第三章

研究方法

本論文主要目的在於藉由具有類似查詢行為的使用者經驗，來自動化地幫助搜尋引擎生手更有效果地找出其所欲搜尋資訊。本研究所提出的方法總共分為兩大部份，第一部份針對長期情境，另一部份則針對短期情境。圖 3.1 為本研究的系統流程圖。

本研究所提出的方法，其流程概述如下：

(1) 在長期情境方面，本研究由查詢日誌著手，先進行 **User Session Identification**(使用者期間辨認)與 **Query Session Identification**(查詢期間辨認)，以找出每個使用者的每個 Query Session 中的所有日誌。接著，由每個 Query Session 的日誌中，針對以往使用者點選過的與未點選過的網頁或**網頁摘要**(Snippet)分別萃取出其**主題關鍵字**(Conceptual Keywords)。目的在將以往使用者感興趣與不感興趣的主題辨別出來，以判斷使用者的資訊需求。

(2) 在短期情境方面，對於搜尋引擎生手未點選的網頁摘要，也取出其主題關鍵字，主要想得知生手目前不想查詢的主題為哪些。

(3) 根據生手提供的短期情境線索，我們找出長期情境中有相似資訊需求的經驗使用者。經驗使用者與生手在查詢行為上往往有特性上的差異。我們首先利用這差異性，由長期情境中濾除生手使用者的查詢行為。接著，由以往經驗使用者之查詢行為中探勘出常出現的關鍵字集合。最後，提供使用者這些關鍵字集合，以自動地幫助使用者更有效果地搜尋資訊。

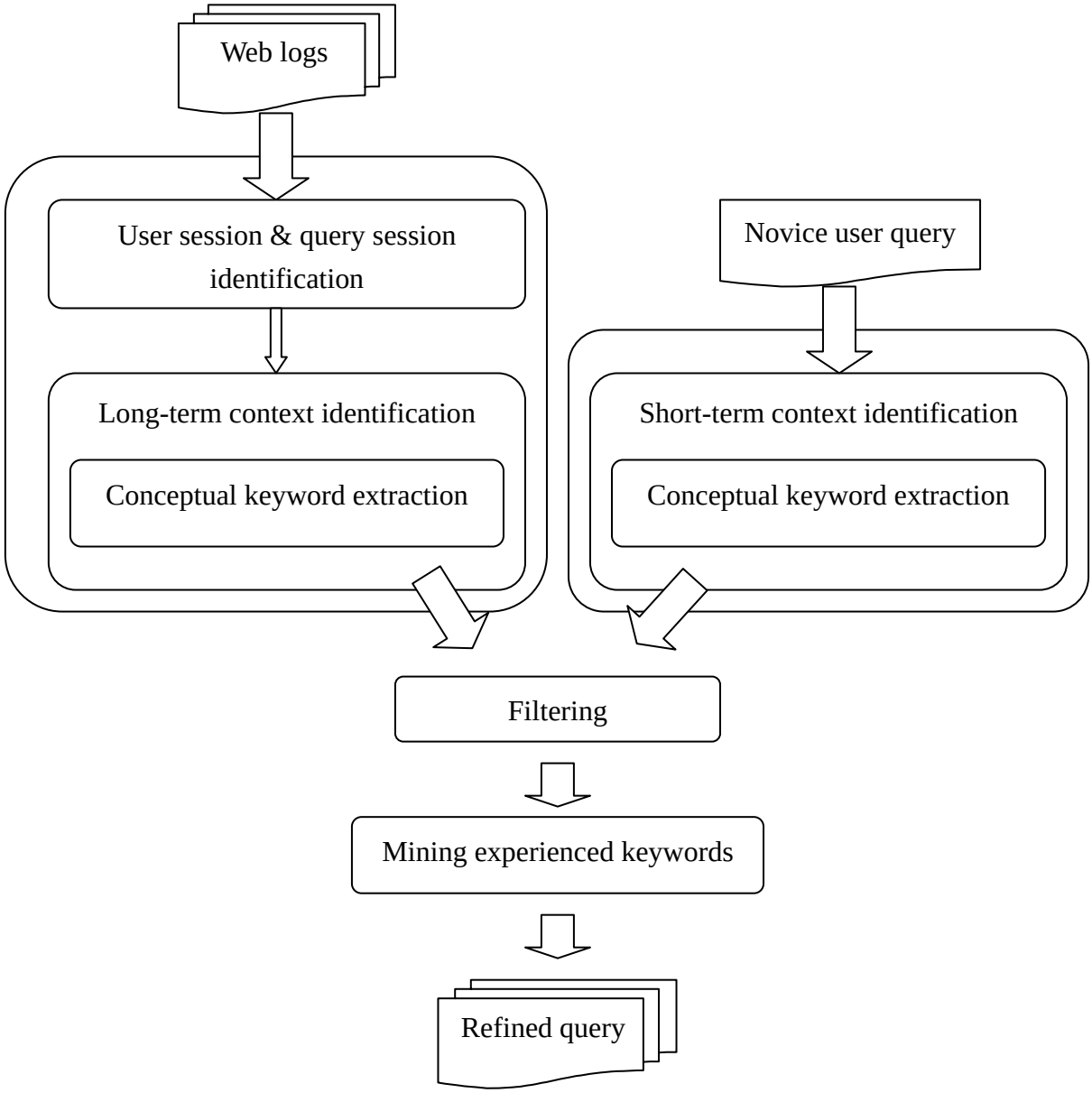

圖 3.1： 系統流程。

以下為長期情境與短期情境兩大部份中各步驟的詳述：

3.1 User Session and Query Session Identification

在長期情境方面，首先我們必須分析查詢日誌，以清楚地得知使用者的搜尋與瀏覽

行為。使用者在網站上的瀏覽行為，網站伺服器軟體都會記錄在網站日誌檔(Web Log File)中。網站日誌檔中的每一筆日誌記錄了使用者對網頁檔案的存取記錄。因此，藉由分析網站日誌的內容，便能夠得知使用者的瀏覽行為。

```
210.70.195.237 - - [13/Oct/1999:11:50:40 +0800]
"GET /~usr/index.php" 200 65536
"http://210.70.195.237/index.html"
"Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)"
```

圖 3.2： 網站日誌範例。

圖 3.2 為一筆網站日誌的範例，內容為使用者對網站發出 request 後的狀態記錄，其中記載了下列欄位：

1. IP address：使用者的來源，例如 210.70.195.237。
2. Timestamp：使用者發出 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. Request：此欄位為用戶端使用者在本次要求伺服器所傳送或回應的動作內容。例如使用者用 GET 方法來要求伺服器回傳使用者 usr 的 index.php 網頁，
4. Response：在 Request 之後是一組三位數字，其代表回應的狀態碼，例如 200 代表已經正確地回應了用戶端使用者的要求。
5. Size：在狀態碼之後的是傳送出的檔案大小，例如 65536 個 byte。
6. Referer：指向 request 的網頁，亦即此 request 的前一個 request。例如例子中前一個 request 為要求“http://210.70.195.237/index.html”網頁。若此 request 為第一個 request 或者是從別的網站所連入，則 Referer 記成 - 。

7. User-Agent：記錄使用者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。

當同一使用者由打開瀏覽器至關閉此瀏覽器的時間區間中，針對同一瀏覽主題所瀏覽的所有路徑稱之為一個 User Session。我們利用網站伺服器日誌中的相關資訊:IP address, Timestamp, Users-agent, Request, 及 Referer 等資訊來分辨不同的 User Session。本研究依序根據三項條件來辨別。首先將不同 IP，不同 User-agent 的使用者做分類。接著，相同使用者的日誌間，如果 Timestamp 差距太大就視為不同的 User Session。但是使用者可能同時打開多個瀏覽視窗，分別瀏覽不同主題的網頁。因此，我們利用 Referer 與 Request 欄位在前後日誌間的對應關係來判斷。我們以每筆日誌中的 Request 和 Referer 欄位形成 Access Pair。Access Pair 代表使用者在網站的整個瀏覽過程中的一步。而整個瀏覽的路徑便可由一連串 Access Pair 表示而成。其中，每對 Access Pair 的 Request 必需是下一對 Access Pair 的 Referer。

然而，由日誌中還原的 User Session 可能不是完整的。因為使用者瀏覽時可能會用 Backward 的方式回到前一網頁。因快取(Cache)或代理伺服器(Proxy)的原因，不須再向伺服器要求。所以要重建完整路徑必須自己補完這些遺失的 Backward Access Pair。而如何補完這些遺失的 Backward Access Pair，詳細的演算法則參考[5]，有非常詳盡的解說，但由於本研究所針對為查詢日誌，並不需要補完遺失的 Backward Access Pair，因此在此並不做詳盡之介紹。

接著，我們要介紹的是如何利用 Request 和 Referer 這兩個欄位來形成一連串的 Access Pair。若我們將一個 Access Pair 表示成 (a→b) 則一條 path 可表為 S：$(s_1 \rightarrow d_1)$, $(s_2 \rightarrow d_2)$, $(s_3 \rightarrow d_3) \cdots (s_n \rightarrow d_n)$，其中 $s_i = d_i - 1$, $1 < i <= n$。以下是一個使用者路徑(User Path)找尋的範例：

<例 1>如表 3.1 中的 8 筆 record 中：

表 3.1： 使用者路徑。

Record	IP	Timestamp	URL	Referrer	Agent
1	140.119.123.213	11:00:00	A	-	Mozilla 2.0
2	140.119.123.213	11:00:20	B	E	Mozilla 2.0
3	140.119.123.213	11:00:40	C	B	Mozilla 2.0
4	140.119.123.213	11:01:00	B	-	MSIE 3.0
5	140.119.123.213	11:01:20	C	B	MSIE 3.0
6	140.119.123.213	11:01:40	F	-	MSIE 3.0
7	140.119.123.213	11:02:00	B	A	Mozilla 2.0
8	140.119.123.213	11:02:20	G	B	Mozilla 2.0

Step 1：產生(- →A)，由於先前無 Referer，因此 User Session s1 將(- →A)加入 s1 之中。

Step 2：產生(E→B)，在現有的 User Session 中找含有 E 的 IP，且 agent 相同，timestamp 相近之 User Session，但是並沒有這樣的 User Session，所以產生一個新的 User Session s2 並將(E→B)加入 s2 中。

Step 3：產生(B→C)，發現 s2 中有(E→B)於是將(B→C)加入 s2 中。

Step 4：產生(- →B)，產生 s3 將(- →B)加入。

Step 5：產生(B→C)，加入 s3。

依此類推最後便產生了 s1：(- →A)，(A→B)，(B→G)，s2：(E→B)，(B→C)，s3：(- →B)，(B→C)，s4：(- →F)。

如此則可產生使用者瀏覽路徑，再根據 Request 欄位所記錄的查詢字串，則可以還原出

同一個使用者的連續查詢字串。

搜尋引擎也是一種網站伺服器。因此，使用者在搜尋引擎上的行為，包括關鍵字查詢、網頁存取與網頁連結的點選等動作也都會記錄在網站日誌中。搜尋引擎上的網站日誌，稱之為查詢日誌。圖 3.3 為查詢日誌的範例(為了清楚起見，這邊的查詢日誌範例，我們只留 Request 欄位，其餘欄位我們在此不列出)。例一為台灣奇摩搜尋引擎上查詢 data mining 的查詢日誌，例二為使用者點選 http://www.kdnuggets.com/這一個回傳網頁的日誌。

例一
http://tw.search.yahoo.com/search?fr=fp-tab-web-t&ei=UTF-8&p=data+mining

例二
http://tw.rd.yahoo.com/referurl/search/a/site/1/0/*http://www.kdnuggets.com/

圖 3.3： 查詢日誌範例。

經過 User Session Identification 後，可以得到使用者的網頁瀏覽與點選動作。接著，我們需從 User Session 中去找出使用者針對同一主題所做的查詢行為，也就是 Query Session Identification。所謂的 Query Session 是指使用者在搜尋引擎上，針對同一搜尋主題，由開始查詢一直到結束此次主題的搜尋，無論結果是有滿足資訊需求亦或者是放棄離開的搜尋，均稱之為一次的 Query Session。因此，我們需要從各個 User Session 中去鑑別出包含於其中的 Query Session。

前述所提出的 User Session Identification 的方法，並不適合直接套用於 Query Session Identification 的方法。原因是兩者所針對的方向並不同，要真正正確地做到 Query Session

Identification，必需實際考慮使用者的查詢意圖，來確定一段查詢過程的起始與結束。

一般最簡單找出 Query Session 的方法即是以使用者點選網頁的動作來做判定，但這樣的結果以及效能都相當的不理想，原因在於，單單使用這樣的方法並未考量到使用者搜尋的主題以及意圖，因此，本研究提出了結合查詢時間、查詢內容，以及使用者點選動作等等特徵的方法，來辨認使用者的查詢意圖，進而判斷出其 Query Session。

我們首先說明如何找出各別的 Query Session：

(1) 定義合理時間範圍

我們定義兩個時間差的門檻值，分別為時間最大門檻值(Time_maximum_threshold)與時間最小門檻值(Time_minimum_threshold)。當兩個查詢日誌之間的時間區間超過時間最大門檻值時，即視為不同 Query Session。而當兩個查詢日誌之間的時間區間小於時間最小門檻值時，則視為同一 Query Session。

(2) 計算兩查詢之間的相似度

如果兩個查詢日誌之間的時間區間介於時間最大門檻值與時間最小門檻值之間，則計算此兩查詢的相似度。首先會先定義一個相似度門檻值(Similarity_Threshold)，如果相似度高於定義的相似度門檻值時，則視兩查詢為同一 Query Session。而在此處的相似度，本研究定義為查詢關鍵字字串查詢結果中，回傳網頁的相似度。我們利用餘弦相似度測量(Cosine Similarity Measure)來幫助我們解決此處的相似度計算，餘弦相似度測量原本是依照兩篇文章出現的關鍵字向量來計算兩文章之間的相似度。同樣的，本研究以回傳結果前一百筆網頁為限，分別對兩查詢前一百筆的回傳網頁萃取關鍵字，將前一百筆的回傳網頁當成一篇文章來看待，一樣的以出現的關鍵字向量來做計算，去算出兩查詢間的餘弦相似度(Cosine Similarity)。

(3) 檢查使用者對於查詢結果的點選行為

如果兩個查詢間的相似度低於相似度門檻值，則再檢查兩個查詢間是否有點選回傳

網頁的動作。在一般情況下，若使用者有找尋到其資訊需求時，最後的動作應是點選且瀏覽所需要的網頁，因此在這部份，一般認為，在一個正常 Query Session 的結束應是以點選網頁的動作來做結束。也因此，若兩查詢的相似度低於我們所定義的相似度門檻值，則再檢查兩查詢間是否有點選回傳網頁的動作，若沒有的話，本研究將其視為同一個 Query Session。而如果有的話，則將其視為不同的 Query Session。

3.2 Implicit Feedback Identification

使用者的搜尋行為當中，不只從使用者的查詢關鍵字可以看出使用者的資訊需求，從使用者點選過的網頁 **(Clickthrough Data)** 甚至是使用者未點選的網頁 **(Non-clickthrough Data)**，同樣地也可以提供使用者查詢需求的線索。所以，查詢關鍵字、使用者點選過的網頁與使用者未點選的網頁都是跟使用者的搜尋目標息息相關的。不只如此，一般來說，使用者做搜尋最終的目標不會是關鍵字，而是網頁。網頁所傳達的資訊比一個關鍵字所可以傳達的資訊還多，而且比較具體、清楚。而如果可以知道目前使用者的查詢與哪些網頁的關連程度非常密切，那麼，使用者給予這些查詢背後的資訊需求，是不是就可以從這些網頁裡得到呢？而得知使用者查詢與網頁的關聯程度後，是不是也會對搜尋引擎生手有很大的幫助呢？

因此，本研究所想利用的隱含回饋資訊包括三類。第一類為使用者的查詢關鍵字。查詢關鍵字是最直接也是最明確地代表使用者資訊需求的資訊；第二類為使用者點選的網頁資訊。使用者面對搜尋系統回傳頁面所點選的網頁資訊，是相當重要的線索。使用者所點選過的網頁與其資訊需求是非常密切相關的，基於這個理由，以往研究[2，22]均顯示出使用者點選的網頁資訊提供了相當強而有力的隱含回饋線索。最後一類則是使用者未點選的網頁資訊，與上述理由相同，藉由使用者未點選的網頁資訊，我們可以推估使用者不感興趣的主題。

除了網頁資訊外，**網頁摘要**(Snippet)在搜尋行為分析扮演重要的角色。一般的搜尋引擎在回傳查詢結果時，每個結果網頁都是以網頁摘要的形式來表示。網頁摘要擷取出網頁中出現查詢關鍵字的主要段落。藉由網頁摘要，使用者可以大致了解網頁的內容，以判別此網頁是否符合其資訊需求，進而決定選擇是否點選瀏覽。因此，使用者在搜尋的過程中，往往是根據網頁摘要的內容來做網頁點選的決定。

使用者每次的查詢過程是由數個查詢回合所構成的。一個查詢回合的動作包括使用者所下的關鍵字查詢及針對回傳結果所做的後續動作。因此，我們可以將長期情境與短期情境中的一個 Query Session 表示成$\{Q_1, C_1, N_1\}\rightarrow\{Q_2, C_2, N_2\}\rightarrow\ldots\ldots\rightarrow\{Q_m, C_m, N_m\}$，其中 Q_i, $1 \le i \le m$, 代表使用者在第 i 次查詢回合中，在搜尋引擎所下的關鍵字查詢; C_i 與 N_i, $1 \le i \le m$, 分別代表使用者在第 i 次查詢回合中，對於搜尋引擎回傳結果所點選的網頁(或網頁摘要)與所沒有點選的網頁(或網頁摘要)。

3.2.1 Conceptual Keyword Extraction

為了透過隱含回饋辨識使用者的資訊需求，我們首先必需分析搜尋引擎回傳的網頁或網頁摘要的的內容，分析其主題關鍵字。

(1)網頁主題關鍵字：一般來說，網頁的內容絕大部分是由多段文字或影像所構成，在本研究中，著重的是在網頁內容的文字部份。因此，為了要了解網頁內容的真正含義，必須從網頁所包含的關鍵字著手。

一般來說，網頁就像一般的文章：越能傳達作者真正意義的關鍵字，越有可能以重要的形式來區隔與其餘關鍵字的不同。例如：一篇文章的標題一定是整篇文章的靈魂所在。因此，在網頁的編輯上，作者會根據他所想傳達給讀者的真正含義用不同的格式來表達。本研究由四種文章格式來萃取關鍵字，以下詳述：

(a)全文檢索：針對網頁內容的文字部分，我們先用斷詞的方法做斷詞的動作，將這

些斷出來的詞去除 Stop Words(非查詢用語)後，所產生出來的結果即為全文萃取出的關鍵字集合，並且記錄下關鍵字的出現次數。

(b) **超連結文字** (Anchor Text)[7]：以往的論文均顯示，超連結文字可以很精確的代表出網頁作者想傳達給讀者的真正意義。因此，超連結文字是本研究萃取的重點之一。

(c)標題：就像傳統文章一般，標題通常是整篇文章所要表達的精神所在，網頁也不例外。

(d)粗體字：就像學術論文一般，直觀而言，粗體字給讀者的感覺就是重要且可令人加深印象的。

由以上方法所論述，本研究可以萃取出各種網頁中所有的關鍵字。緊接著，在萃取完網頁所有的關鍵字後，接下來要做的是，給定每個關鍵字的權重，以找出最符合網頁意義的關鍵字。本研究結合兩種方法來給定每個關鍵字的權重值。以下闡述這兩種方法：

(a)出現次數(Frequency)：

直觀而言，一個關鍵字出現越多次於網頁中，表示在這個網頁中，這個關鍵字是越重要的。因此，本研究會去計算各個關鍵字的出現次數，出現越多次的，則給予較高的權重。

(b)不同的 HTML 格式：

依照上述說明的原因，我們可以知道，不同的 HTML 格式萃取的關鍵字，所代表的真實意義是有差別的，因此，根據剛才所使用的條件：超連結文字、標題、粗體字這三種格式萃取出的關鍵字則給予較高的權重值。而本研究在這三種格式上權重高低比例分配上分別為超連結文字> 標題 > 粗體字。

因此，針對一個網頁，將所有的關鍵字萃取出來，依照權重值高低作排序後，本研究取權重值前十高的關鍵字，定義為這個網頁的主題關鍵字。

(2)網頁摘要主題關鍵字：網頁摘要不像網頁一樣有多樣的 HTML 格式可供參考，因此我們僅以出現次數最高的前十名關鍵字來表示網頁摘要的主題關鍵字。

3.2.2 Long-term Context Identification

一般來說，使用者在搜尋引擎上針對同一搜尋主題進行搜尋時，都會歷經多次回合的查詢，直到資訊需求已滿足或使用者放棄搜尋。也因此，在最後一回合之前的查詢回合中，使用者所點選的網頁可能與其搜尋需求相符，也可能與其需求不相關。那麼，我們要如何從使用者所點選的資訊中去獲得我們所需的情境線索呢？於此，網頁摘要扮演了一個相當重要的角色。大部分的使用者都是閱讀了回傳結果的的網頁摘要，根據網頁摘要與其資訊需求的關連性，來做網頁點選與否的判斷。因此，網頁摘要在此比網頁提供更有利的線索。相反的，在假設有經驗的查詢者都搜尋到所需資訊的前提下，使用者在最後一回合所點選的網頁資訊就提供了有利的線索。

基於上述之理由，在長期情境的隱含回饋中，針對一個使用者的 Query Session，本研究會記錄的是，非最後一次查詢回合中，使用者所點選的網頁摘要資訊，以及最後一次查詢回合中，使用者點選的網頁實際資訊。

根據 3.2 節一開始所述，本研究將一個 Query Session 記成$\{Q_1, C_1, N_1\}\rightarrow\ldots\ldots\rightarrow\{Q_m, C_m, N_m\}$，經由 3.2.1 節萃取主題關鍵字後，長期情境的隱含回饋資訊則紀錄如下，$\{Q_1, CK_1\}\rightarrow\ldots\ldots\{Q_{m-1}, CK_{m-1}\}\rightarrow\{Q_m, CK_m\}$，同樣的，$Q_i, 1 \le i \le m$ 代表了在第 i 次查詢回合中，使用者在搜尋引擎所下的查詢關鍵字集合；$CK_i, 1 \le i \le (m-1)$則代表了第 i 次查詢回合中，使用者點選的網頁摘要的主題關鍵字；而 CK_m 則是第 m 回合(最後一次查詢回合)使用者點選的網頁的主題關鍵字。

3.2.3 Short-term Context Identification

在上一小節中，對於如何取得本研究所需要的長期情境資訊已經相當清楚了，而本小節則繼續談到有關於短期情境的隱含回饋資訊。

上一小節中，有談到在長期情境的部分，一般認為，對於能夠察覺使用者資訊需求有幫助的資訊，是最後一次查詢回合中，使用者點選的網頁實際資訊，以及非最後一次查詢回合中，使用者所點選的網頁摘要資訊。但是，長期情境中，使用者的一次查詢過程是完整的，因此可以用上述的角度來獲得長期情境的資訊。而短期情境中，由於目前使用者尚未滿意其資訊需求，才會求助於系統提供自動查詢修正的功能。所以，我們將長期情境資訊與短期情境資訊以不同的角度來獲取隱含回饋資訊。在下一段中，我們將詳述如何取得短期情境的資訊。

由於長期情境中，使用者的查詢過程是完整的，因此本研究在查詢過程的最後一次查詢回合採用了網頁實際內容的資訊。不過既然目前使用者並未滿足其資訊需求，那麼，網頁實際內容的資訊則不符合使用於短期情境，所以，在短期情境中，我們將充分利用網頁摘要資訊來獲得短期情境的隱含回饋資訊。另外一方面，由於本研究所針對的使用者族群為搜尋引擎生手，搜尋引擎生手在搜尋的過程中最大的問題即在於他們並不懂得如何去給予最合適於其資訊需求的查詢關鍵字，以致於搜尋效果不彰，也基於這個原因，本研究認為，既然搜尋引擎生手不知如何查詢適當的關鍵字，那麼，搜尋引擎根據生手的查詢關鍵字所回傳的網頁也會與生手的資訊需求相關程度較低，在較低相關程度的回傳網頁中找尋，自然而然的也就不能夠期待目前使用者能夠點選到很相關於其資訊需求的網頁，而對本研究取得短期情境資訊有所幫助。因此，在取得短期情境資訊這部分，本研究決定不使用使用者點選過網頁的資訊，而改以使用使用者未點選的網頁資訊。一般來說，搜尋引擎生手只是不知如何查詢適當的關鍵字，但肯定非常清楚自己的

資訊需求。換個角度而言，我們可以經由使用者未點選的網頁來獲得使用者不感興趣的資訊，而藉以刪除使用者不感興趣的主題，來慢慢地趨近使用者的資訊需求，這不正好也是另外一種反向思考模式的查詢修正嗎？因此，「使用者點選的網頁資訊未必與其搜尋需求相關，但沒點選的網頁資訊很可能與其搜尋需求不相關」，正是我們取得短期情境的隱含回饋資訊的最主要之立足點。

基於上述之理由，關於短期情境的隱含回饋資訊，我們將從目前使用者所提供的搜尋行為中，去針對目前使用者每一次查詢回合中未點選的網頁摘要資訊，來當作本研究取得短期情境資訊的重要考量。

本研究中短期情境的資訊紀錄為$\{Q_1, NK_1\}\rightarrow\ldots\ldots\rightarrow\{Q_m, NK_m\}$，同樣地，$Q_i, 1 \le i \le m$ 代表了在第 i 次查詢回合中，使用者給予搜尋系統的查詢關鍵字集合；而 $NK_i, 1 \le i \le m$ 則是第 i 回合中使用者沒有點選的網頁摘要的主題關鍵字。

3.3 Filtering

在 3.1 節與 3.2 節中，已經將本研究需要的隱含回饋的資訊，包含長期情境與短期情境，做了完整的介紹。因此，在本節中，在本節中，針對短期情境資訊，我們將說明如何萃取出有助於目前使用者的長期情境資訊。在此共有三個步驟。首先，根據目前使用者的查詢關鍵字，由長期情境中找出相關的 Query Session。接著，將查詢生手的 Query Session 濾除。最後，濾除**同型異義**(homograph)的 Query Session。每個步驟詳述如下：

一般使用者的查詢行為中，最能表達使用者的資訊需求即是使用者的關鍵字查詢，那是使用者對於其資訊需求所做最直接的判斷，以及根據搜尋引擎回傳結果所再修正後的人為線索判斷。也就是說，使用者的關鍵字查詢提供了搜尋引擎一個關於其資訊需求的大方向，而本研究同樣地也根據使用者的關鍵字查詢，來判別長期情境與短期情境是

否互相具有關連。因此，在第一個步驟中，我們先根據使用者的查詢關鍵字，由長期情境資訊中找出相關的 Query Session。根據目前使用者的關鍵字查詢{Q_1, ……, Q_m}，我們從長期情境眾多 Query Sessions 中，去找出有包含{Q_1, ……, Q_m}關鍵字集合中一個或一個以上的關鍵字的 Query Sessions，而這些長期情境的 Query Sessions，本研究即將其當成與短期情境有關連，記成 QS。

在第一個步驟中，我們已經經由查詢關鍵字的比對，來找出長期情境中與短期情境資訊相關的 Query Sessions，而誠如大家所知悉的，查詢日誌是在一段時間中，眾多使用者在搜尋引擎上搜尋所留下的紀錄，因此，以往不管是經驗使用者亦或者是搜尋引擎生手均會留下其查詢行為之紀錄於查詢日誌中。然而本研究所針對的對象便是搜尋引擎生手，最主要的的目的也就是希望藉由以往經驗使用者的搜尋行為來幫助搜尋引擎生手，使其能更快、更準確地找尋到其資訊需求。由此來看，在長期情境的資訊中，我們需要的是經驗使用者的搜尋行為，但是在長期情境所有的 Query Sessions 中，是包含所有不同種類的使用者的搜尋行為，也因此在第二步驟中，必須將以往屬於搜尋引擎生手搜尋行為的 Query Sessions 予以刪除，保留下以往屬於經驗使用者搜尋行為的 Query Sessions。

根據搜尋行為的相關研究[11]指出，經驗使用者與搜尋引擎生手在搜尋行為上有顯著的不同。首先，在搜尋過程中，搜尋引擎生手在兩查詢回合間傾向於點選較少的網頁。此外，在搜尋過程中，搜尋引擎生手會很頻繁地變換所查詢的關鍵字，並且經常更換查詢關鍵字中的部分關鍵字，且這些變換都無助於其資訊需求。

因此，由以上所述，若能根據這兩種以往研究所指出的搜尋行為不同點去做判斷，那麼，則可以大致地判斷出長期情境中經驗使用者與搜尋引擎生手的 Query Sessions 各為哪些。根據上述的第一個不同點，本研究認為在一個 Query Session 當中，若是存在

著較少數目的使用者點選網頁，則可以判定為屬於搜尋引擎生手的 Query Sessions。而根據第二個不同點，則是越多查詢關鍵字次數存在於一個 Query Session 中，將判斷為屬於搜尋引擎生手的 Query Sessions。所以，在本步驟中，本研究依據上述的兩項搜尋行為的觀察，分別提出兩個指標衡量來判斷生手。第一個指標是查詢關鍵字密度 *qd*，

$$qd = \frac{m}{m + cn}$$

其中 *m* 代表查詢回合數、*cn* 代表點選網頁個數。換句話說，查詢關鍵字密度越高越可能是搜尋引擎生手的 Query Session。第二個指標是點選網頁密度 *cd*，

$$cd = \frac{cn}{m + cn}$$

換句話說，點選網頁密度越低越可能是搜尋引擎生手的 Query Session。綜合以上所述，高於查詢關鍵字密度門檻值且低於點選網頁密度門檻值的 Query Session，我們則將其判定為搜尋引擎生手的 Query Session。我們將查詢關鍵字密度門檻值、點選網頁密度門檻值分別設定為長期情境中所有 Query Session 的查詢關鍵字密度之平均值、點選網頁密度之平均值。

由於前兩步驟已經大略地取得了與短期目前使用者資訊有關連的長期情境資訊，因此在最後一個步驟裡，本研究想要再根據目前使用者所提供的有用線索，更進一步地挑出長期情境中符合目前使用者資訊需求的 Query Session，則可獲得與目前使用者有類似查詢行為之經驗使用者的查詢行為。

而目前使用者所提供的有用資訊，則是使用到「**同型異義字**」(**homograph**)的概念，何謂「同型異義字」呢？顧名思義，就是同樣的一個字，有著多種的意思。在第一章的例子中，“Jordan”本身就屬於同型異義字，在此，“Jordan”至少有著三種意思，美國職業

籃球員 Michael Jordan，中東國家 Jordan，以及數學線性代數中的 Jordan Form，由此來看，一個字有著至少兩個以上的意義存在，我們則稱其為「同型異義字」。那麼同型異義字該怎麼使用到本步驟的做法呢？在前兩步驟中，本研究已經找出了包含目前使用者查詢關鍵字的以往經驗使用者的 Query Sessions，但是卻還不能夠肯定這些 Query Sessions 與目前使用者的資訊需求相符。以第一章的例子而言，假如目前使用者想查的為 NBA 籃球員 Michael Jordan，但是以往經驗使用者的 Query Sessions 卻有包含搜尋 NBA 籃球員 Michael Jordan，中東國家 Jordan，以及數學線性代數 Jordan Form 等等資訊需求，因此，若我們能將包含目前使用者未點選網頁的網頁摘要之主題關鍵字的以往經驗使用者 Query Sessions 予以去除，那麼，所剩下來的即是與目前使用者資訊需求相符之以往經驗使用者的 Query Sessions。

因此，我們利用前述短期情境$\{Q_1, NK_1\}\rightarrow\{Q_2, NK_2\}\ldots\ldots\rightarrow\{Q_m, NK_m\}$中的 NK_1, NK_2, …,NK_m。如果經驗使用者的 Query Session 中，包含目前使用者未點選網頁摘要之主題關鍵字，則予以濾除。因為這些可能是與目前使用者查詢屬於同型異義關係的 Query Session。

3.4 Mining Experienced Keywords

前一節中，本研究依據短期情境的資訊，從長期情境中找出了與目前使用者的資訊需求相符合的以往經驗使用者之查詢行為，接著，我們針對這些經驗使用者的查詢行為去做探勘分析，目的即是想要探勘出這些具有與目前使用者相類似資訊需求的有經驗使用者，他們的查詢行為中大多具有哪些關鍵字。而這些探勘出的關鍵字，也正是可以利用來幫助目前使用者的修正查詢。

我們將前述的每筆長期情境資訊$\{Q_1, CK_1\}\rightarrow\ldots\ldots\{Q_{m-1}, CK_{m-1}\}\rightarrow\{Q_m, CK_m\}$表示成

集合 $Q_1 \cup CK_1 \cup \ldots\ldots \cup Q_m \cup CK_m$。換句話說，我們將每個與目前使用者查詢相關的經驗使用者 Query Session 表示成關鍵字的集合。而這集合是由經驗使用者的查詢關鍵字和點選網頁摘要(或網頁)的主題關鍵字所組成。

我們利用資料探勘中關聯法則探勘(Association Rule Mining)的**頻繁項目集探勘**(Frequent Itemset Mining)，由這些集合中探勘出常共同出現的關鍵字集合。

值得注意的是，本研究使用的探勘模式為頻繁項目集探勘，而非使用循序樣式探勘(Sequential Pattern Mining)。一般認為，搜尋行為是一連串使用者與搜尋引擎互動的過程所構成的，也是一連串有順序的行為所構成，因此，直觀來看，利用循序樣本探勘會是比較合理的。但我們認為，使用者的資訊需求是相當主觀的，即使兩個有著完全相同資訊需求的使用者，只要他們一開始的想法不同，切入搜尋的角度不同，在在的都會造成之後搜尋行為的差異。正因為如此，使用循序樣式探勘並不一定能完全地探勘出具有相同資訊需求使用者的查詢行為，因此，本研究不採用循序樣式探勘，而採用頻繁項目集探勘。目的即是以使用者整個的查詢過程中重要之關鍵字為一組項目集，不受到順序性的限制，如此，即可探勘出具有相同資訊需求之以往使用者其搜尋過程中重要的關鍵字集合，藉以回傳給目前使用者，達到查詢修正之目的。

依照上述之方法，將會得到與目前使用者具有相同資訊需求之以往使用者其搜尋過程中重要的關鍵字集合，而這些關鍵字集合，很可能不只一組，如何選定最適合目前使用者的那一組關鍵字集合，回傳給使用者呢？我們利用**支持度**(Support)及關鍵字字數兩個條件來做篩選。針對多組的關鍵字集合，本研究會先用支持度來做過濾條件，過濾完剩下仍不只一組時，再繼續使用關鍵字字數的多寡來做判定。因此，當多組關鍵字集合均超過最小支持度時，即取最多支持度的關鍵字集合，當成回傳的修正查詢。而若最多支持度不只一組關鍵字集合時，則再利用第二個過濾條件關鍵字字數來做篩選。在研究

搜尋行為的相關論文[11]曾指出，有經驗的使用者在一次的關鍵字查詢中，所給予搜尋引擎的關鍵字字數是較多的，並且，一次查詢中包含較多的關鍵字字數，直觀來看，也是可以獲得較完整及完善的回傳網頁。因此，利用這兩種條件，即可挑選出最適合目前使用者的關鍵字集合，以此來當作回傳的修正查詢。

從 3.2 節到本節所詳述的方法，是本研究針對一位搜尋引擎生手所做一次查詢修正的過程，從本系統萃取其搜尋行為資訊一直到回傳修正後查詢予使用者。當第一次過程的查詢修正後，若使用者仍感受到沒有符合其資訊需求，則我們會持續下去，重複地做 3.2 節到本節的方法，直至滿足使用者的需求為止。