於使用某些特定字串，以幫助我們可以更精準地判斷控制項的欄位語義。

目前針對特定主題的封閉式欄位，由於欄位具備特殊意義以及可能有特殊的格式要求，我們仍須仰賴人工建立欄位資料庫作為給值時的參考，未來我們希望能夠發展出一個自動化的方式對這類欄位設值。相信這個議題會是一個很大的挑戰。

另外，我們提出的查詢結果擷取策略雖然可以有效過濾大部分的雜訊，然而當回傳頁面的資料內容越多或是回傳頁面的版面結構越複雜，則解析出來的 DOM tree 也會越大，在龐大的 DOM tree 之間爬行，可能導致整體的執行時間越長，未來我們也希望針對這個部分的效能進行優化。

## 參考文獻

[1] Barbosa, L. & Freire, J. (2004). Siphoning hidden-web data through

keyword -based interfaces. Proceedings of the 19th Brazilian Symposium on Databases (SBBD), pp. 309-321.

[2] Bergholz, A. & Chidlovskii, B. (2003). Crawling for domain-specific hidden web resources. Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE), pp. 125-133.

[3] Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine, Proceedings of the 7th International Conference on World Wide Web, pp. 107-117.

[4] Caverlee, J., Liu, L. & Buttler, D. (2004). Probe, cluster, and discover: focused extraction of qa-pagelets from the deep web. Proceedings of the 28th International Conference on Very Large Data Bases, pp. 103-114.

[5] Chakrabarti, S., van den Berg, M. & Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. Computer Networks and ISDN Systems, 31(11-16), pp. 1623-1640.

[6] Chang, K. C.-C., He, B., Li, C., Patel, M. & Zhang, Z. (2004). Structured databases on the web: observations and implications. SIGMOD Record, 33(3), pp. 61–70.

[7] Cope, J., Craswell, N. & Hawking, D. (2003). Automated discovery of search interfaces on the web. Proceedings of the 14th Australasian Database Conference, pp. 181-189.

[8] He, B., Li, C., Killian, D., Patel, M., Tseng, Y. & Chang, K. C.-C. (2006). A structure-driven yield-aware web form crawler: building a database of online databases. Technical Report, University of Illinois at Urbana-Champaign.

[9] He, H., Meng, W., Lu, Y., Yu, C. & Wu, Z. (2007). Towards deeper understanding of the search interfaces of the deep web. World Wide Web, 10(2), pp. 133-155.

[10] Hedley, Y.L., Younas, M., James, A. & Sanderson, M. (2004). A two-phase sampling technique for information extraction from hidden web databases. Proceedings of the 6th annual ACM international workshop on Web information and data management, pp. 1-8.

[11] Nguyen, H., Nguyen, T. & Freire, J. (2008). Learning to extract form labels. Proceedings of the VLDB Endowment, 1(1), pp. 684-694.

[12] Shu, L., Meng, W., He, H. & Yu, C. (2007). Querying capability modeling and construction of deep web sources. Proceedings of the 8th International Conference on Web Information Systems Engineering, pp. 13-25.

[13] Wu, P., Wen, J.-R., Liu, H. & Ma, W.-Y. (2006). Query selection techniques for efficient crawling of structured web sources. Proceedings of the 22nd International Conference on Data Engineering, pp. 47-47.