Sampling, Information Extraction and Summarisation of Hidden Web Databases

Yih-Ling Hedley [a], Muhammad Younas [b], Anne James [a], Mark Sanderson [c]

[a] *School of Mathematical and Information Sciences*
Coventry University, Coventry CV1 5FB, United Kingdom
[b] *Department of Computing*
Oxford Brookes University, Oxford OX33 1HP, United Kingdom
[c] *Department of Information Studies*
University of Sheffield, Sheffield, S1 4DP, United Kingdom

Abstract

Hidden Web databases maintain a collection of specialised documents, which are dynamically generated in response to users' queries. The majority of these documents are generated through Web page templates, which contain information that is often irrelevant to queries. In this paper, we present a system designed to detect and extract query-related information from documents sampled from databases. The proposed system, 2PS, is based on a two-phase framework for the sampling, extraction and summarisation of Hidden Web documents. In the first phase, 2PS queries databases with random terms selected from those contained in their search interface pages and the subsequently retrieved documents – this phase retrieves a pre-determined number of sampled documents. In the second phase, it detects Web page templates from the sampled documents in order to extract information relevant to respective queries from which a content summary is generated. 2PS is validated through the implmementation of a prototype system. Its evaluation is performed through experiments on a number of real-world Hidden Web databases. The experimental results demonstrate that 2PS effectively eliminates irrelevant information contained in Web page templates and generates terms and frequencies with improved accuracy.

Keywords: Hidden Web Databases; Information Extraction; Document Sampling.

1. Introduction

Conventional and general-purpose search engines (e.g., Yahoo and Google) continuously expand their indexing capacity in order to accommodate the growth of Web pages. Search engines employ Web crawlers (a.k.a. spiders, robots) to index static Web pages through hyperlinks. However, these crawlers are unable to index a large amount of publicly accessible information, which includes dynamically generated Web pages through scripts or queries, multimedia objects and non-HTML files. Collectively, such information has become the largest growing category of information on the Internet and is referred to as 'Hidden Web' [8], 'Invisible Web' [23] or 'Deep Web' [3].

In this paper, we focus on databases that dynamically generate documents through queries. The examples of such documents are archives, user manuals and news articles, which possess the following distinguishing characteristics.

- Hidden Web documents are only accessible through the search interfaces of underlying databases.
- Knowledge about the contents of documents maintained in databases is often unavailable.
- Hidden Web documents are dynamically generated through one or more Web page templates.

The above characteristics of Hidden Web document databases raise a number of issues. For instance, with the proliferation of databases, it has become prohibitive for specialised search services (e.g., search.com) and subject directories (e.g., Open Directory Project) [16] to manually evaluate databases in order to assist users in their searches.

Current approaches [5, 8, 14, 15, 25] employ the techniques of database selection or categorisation to facilitate information search of databases. The former selects databases that contain information relevant to a user's query. Database categorisation automatically assigns databases into subject categories in order to assist users in browsing or searching for information. These techniques typically require knowledge about the contents of databases. In the domain of the Hidden Web, knowledge about the contents of databases is unavailable. Thus, a common approach to acquiring such knowledge is through the generation of statistics (i.e., terms and frequencies) from documents stored in databases. As it is not practical to retrieve all documents of a database to generate such statistical information, a number of research studies [5, 14, 15, 25] obtain a number of documents from the underlying database through sampling. A major issue associated with these techniques is that they also extract irrelevant information. Such information is often found in Web page templates, which contain navigation panels, search interfaces and advertisements. Consequently, the accuracy of terms and frequencies generated is reduced.

Given that current sampling process generates statistics without excluding irrelevant information, a mechanism to extract relevant information from sampled documents is necessary. We review several techniques [1, 6, 12, 13, 17, 18] that access or extract information from dynamic Web pages. In particular, the techniques in [1, 6, 17] perform well in the extraction of data from dynamic pages when a search interface contains a set of clearly labelled fields. For instance, a bookstore Web site enables users to query its database about authors and titles of publications. Information related to a given author and publication title can be identified from the dynamically generated Web pages. However, it is more difficult to identify relevant information from result pages, when little or no information is available from keyword-only search interfaces. The research in [19, 20] analyses dynamically generated Web pages, but it does not further determine whether information contained in the pages is relevant to queries.

The review of sampling and information extraction techniques identifies their limitations and associated problems. This provides the motivation for our research in addressing the aforementioned issues. The focus of this research is placed on the generation of statistics (i.e., terms and frequencies) about a database with improved accuracy, in particular, the problem associated with the extraction of relevant information from documents through random sampling. For instance, when a document database is queried with a randomly selected term, dynamically generated documents often contain irrelevant information. Such information is typically used to assist users in navigation or to describe the document contents. The extraction of such information results in the generation of terms and frequencies with reduced accuracy.

Therefore, this paper proposes a system, Two-Phase Sampling (2PS), which aims to extract relevant information from dynamic documents through which statistics with improved accuracy can be generated. 2PS is based on a two-phase framework for the sampling, extraction and summarisation of Hidden Web databases and is validated through the implementation of a prototype system. The evaluation is performed through experiments, which are conducted on a number of real-world Hidden Web databases. These databases contain computer manuals, healthcare archives and news articles. Our technique is assessed in terms of: (i) effectiveness in detecting Web page templates from sampled documents (ii) relevancy of information (extracted from the documents) to respective queries and (iii) the accuracy of terms and frequencies that are generated from sampled documents.

Experimental results demonstrate that 2PS effectively detects a large number of Web page templates from the sampled documents. We also found that our technique provides an effective mechanism to detect dynamically generated Web pages that do not contain query results, such as error and exception pages. Moreover, 2PS effectively identifies and eliminates information contained in Web page templates from sampled documents. Therefore, terms that are relevant to queries have attained high frequencies when top 20 terms are examined. When compared with query-based sampling, the results also show that 2PS generates statistics with improved accuracy, particularly in terms of precision.

The contributions of the proposed approach are summarised as follows.

- 2PS effectively detects information contained in Web page templates from sampled documents. The detection of such information enhances the effectiveness of extracting relevant information, as templates contain irrelevant terms for navigation or descriptive purposes. This approach differs from sampling techniques currently employed, which do not further analyse the contents of sampled documents to extract relevant information only.
- 2PS extracts terms that are relevant to queries from documents and generates statistics (i.e., terms and frequencies) with improved accuracy. This is opposed to the techniques employed in [5, 14, 25], which extract all terms from sampled documents, including those contained in Web page templates. Consequently, information that is irrelevant to queries is also extracted.

The remainder of the paper is organised as follows. Section 2 introduces current approaches to the discovery of information contents in Hidden Web databases. Related work on information extraction from dynamically generated Web pages is also discussed. Section 3 describes the proposed 2PS technique. Section 4 presents the evaluation of the proposed approach and reports on the experimental results. Section 5 concludes the paper.

2. Related Work

This section reviews current research and addresses a number of associated issues. A major area of current research into Hidden Web databases focuses on the automatic discovery of their information contents, in order to facilitate the process of database selection or categorisation. For instance, the technique proposed in [9] analyses the hyperlink structures of databases to determine their similarity in content. That is, databases with similar contents are discovered through Web pages that are linked to databases. This approach considers that all information contained in Web pages that are linked to a database is relevant to its content, but in practice Web pages may contain information irrelevant to database content.

An alternative solution is to examine the actual content of an underlying database by retrieving its documents. A problem associated with this approach is that it is difficult to obtain all documents from a database, particularly in the domain of Hidden Web. Thus, a number of techniques [5, 14, 25] retrieve the required number of documents through sampling. For instance, query-based sampling [5] generates terms and frequencies from sampled documents. This technique is applied in database selection where databases relevant to a user's query can be selected based on their terms and frequencies. The work conducted in [14, 25] sample databases with terms extracted from Web logs to obtain additional topic terms and frequencies. These terms and frequencies are also referred to as Language Models [5], Textual Models [14, 25] or Centroids [15].

A major issue regarding the aforementioned sampling techniques is that they extract information that is not relevant to queries. Irrelevant information (e.g., navigation or descriptive contents) is often found in Web page templates. For example, a language model generated from the sampled documents of the Combined Health Information Database (CHID) contains terms such as 'author' and 'format' with high frequencies. These terms are not relevant to queries but are used for descriptive purposes. Consequently, the accuracy of terms and frequencies generated from the documents has been reduced. The use of additional stop-word lists has been considered in [5] to eliminate irrelevant terms - but the study maintains that such a technique can be difficult to apply in practice.

Existing techniques in information extraction from Web pages are of varying degrees of complexity. A simple text-based analysis of Web pages is performed in [18], which applies approximate string matching techniques to extract texts that are different. This approach is limited to finding textual similarities and differences.

Several techniques have also been developed to address issues associated with the access or extraction of information contained in Hidden Web databases. For instance, the research in [12, 13] focuses on the submission of query forms in preparation for data extraction from dynamic Web pages. The techniques applied in [1, 6, 17] work well in the extraction of information from Web pages dynamically generated in response to the query entered in the fields of a search form. For example, a library database typically requires users to enter the name of an author and title of a publication. Information associated with these fields can then be identified from dynamic Web pages. However, in instances of keyword-only search interfaces where little or no additional information is given, it may be difficult to determine which information from dynamic Web pages is related to respective queries.

The approach proposed in [19, 20] analyses textual contents and tag structures in order to extract data from Web pages. However, it requires Web pages with well-conformed HTML tag-trees. Computation is also required to convert and analyse Web pages in a tree-like structure. Moreover, this approach identifies Web page templates based on a number of pre-defined templates, such as exception page templates and result page templates.

A number of studies [4, 7, 24] divide the contents of Web pages into blocks from which informative contents are identified. The approach in [24] assigns importance values to blocks in a Web page. However, when applying such an approach in the generation of statistics from the relevant contents of pages, the degree of importance with which blocks contain informative contents has yet to be determined. Similarly, [4] adopts a block-based approach to the analysis of Web page contents, which determines the rankings of Web pages to facilitate information search. A mechanism is also required in order to apply this technique in the extraction of relevant contents from pages. Moreover, blocks that are considered to be informative by [7] may also contain irrelevant contents. For instance, a block of text found in a CHID sampled document contains information relevant to healthcare. However, irrelevant terms such as 'author' and 'abstract' are also found in the text.

In terms of template detection, the algorithms proposed by [2] require that pages, links, and pagelets from a given collection of documents are stored as local database tables. This incurs extra processing time and storage.

In this paper, we focus on databases with a keyword-only search interface, which dynamically generate documents (e.g., archives and news articles). A distinct characteristic of documents found in such a domain is that the content of a document is often accompanied by other information for descriptive or navigation purposes. The proposed 2PS technique detects and eliminates template related information from sampled documents in order to identify query-related information. This differs from the approach in [6, 17], which extracts a set of data from dynamic Web pages in relation to query fields given in the search interface. Our technique also differs from [1] since we further determine the relevancy of information from sampled documents and eliminate information that is dynamically generated but irrelevant to queries.

In addition, we analyse Web documents based on textual contents and associated neighbouring tag structures rather than analysing their contents in a tree-like structure (e.g., Document Object Model (DOM)). A preliminary experiment conducted in [10] found that the proposed TNATS approach is faster when compared with the approach based on tree-like structures. Our approach requires less computation to analyse dynamic Web page contents whilst the accuracy in extracting relevant information (in terms of recall and precision) has been maintained or improved. We also detect information contained in different templates through which documents are generated. Therefore, it is not restricted to a pre-defined set of page templates.

3. The Two-Phase Sampling (2PS) Approach

This section presents the proposed technique for extracting information from Hidden Web document databases in two phases, which we refer to as *Two-Phase Sampling* (*2PS*). Figure 1 depicts the process of sampling a database and extracting query-related information from the sampled documents. In phase one, 2PS obtains randomly sampled documents. In phase two, it detects Web page templates, extracts information relevant to the queries and then generates terms and frequencies to summarise the database content. The two phases are detailed in section 3.1 and 3.2.

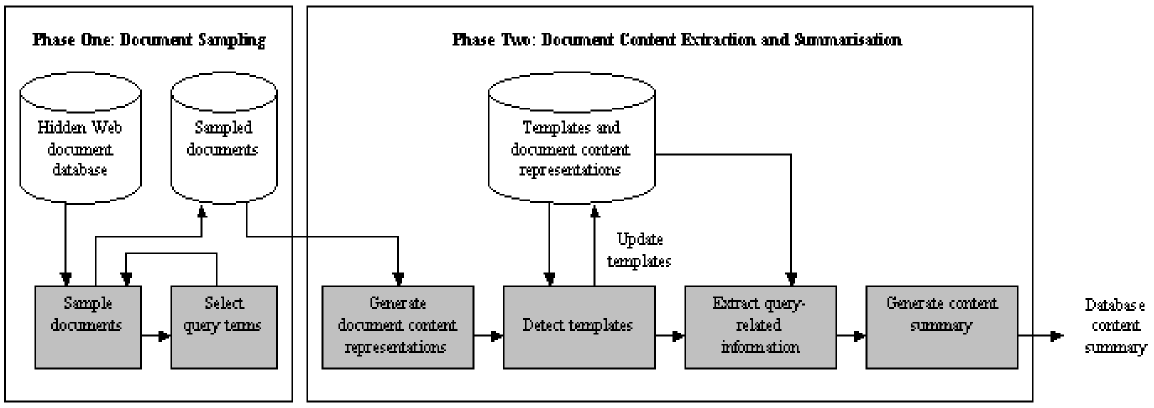

Phase One: Document Sampling
Hidden Web document database
Sampled documents
Sample documents
Select query terms
Phase Two: Document Content Extraction and Summarisation
Templates and document content representations
Update templates
Generate document content representations
Detect templates
Extract query-related information
Generate content summary
Database content summary

Figure 1. Overview of Two-Phase Sampling (2PS). In phase one, documents are sampled by submitting queries with terms randomly selected from those contained in search interface pages and documents that have been sampled. Phase two detects Web page templates from sampled documents and extracts relevant information, from which terms and frequencies are generated.

3.1. Phase One: Document Sampling

In the first phase, two sources of information are utilised to provide terms for sampling a database. These include terms extracted from the search interface pages of the database and subsequently sampled documents. That is, we initiate the process of sampling documents from a database with a randomly selected term from those contained in the search interface pages of the database. As search interface pages often contain information related to the underlying database, this provides a good collection of terms for initiating the sampling process. Alternatively, initial terms can also be selected from frequently used terms to query the database, which is employed in [5]. This retrieves top N documents where N represents the number of documents that are the most relevant to the query.

A subsequent query term is then randomly selected from terms extracted from the sampled documents. This process is repeated until a pre-determined number of documents are sampled. The number of documents to be sampled from a database is determined by the requirement of the system. For instance, the research work [5] shows that 300 sampled documents provide a sufficient representation for the content of a database. The sampled documents are stored locally for further analysis in the second phase.

Algorithm 1 The SampleDocument Algorithm

```
//Input: the URL address of the search interface page of a database
//Output: a given number of sampled documents
Extract $t_q$ from terms contained in the search interface pages of $D$, $Q = t_q$
$i = 0$     // $i$ as the number of documents that have been sampled
While ($i < n$)    // $n$ as the number of documents to sample
    Randomly select a query temrs, $qt_p$, from $Q$
    If ($qt_p$ has not been selected previously)
        Submit the query, $qt_p$, to $D$
        $j = 0$       // $j$ as the number of documents that have been retrieved for the query
        While $j <= N$      //Retrieve $N$ documents
            If ($d_i \notin R$)      //$R$ as the group of documents that have been previously retrieved
                Retrieve $d_i$ from $D$
                Extract $t_r$ from $d_i$,
                $R = d_i$, $Q = t_r$        //Update the group of previously retrieved documents, $R$, with
                                    //$d_i$ and update the term collection, $Q$, with $t_r$
                $j = j + 1$
            End if
        End while
    End if
    $i = i + 1$
End while
```

Algorithm 1 illustrates the process that obtains a number of randomly sampled documents. t_q denotes a term extracted from the search interface pages of a database, D. qt_p represents a query term selected from a collection of terms, Q, $qt_p \in Q$, $1 \leq p \leq m$; where m is the distinct number of terms extracted from the search interface pages and the documents that have been sampled. R represents the set of documents randomly sampled from D. t_r is a term extracted from d_i. d_i represents a sampled document from D, $d_i \in D$, $1 \leq i \leq n$, where n is the number of documents to sample.

2PS differs from query-based sampling in terms of selecting an initial query. The query-based sampling technique selects an initial term from a list of frequently used terms. 2PS initiates the sampling process with a term randomly selected from those contained in the search interface pages of the database. This utilises a source that may contain information related to its content. Moreover, 2PS analyses the sampled documents in the second phase in order to extract query-related information. By contrast, query-based sampling does not analyse their contents to determine whether terms are relevant to queries.

3.2 Phase Two: Document Content Extraction and Summarisation

The documents sampled from the first phase are further analysed in order to extract information relevant to the queries. This is followed by the generation of terms and frequencies to represent the content of the underlying database. This phase contains the following processes.

3.2.1 Generate Document Content Representations

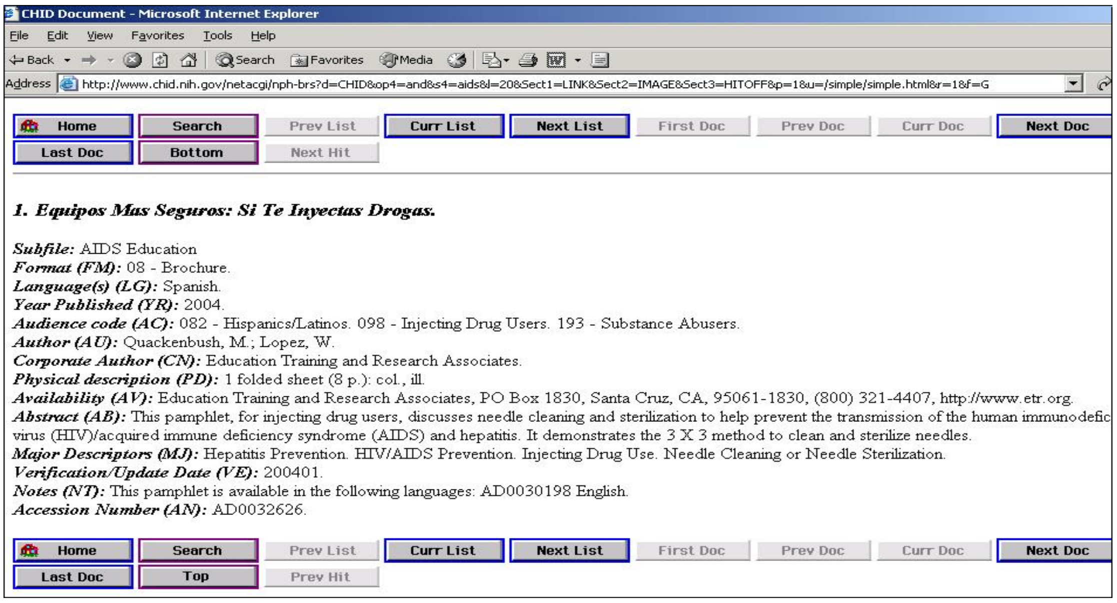
The content of each sampled document is converted into a list of text and tag segments. Tag segments include start tags, end tags and single tags specified in HyperText Markup Language

(HTML). Text segments are text that resides between two tag segments. The document content is then represented by text segments and their neighbouring tag segments, which we refer to as *Text with Neighbouring Adjacent Tag Segments* (*TNATS*). The neighbouring adjacent tag segments of a text segment are defined as the list of tag segments that are located immediately before and after the text segment until another text segment is reached. The neighbouring tag segments of a text segment describe how the text segment is structured and its relation to the nearest text segments. Assume that a document contains n segments, a text segment, txs, is defined as:

$$txs = (tx_i, tg\text{-}lst_j, tg\text{-}lst_k)$$

where tx_i is the textual content of the i^{th} text segment, $1 \le i \le n$; $tg\text{-}lst_j$ represents p tag segments located before tx_i and $tg\text{-}lst_k$ represents q tag segments located after tx_i until another text segment is reached. $tg\text{-}lst_j = (tg_1, \ldots, tg_p)$, $1 \le j \le p$ and $tg\text{-}lst_k = (tg_1, \ldots, tg_q)$, $1 \le k \le q$.

Figure 2 shows a template-generated document retrieved from the CHID database. The source code for this document is given in Figure 3.


CHID Document - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Home Search Prev List Curr List Next List First Doc Prev Doc Curr Doc Next Doc
Last Doc Bottom Next Hit
Home Search Prev List Curr List Next List First Doc Prev Doc Curr Doc Next Doc
Last Doc Top Prev Hit

http://www.chid.nih.gov/netacgi/nph-brs?d=CHID&op4=and&s4=aids&l=20&Sect1=LINK&Sect2=IMAGE&Sect3=HITOFF&p=1&u=/simple/simple.html&r=1&f=G

1. Equipos Mas Seguros: Si Te Inyectas Drogas.

Subfile: AIDS Education
Format (FM): 08 - Brochure.
Language(s) (LG): Spanish.
Year Published (YR): 2004.
Audience code (AC): 082 - Hispanics/Latinos. 098 - Injecting Drug Users. 193 - Substance Abusers.
Author (AU): Quackenbush, M.; Lopez, W.
Corporate Author (CN): Education Training and Research Associates.
Physical description (PD): 1 folded sheet (8 p.): col., ill.
Availability (AV): Education Training and Research Associates, PO Box 1830, Santa Cruz, CA, 95061-1830, (800) 321-4407, http://www.etr.org.
Abstract (AB): This pamphlet, for injecting drug users, discusses needle cleaning and sterilization to help prevent the transmission of the human immunodefic virus (HIV)/acquired immune deficiency syndrome (AIDS) and hepatitis. It demonstrates the 3 X 3 method to clean and sterilize needles.
Major Descriptors (MJ): Hepatitis Prevention. HIV/AIDS Prevention. Injecting Drug Use. Needle Cleaning or Needle Sterilization.
Verification/Update Date (VE): 200401.
Notes (NT): This pamphlet is available in the following languages: AD0030198 English.
Accession Number (AN): AD0032626.

Figure 2. A document dynamically generated from CHID contains navigational and descriptive contents.

```
…
<HTML><HEAD><TITLE>CHID Document
</TITLE></HEAD>
<BODY>
<HR><H3><B><I> 1.  Equipos Mas Seguros: Si
Te Inyectas Drogas.
</I></B></H3>
<I><B>Subfile: </B></I>
AIDS Education<BR>
<I><B>Format (FM): </B></I>
08 - Brochure.
<BR>
…
```

Figure 3. Source code for the CHID document.

In this example, text segment, ‘1. Equipos Mas Seguros: Si Te Inyectas Drogas.’, can be identified by the text (i.e., ‘1. Equipos Mas Seguros: Si Te Inyectas Drogas.’) and its neighbouring tag segments. These include the list of tags located before the text (i.e., </TITLE>, </HEAD>, <BODY>, <HR>, <H3>, <B> and <I>) and the neighbouring tags located after the text (i.e., </I>, </B>, </H3>, <I> and <B>). Therefore, this segment is represented as (‘1. Equipos Mas Seguros: Si Te Inyectas Drogas.’, (</TITLE>, </HEAD>, <BODY>, <HR>, <H3>, <B> ,<I>), (</I>, </B>, </H3>, <I>, <B>)). Figure 4 shows the generated TNATS content representation of the CHID document (given in Figure 3).

Assume that a sampled document, *d*, contains *n* text segments, the content of *d* is then represented as:

$$Content(d) = \{txs_1, \ldots, txs_n\}$$

where txs_i represents a text segment, $1 \leq i \leq n$.

```
…
'CHID Document', (<HTML>, <HEAD>,
<TITLE>), (</TITLE>, </HEAD>,  <BODY>,
<HR>, <H3>, <B>, <I>);
'1.  Equipos Mas Seguros: Si Te Inyectas
Drogas.', (</TITLE>, </HEAD>, <BODY>,
<HR>, <H3>, <B>, <I>), (</I>, </B>, </H3>,
<I>, <B>);
'Subfile:', (</I>, </B>, </H3>, <I>, <B>), (</B>,
</I>);
'AIDS Education', (</B>, </I>), (<BR>, <I>,
<B>);
'Format (FM):', (<BR>, <I>, <B>), (</B>, </I>);
…
```

Figure 4. TNATS content representation of a CHID document.

3.2.2Detect Templates

In the domain of Hidden Web databases, documents are often presented to users through one or more templates. Templates are typically employed in order to describe document contents or to assist users in navigation. For example, information contained in the document (as shown previously in Figure 2) can be classified into two categories as follows.

(i) Template-Generated Information. This includes information such as navigation panels, search interfaces and advertisements. In addition, information may be given to describe the content of a document. Such information is irrelevant to a user's query. For example, navigation links (such as 'Next Doc' and 'Last Doc') and headings (such 'Subfile' and 'Format') are found in the document.

(ii) Query-Related Information. This information is retrieved in response to a user's query, i.e., '1. Equipos Mas Seguros: Si Te Inyectas Drogas. …'.

The 2PS technique detects Web page templates employed by databases to generate documents in order to extract information that is relevant to queries. Algorithm 2 describes the process that detects information contained in Web page templates from *n* sampled documents. d_i, d_j represent a sampled document from the database D, d_i, $d_j \in D$, $1 \le i, j \le n$. *Content*(d_i) denotes the content representation of d_i where *Content*(d_j) denotes the content representation of d_j.

Similar to the representation for the contents of sampled documents, the content of a Web page template, *wpt*, is represented as *Content*(*wpt*) = $\{txs_1, \ldots, txs_q\}$, where q is the number of text segments, txs_j, $1 \le j \le q$. T represents a set of templates detected. $T = \{wpt_1, \ldots, wpt_r\}$, where r is the distinct number of templates, wpt_k, $1 \le k \le r$. G_k represents a group of documents generated from wpt_k. Furthermore, S represents the sampled documents from which no templates have yet been detected. Thus, $S = \{d_1, \ldots, d_s\}$, where s is the number of temporarily stored documents.

Algorithm 2 The DetectTemplate Algorithm

```
//Input: the sampled documents
//Output: the sampled documents with information contained in templates eliminated

$s = 0$
For $i = 1$ to $n$
    If $T = \emptyset$
        If $S = \emptyset$
          $S = d_i$, $s = s + 1$   //Update $S$ (with $d_i$,) in which no templates have yet been detected
        Else if $S \neq \emptyset$
            $j = 0$
            $k = 0$
            //Check $d_i$ with $S$ in which no templates have been detected
             While $j <= s$ AND $T = \emptyset$
                 Compare ($Content(d_i)$,$Content(d_j)$)   //Identify any repeated patterns from $d_i$, $d_j$
                 If $Content(d_i) \approx Content(d_j)$
                     $wpt_k = Content(d_i) \cap Content(d_j)$
                     Store $wpt_k$, $T = wpt_k$       //Store repeated patterns as a template
                     Delete ($Content(d_i) \cap Content(d_j)$) from $Content(d_i)$, $Content(d_j)$
                     $G_k = d_i$, $G_k = d_j$              //Update $G_k$ with $d_i$, $d_j$
                     Delete $d_j$ from $S$            //Remove $d_j$ from $S$
                 End if
                 $j = j + 1$
                 $k = k + 1$
             End while
                 If $T = \emptyset$    //If no template found from $d_i$, $d_j$
                 $S = d_i$          //Update $S$ with $d_i$
                 End if
        End if
```

(Contd.)

```
    Else if $T \neq \emptyset$
        $k = 0$
        While $d_i \notin G$
            Compare ($Content(wpt_k)$, $Content(d_i)$)
            If $Content(wpt_k) \approx Content(d_i)$             //If repeated patterns are found
                Delete ($Content(wpt_k) \cap Content(d_i)$) from $Content(d_i)$ //Eliminate template from $d_i$
                $G_k = d_i$        //Update $G_k$ with $d_i$
            End if
            $k = k + 1$
        End while

        //Check $d_i$ with temporarily stored documents, $S$, for templates
        If $S \neq \emptyset$ AND $d_i \notin G$
            $j = 0$
            While $j$ <= s AND $d_i \notin G$
                Compare ($Content(d_i)$,$Content(d_j)$)
                If $Content(d_i) \approx Content(d_j)$             //If repeated patterns are found
                    $wpt = Content(d_i) \cap Content(d_j)$
                    Store $wpt$, $T = wpt$                    //Store repeated patterns as a template
                    Delete ($Content(d_i) \cap Content(d_j)$) from $Content(d_i)$, $Content(d_j)$
                    // Create $G_{k+1}$ to store $d_i$, $d_j$, in which a template is found
                    $G_{k+1} = d_i$, $G_{k+1} = d_j$
                    Delete $d_j$ from $S$            //Remove $d_j$ from $S$
                End if
                $j = j + 1$
            End while
        End if

        If $d_i \notin G$            //If no template found
            $S = d_i$                //Update $S$ with $d_i$
        End if
    End if
End for
```

The process of detecting templates is executed until all sampled documents are analysed. This results in the identification of one or more templates. For each template, two or more documents are assigned to a group associated with the template from which the documents are generated. Each document contains text segments that are not found in their respective template. These text segments are partially related to their queries. In addition to a set of templates, the content representations of zero or more documents in which no matched patterns are found are stored for further analysis.

3.2.3 Extract Query-Related Information

This section illustrates the process of analysing sampled documents, which are associated with the templates detected previously (as described in section 3.2.2). The group of documents associated with each template are analysed. It further identifies any repeated patterns from the remaining text segments of the documents in order to extract query-related information.

The process of extracting query-related information is detailed as follows.

(i) Identify text segments that are highly similar in content. For each template detected, the TNATS representations of the documents are analysed if a group of documents are generated from the template. Text similarity is computed on text segments (with identical tag structures) from different documents. Two segments are considered to be similar if their similarity exceeds a threshold value.

(ii) Extract the textural contents of text segments that are relevant to queries. This extracts the textural contents of the text segments with different tag structures, in addition to those of the text segments that have identical adjacent tag structures - but which are significantly different in contents.

Cosine similarity [22] is applied to determine the similarities between the text segments of different documents that are associated with the template from which the documents are generated. The textual content of each text segment is represented as a vector of terms with weights. The weight of a term is obtained by the number of its occurrences in the segment.

The computation of cosine similarity is given as follows.

$$Sim(txs_i, txs_j) = \sum_{k=1}^{t} (tw_{ik} * tw_{jk}) \;/\; \sqrt{\sum_{k=1}^{t} (tw_{ik})^2} * \sqrt{\sum_{k=1}^{t} (tw_{jk})^2}$$

txs_i and txs_j represent two text segments in a document; tw_{ik} is the weight of term k in txs_i, and tw_{jk} is the weight of term k in txs_j. The computation of text similarities is performed on the text segments with identical adjacent tag segments only. Two segments are considered to be similar if their similarity exceeds a threshold value. The threshold value is used in order to determine the relevancy of text segments in the documents. Such a value is determined experimentally from the given databases.

Algorithm 3 gives the process of extracting information relevant to respective queries from the documents. d_a and d_b represent the sampled documents from the database, D, $d_a, d_b \in G_k$, where G_k denotes a group of documents associated with the template, wpt_k, from which the documents are generated. tx_m represents the textual content of a text segment, txs_m, contained in d_i, $d_i \in G_k$. tx_n represents the textual content of a text segment, txs_n, contained in d_l, $d_l \in S$. S represents the sampled documents from which no templates are detected.

Algorithm 3 The ExtractQueryInfo Algorithm

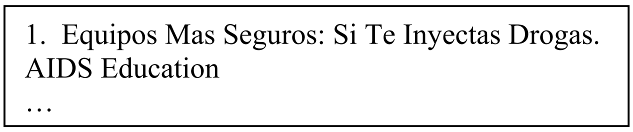
```
//Input: a number of templates identified, each with a group of associated sampled documents
//Output: the textual contents of remaining text segments from each of the documents

//For each template detected, check the associated group of documents, $G_k$, for more template
related information
For each ($d_a \in G_k$)
    For each ($d_b \in G_k$), $d_a \neq d_b$
        Compare ($Content(d_a)$,$Content(d_b)$)
        If $Content(d_a) \approx Content(d_b)$        //if repeated patterns are found
            //eliminate template from $d_a$, $d_b$
            Delete ($Content(d_a) \cap Content(d_b)$) from $Content(d_a)$, $Content(d_b)$
        End if
    End for
End for

//Extract the textual contents of remaining text segments from the documents in which templates
are found
For each ($d_i \in G$)
    Extract $tx_m$ of $txs_m$ from $Content(d_i)$
End for
```

(Contd.)

```
//Extract the textual contents of remaining text segments from the documents from which no templates are found
For each ($d_j \in S$)
    Extract $tx_n$ of $txs_n$ from $Content(d_j)$
End for
```



1. Equipos Mas Seguros: Si Te Inyectas Drogas.
AIDS Education
…

Figure 5. Query-related information extracted from the CHID document.

In brief, the above process identifies the text segments that have identical adjacent tag structures and are highly similar in their textual contents. Such segments are often used as page templates. For instance, text segments (including "…this library is free software; you can redistribute it and/or modify …" and "…this module is free software. It may be used, redistributed and/or modified… ") are found in the Help Site sampled documents. These segments provide auxiliary information and are highly similar in content. We eliminate these segments from the documents.

Remaining text segments in each document thus include the segments that have different neighbouring tag structures, and those segments that have identical adjacent tag structures but are significantly different in their textual contents. We consider remaining text segments to be relevant to queries since segments contained in templates are eliminated. The process results in the extraction of the textual contents from remaining text segments. Figure 5 shows a fraction of information extracted from the document content representation (as previously shown in Figure 4) as a result of eliminating information that is found in the Web page template.

3.2.4 Generate Content Summary

We compute frequencies for the terms extracted from the previous process described in Section 3.2.3. These summarise the information content of a database, which we refer to as *Content Summary*.

Previous experiments in the research study [5] demonstrate that randomly sampled documents (i.e., 300 documents, 4 documents per query) sufficiently represent the content of a database, which is referred to as 'language models'. Such a representation contains a number of term frequencies (i.e., the document frequency, collection term frequency and average term frequency). The study considers that these frequencies are more appropriate than inverse document frequency (*idf*) applied in traditional information retrieval - since the total number of documents in a database is typically unknown. In this paper, we also apply these frequencies in order to compare experimental results with those generated in [5]. The computation of the aforementioned frequencies is described as follows.

- Document frequency (*df*): the number of documents in the collection of documents sampled that contain term *t*, where *d* is the document and *f* is the frequency
- Collection term frequency (*ctf*): the occurrence of a term in the collection of documents sampled, where *c* is the collection, *t* is the term and *f* is the frequency
- Average term frequency (*avg_tf*): the average frequency of a term obtained from dividing collection term frequency by document frequency (i.e., $avg\_tf = ctf / df$)

The content summary of a document database is defined as follows. Assume that a Hidden Web database, D, is sampled with N documents. Each sampled document, d, is represented as a vector of terms and their associated weights [22]. Thus $d = (w_1, \ldots, w_m)$, where w_i is the weight of term t_i, and m is the number of distinct terms in $d \in D$, $1 \le i \le m$. Each w_i is computed using term frequency metrics, i.e., *ctf*, *df* and *avg_tf*. The content summary is then denoted as *CS*(*D*), which is generated from the vectors of sampled documents. Assume that *n* is the number of distinct terms in all sampled documents. *CS*(*D*) is then expressed as a vector of terms:

$$CS(D) = \{w_1, \ldots, w_n\}$$

where w_i is computed by adding the weights of t_i in the documents sampled from *D* and dividing the sum by the number of sampled documents that contain t_i, $1 \leq i \leq n$.

4. Evaluation of the 2PS Approach

This section presents the evaluation of 2PS. We report on the experiments conducted to assess its effectiveness in terms of: (i) detecting Web page templates contained in dynamically generated documents from Hidden Web databases through sampling and (ii) extracting information that is relevant to queries from sampled documents.

First, it describes the set up of our experiments. This is followed by the experimental results, which are compared with those from query-based sampling (QS). QS has been applied by other relevant studies for database selection or categorisation purposes [14, 15, 25]. Moreover, we compare 2PS with QS since the focus of this paper is to compare the proposed mechanism with a widely adopted sampling technique, which does not eliminate irrelevant information during the process of generating terms and frequencies from sampled documents.

4.1 Experimental Setup

Experiments are carried out on three real-world Hidden Web document databases, including Help Site, CHID and Wired News, which provide information about user manuals, healthcare archives and news articles, respectively. Table 1 summarises these databases in terms of their subjects, contents and the templates employed.

Table 1. Three Hidden Web databases used in the experiments

Database	**URL**	**Subject**	**Content**	**Template**
Help Site	www.help-site.com	Computer manuals	Homogeneous	Multiple templates
CHID	www.chid.nih.gov	Healthcare articles	Homogeneous	Single template
Wired News	www.wired.com	General news articles	Heterogeneous	Single template

For instance, Help Site and CHID contain documents relating to subjects on computing and healthcare, respectively. These documents contain information that is homogeneous in content. By contrast, Wired News contains articles that relate to different subjects of interest. Where the number of templates used to generate documents is concerned, CHID and Wired News generate documents using a Web page template. Help Site maintains a collection of documents produced by other information sources. Subsequently, different Web page templates are found in Help Site sampled documents.

Experiments are conducted on the set of Hidden Web document databases from which results are generated and compared for both 2PS and QS. We obtain the sampled documents for QS by submitting the first query to a database with a frequently used term. Subsequent query terms are randomly selected from those contained in the sampled documents. QS extracts terms (including terms contained in Web page templates) and updates the frequencies after each document is sampled. By contrast, 2PS initiates the sampling process with a term contained in the search interface pages of a database. 2PS also analyses sampled documents in the second phase in order to extract query-related information, from which terms and frequencies are then generated.

The experimental results in [5] conclude that QS obtains approximately 80% of terms from a database, when top 4 documents are retrieved for each query and 300 documents are sampled. These two parameters are therefore used in our experiments for QS and 2PS. We obtain five sets of samples for each database, from which four documents are sampled for each query and a total of 300 documents are retrieved. Terms and frequencies are generated from the documents sampled.

We manually process documents sampled from each database in order to observe the number of Web page templates employed by the databases and to identify information relevant to respective queries. The results provide the baseline for the experiments. Sampled documents are manually analysed as follows. First, each set of sampled documents is manually examined to obtain the number of Web page templates used to generate the documents. This is then compared with the number of templates detected by 2PS. The detection of Web page templates from the sampled documents is important as this determines whether irrelevant information is effectively eliminated.

Next, given the Web page templates that have been identified previously through manual observation, we eliminate any information found in page templates from each sampled document. Such information is included for navigation or descriptive purposes. For instance, a number of terms (including 'author' and 'abstract') are manually eliminated from the CHID documents, since these terms are used as auxiliary information only. We then generate terms and frequencies from those that remain in the document. The result of the above process provides the baseline in order to compare statistics generated using 2PS and QS. The number of relevant terms (from top 50 terms) retrieved using 2PS is compared with the number obtained by QS. Terms are ranked according to *ctf* and *df* frequencies to determine their relevancy to the queries. The *ctf* frequency represents the occurrences of a term contained in the sampled documents. Similarly, the *df* frequency of a term represents the number of documents in which the term appears. These frequencies are used to demonstrate the effectiveness of extracting query-related information from sampled documents since the terms extracted from Web page templates are often ranked with high *ctf* or *df* frequencies.

Finally, we compare the performance of QS and 2PS in terms of recall and precision. Recall and precision techniques [22] (of information retrieval systems) are modified (shown below) in order to measure their performance.

$$\text{recall} = \frac{\text{number of relevant terms retrieved}}{\text{number of relevant terms in sampled documents}}$$

$$\text{precision} = \frac{\text{number of relevant terms retrieved}}{\text{number of terms retrieved from sampled documents}}$$

A single measure, the F1 measure [21], is also computed in order to compare the overall results of the algorithms. This measures a combination of precision and recall and is defined as follows.

$$\text{F1} = (2 * \text{recall} * \text{precision}) / (\text{recall} + \text{precision})$$

4.2 Results and Discussion

This section reports on the experimental results, which are presented in the following categories. These categories include the detection of Web page templates from sampled documents, relevancy of terms retrieved and performance in terms of recall and precision.

4.2.1 Detection of Web page Templates

We compare the number of Web page templates employed by each of the databases and the number detected by 2PS in order to assess the effectiveness of template detection. Table 2 shows that 2PS effectively identifies a larger number of templates from the Help Site sampled documents. However, a small number of templates are not detected from Help Site. For instance, 2PS detects 15 templates from the first set of sampled documents. Two of the templates employed are not detected, since these templates are very similar in terms of content and structure.

Table 2. The number of templates employed by databases and the number detected by 2PS

Databases/ Sample set		**Number of templates**	
		Employed	**Detected**
Help Site	1	17	15
	2	17	16
	3	19	17
	4	16	14
	5	18	14
CHID	1-5	1	1
Wired News	1-5	1	1

4.2.2 Retrieval of Relevant Terms

We assess the effectiveness of 2PS by examining the number of relevant terms from top 50 terms retrieved from the sampled documents of Help Site, CHID and Wired News. This is compared to the number of relevant terms retrieved using QS. In our experiments, the relevancy of terms to queries is determined based on whether the terms are found in Web page templates.

Table 3, 4 and 5 summarise the number of relevant terms from top 50 terms ranked by *ctf, df* frequencies. The results generated from CHID and Wired News demonstrate that 2PS retrieves more relevant terms, as a large number of terms contained in the templates have been successfully eliminated from the top 50 terms. For instance, in the first set of documents sampled from CHID using 2PS, the number of relevant terms retrieved (ranked by *ctf* and *df* frequencies) is 47 and 44 respectively (as shown in Table 4). By comparison, the number of relevant terms obtained for QS is 20 and 12 respectively. Similarly, Table 5 shows that the number of relevant terms (ranked by *ctf* and *df* frequencies) retrieved from the first set of Wired News sampled documents using 2PS is 42 and 40, respectively. The number of relevant terms (ranked by *ctf* and *df* frequencies) obtained for QS is 10 and 5, respectively.

Table 3. The number of relevant terms retrieved from Help Site sampled documents (from top 50 terms ranked by *ctf*, *df* frequencies)

Sample set	**Number of relevant terms**			
	ctf		**df**	
	QS	**2PS**	**QS**	**2PS**
1	46	48	35	42
2	47	48	36	42
3	46	48	35	41
4	45	47	34	42
5	46	48	34	41

Table 4. The number of relevant terms retrieved from CHID sampled documents (from top 50 terms ranked by *ctf*, *df* frequencies)

Sample set	**Number of relevant terms**			
	ctf		**df**	
	QS	**2PS**	**QS**	**2PS**
1	20	47	12	44
2	19	47	12	45
3	20	47	11	44
4	18	46	11	44
5	17	46	12	44

Table 5. The number of relevant terms retrieved from Wired News sampled documents (from top 50 terms ranked by *ctf*, *df* frequencies)

Sample set	**Number of relevant terms**			
	ctf		**df**	
	QS	**2PS**	**QS**	**2PS**
1	10	42	5	40
2	10	43	4	38
3	11	39	4	39
4	10	40	4	40
5	12	42	5	39

However, the difference in the number of relevant terms (which appear in top 50 terms) retrieved by 2PS is less noticeable than the number by QS. Our observation is that template terms attain high frequencies since the CHID and Wired News databases generate documents using one Web page template only. By comparison, a larger number of Web page templates are found in the documents sampled from Help Site. As a result, only a small number of terms contained in the templates appear in top 50 terms compared with those found in the templates employed by CHID and Wired News. For instance, in the first set of documents sampled from Help Site using 2PS, the number of relevant terms retrieved (ranked by *ctf* and *df* frequencies) is 48 and 42, respectively (as shown in Table 3). The number of relevant terms obtained for QS is 46 and 35, according to their *ctf* and *df* frequencies, respectively.

Table 6. Top 50 terms ranked by *ctf* generated from CHID when QS is applied

Rank	**Term**	**Rank**	**Term**	**Rank**	**Term**
1	hiv	18	document	35	lg
2	aids	19	disease	36	ve
3	information	20	published	37	yr
4	health	21	physical	38	ac
5	prevention	22	subfile	39	corporate
6	education	23	audience	40	mj
7	tb	24	update	41	description
8	accession	25	verification	42	www
9	number	26	major	43	cn
10	author	27	pamphlet	44	pd
11	persons	28	chid	45	english
12	language	29	human	46	national
13	sheet	30	date	47	public
14	format	31	abstract	48	immunodeficiency
15	treatment	32	code	49	virus
16	descriptors	33	ab	50	org
17	availability	34	fm		

Table 6 and 7 show the top 50 terms that are ranked according to their *ctf* frequencies retrieved from the first set of sampled documents of the CHID database. Table 6 shows the top 50 terms retrieved for QS whereby terms contained in Web page templates are not excluded. As a result, a number of terms (such as ‘author’, ‘language’ and ‘format’) have attained much higher frequencies. By contrast, Table 7 lists the top 50 terms retrieved using 2PS. Our technique eliminates terms (such as ‘author’ and ‘format’) and obtains terms (such as ‘treatment’, ‘disease’ and ‘immunodeficiency’) in the higher rank.

Table 7. Top 50 terms ranked by *ctf* generated from CHID when 2PS is applied

Rank	**Term**	**Rank**	**Term**	**Rank**	**Term**
1	hiv	18	education	35	testing
2	aids	19	virus	36	programs
3	information	20	org	37	services
4	health	21	notes	38	clinical
5	prevention	22	nt	39	people
6	tb	23	cdc	40	hepatitis
7	persons	24	service	41	community
8	sheet	25	box	42	world
9	treatment	26	research	43	listed
10	disease	27	department	44	professionals
11	human	28	positive	45	training
12	pamphlet	29	tuberculosis	46	diseases
13	www	30	control	47	accession
14	http	31	drug	48	network
15	national	32	discusses	49	general
16	public	33	ill	50	std
17	immunodeficiency	34	organizations		

4.2.3 Extracting Query-Related Information

The section presents the results of experiments with regard to the performance of 2PS and QS in the extraction of information from sampled documents. For each database, we examine whether information extracted from its documents is relevant to the query submitted to the database. The performance of these two techniques is measured in terms of recall and precision.

Table 8, 9 and 10 summarise the performance of QS and 2PS in extracting query-related information from the sampled documents of Help Site, CHID and Wired News, respectively. Performance is measured in terms of recall and precision. As QS extracts all terms (including relevant and irrelevant terms) from the sampled documents, it attains a higher recall for all sample sets. On the other hand, QS results in a much lower degree of precision. By comparison, 2PS attains a lower recall since a number of relevant terms are eliminated as a result of their identical contents and tag structures. However, a higher degree of precision is obtained for 2PS, as a large amount of irrelevant information is eliminated. With regard to the overall performance, the results obtained from F1 demonstrate that 2PS performs better than QS for all sampled document sets from the three databases. In particular, 2PS attains much higher scores in F1 for the CHID database.

Table 8. The performance of extracting query-related information from Help Site sampled documents for QS and 2PS, measured in recall, precision and F1

Sample set	**QS**			**2PS**		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**
1	0.99	0.88	0.93	0.93	0.97	0.95
2	0.99	0.88	0.93	0.91	0.98	0.94
3	0.98	0.86	0.92	0.91	0.97	0.94
4	0.98	0.88	0.93	0.94	0.96	0.95
5	0.99	0.85	0.91	0.92	0.97	0.94

Table 9. The performance of extracting query-related information from CHID sampled documents for QS and 2PS, measured in recall, precision and F1

Sample set	**QS**			**2PS**		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**
1	1.0	0.81	0.90	0.98	0.97	0.97
2	1.0	0.80	0.89	0.98	0.98	0.98
3	1.0	0.80	0.89	0.98	0.97	0.97
4	1.0	0.81	0.90	0.98	0.97	0.97
5	1.0	0.80	0.89	0.98	0.97	0.97

Table 10. The performance of extracting query-related information from Wired News sampled documents for QS and 2PS, measured in recall, precision and F1

Sample set	**QS**			**2PS**		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**
1	1.0	0.75	0.86	0.91	0.98	0.94
2	0.99	0.74	0.85	0.90	0.97	0.93
3	0.98	0.75	0.85	0.90	0.97	0.93
4	0.99	0.75	0.85	0.91	0.98	0.94
5	0.99	0.76	0.86	0.92	0.98	0.95

Experimental results from the three document databases demonstrate that 2PS performs better than QS in the extraction of information when measured by the combination of recall and precision. However, 2PS incurs additional processing in detecting Web page templates from sampled documents. The latter requires algorithms with a higher degree of complexity in order to generate statistics with relevant terms. As opposed to 2PS, QS generates language models from sampled documents but does not exclude irrelevant terms.

5. Conclusions and Future Work

This paper presents a two-phase framework, 2PS, for the sampling, extraction, and summarisation of information from Hidden Web databases. 2PS extracts information relevant to queries from sampled documents and therefore generates statistics (i.e., terms and frequencies) with improved accuracy.

We conduct a number of experiments on three real-world databases to determine the effectiveness of 2PS in the detection of Web page templates employed by databases to generate dynamic documents. The performance for the extraction of query-related information is also assessed. Experimental results demonstrate that our technique effectively identifies a large number of Web page templates from sampled documents. The detection of templates facilitates the elimination of information contained in templates. The results show that 2PS generates terms and frequencies of a higher degree of relevancy in terms of precision, when compared with query-based sampling.

We obtain promising results by applying 2PS in experiments on three real-world document databases. Future work will include experiments on a larger number of Hidden Web databases to further assess the effectiveness of the proposed technique.

References

[1] A. Arasu and H. Garcia-Molina, Extracting Structured Data from Web Pages, in: Proc. of the ACM International Conference on Management (SIGMOD’03) 337-348.

[2] Z. Bar-Yossef and S. Rajagopalan, Template Detection via Data Mining and its Applications, in: Proc. of WWW2002, 580–591.

[3] M. K. Bergman, The Deep Web: Surfacing Hidden Value, Appeared in The Journal of Electronic Publishing from the University of Michigan (2001). Retrieved: 10 January 2005, from http://www.press.umich.edu/jep/07-01/bergman.html

[4] D. Cai, S. Yu, J.R. Wen and W.Y. Ma, Block-based Web Search, in: Proc. of the 27th Annual International Conference on Research and Development in Information Retrieval (SIGIR’04) 456-463.

[5] J. Callan and M. Connell, Query-Based Sampling of Text Databases, ACM Transactions on Information Systems (TOIS) 19 (2) (2001) 97-130.

[6] V. Crescenzi, G. Mecca and P. Merialdo, ROADRUNNER: Towards Automatic Data Extraction from Large Web Sites, in: Proc. of the 27th International Conference on Very Large Data Bases (VLDB’01) 109-118.

[7] S. Debnath, P. Mitra, and C. L. Giles, Automatic Extraction of Informative Blocks from Webpages, in: Proc. of the Special Track on Web Technologies and Applications in the ACM Symposium of Applied Computing 2005, 1722–1726.

[8] L. Gravano, P. G. Ipeirotis and M. Sahami, QProber: A System for Automatic Classification of Hidden-Web Databases, ACM Transactions on Information Systems (TOIS) 21(1) (2003) 1-41.

[9] M. Heß and O. Drobnik, Clustering Specialised Web-databases by Exploiting Hyperlinks, in: Proc. of the Second Asian Digital Library Conference, 1999.

[10] Y.L. Hedley, M. Younas and A. James, A TNATS Approach to Hidden Web Documents, in: Proc. of the First International Conference on Distributed Computing & Internet Technology (ICDCIT'04) 158-167.

[11] Y.L. Hedley, M. Younas, A. James and M. Sanderson, A Two-Phase Sampling Technique for Information Extraction from Hidden Web Databases, in: Proc. of the 6th ACM CIKM Workshop on Web Information and Data Management (WIDM'04) 1-8.

[12] J. P. Lage, A. S. da Silva, P. B. Golgher and A. H. F. Laender, Automatic Generation of Agents for Collecting Hidden Web Pages for Data Extraction, Data & Knowledge Engineering 49 (2) (2004) 177-196.

[13] S.W. Liddle, S.H. Yau and D. W. Embley, On the Automatic Extraction of Data from the Hidden Web, in: Proc. of the 20th International Conference on Conceptual Modeling Workshops (ER'01) 212-226.

[14] K.I. Lin and H. Chen, Automatic Information Discovery from the Invisible Web, in: Proc. of International Conference on Information Technology: Coding and Computing (ITCC'02) 332-337.

[15] W. Meng, W. Wang, H. Sun and C. Yu, Concept Hierarchy Based Text Database Categorization, International Journal on Knowledge and Information Systems 4 (2) (2002) 132-150.

[16] Open Directory Project (ODP), http://www.dmoz.org/.

[17] S. Raghavan and H. Garcia-Molina, Crawling the Hidden Web, in: Proc. of the 27th International Conference on Very Large Databases (VLDB'01) 129-138.

[18] B. Rahardjo and R. Yap, Automatic Information Extraction from Web Pages, in: Proc. of the 24th Annual International ACM Conference (SIGIR'01) 430-431.

[19] L. Ramaswamy, A. Iyengar, L. Liu, and F. Douglis, Automatic Detection of Fragments in Dynamically Generated Web Pages, in: Proc. of the 13th World Wide Web conference, 2004, 443–454.

[20] L. Ramaswamy, A. Iyengar, L. Liu, and F. Douglis, Automatic Fragment Detection in Dynamic Web Pages and its Impact on Caching, IEEE Transactions on Knowledge and Data Engineering, 17(5) (2005) 1–16.

[21] C. J. van Rijsbergen, Information Retrieval (Butterworths, London, 1979).

[22] G. Salton and M. McGill, Introduction to Modern Information Retrieval (McCraw-Hill, New York, 1983).

[23] C. Sherman, The Invisible Web, 2001. Retrieved: March 1, 2005, from http://www.freepint.com/issues/080600.htm

[24] R. Song, H. Liu, J.R. Wen, and W.Y. Ma, Learning Block Importance Models for Web Pages, in: Proc. of the 13th World Wide Web conference, pages, 2004, 203–211.

[25] A. Sugiura and O. Etzioni, Query Routing for Web Search Engines: Architecture and Experiments, in: Proc. of the 9th International World Wide Web Conference: The Web: The Next Generation (WWW9) 2000, 417-430.