DOI: 10.2298/CSIS100322028W

Research on Discovering Deep Web Entries

Ying Wang[1,2], Huilai Li[3], Wanli Zuo[1,2], Fengling He[1,2], Xin Wang[1,4], and Kerui Chen[1,2]

[1] College of Computer Science and Technology, Jilin University, 130012 Changchun, China
[2] Key Laboratory of Computation and Knowledge Engineering, Ministry of Education, China
{wangying2010, zuowl, hefl}@jlu.edu.cn, Chenke0616@163.com
[3] College of Mathematics, Jilin University, 130012 Changchun, China
lihuilai@jlu.edu.cn
[4] College of Software, Changchun Institute of Technology, 130012 Changchun, China
wangxccs@126.com
Wanli Zuo, zuowl@jlu.edu.cn

Abstract. Ontology plays an important role in locating Domain-Specific Deep Web contents, therefore, this paper presents a novel framework WFF for efficiently locating Domain-Specific Deep Web databases based on focused crawling and ontology by constructing Web Page Classifier(WPC), Form Structure Classifier(FSC) and Form Content Classifier(FCC) in a hierarchical fashion. Firstly, WPC discovers potentially interesting pages based on ontology-assisted focused crawler. Then, FSC analyzes the interesting pages and determines whether these pages subsume searchable forms based on structural characteristics. Lastly, FCC identifies searchable forms that belong to a given domain in the semantic level, and stores these URLs of Domain-Specific searchable forms to a database. Through a detailed experimental evaluation, WFF framework not only simplifies discovering process, but also effectively determines Domain-Specific databases.

Keywords: Deep Web, ontology, WPC, FSC, FCC.

1. Introduction

With the rapid development of the web, more and more information has been transferred from static web pages (that is Surface Web) into web databases (that is Deep Web) managed by web servers[1][2]. As Fig.1 conceptually illustrates, on this so-called “Deep Web”, numerous online databases provide dynamic query-based data access through their query interfaces, instead of static URL links[3]. The data in Deep Web are of great value, but difficult to query and search. With new web databases added and old web databases modified and removed constantly, artificial classification is a laborious and time-consuming task, so it is imperative to accelerate research on discovering effectively which searchable databases are most likely to contain the relevant information for which a user is looking.

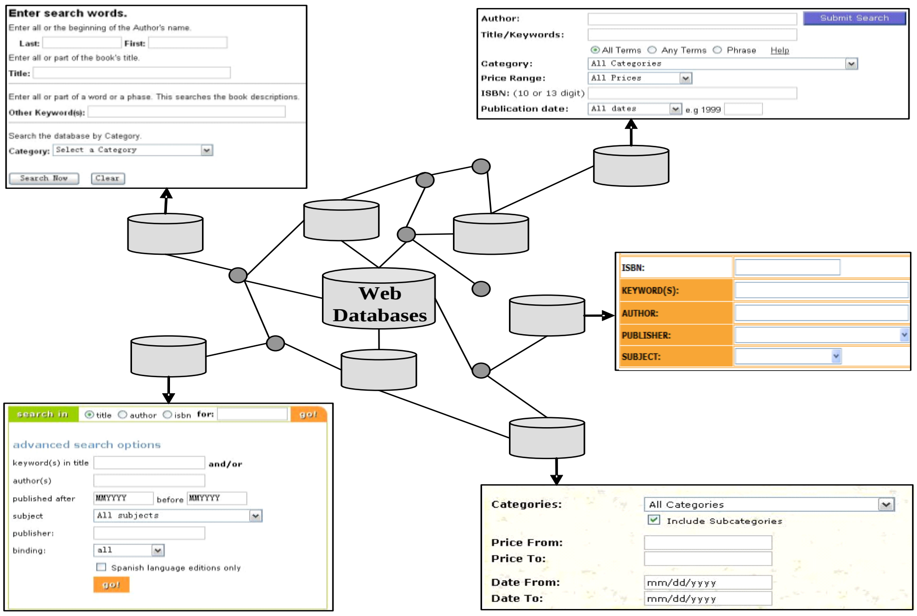

Fig. 1. Deep Web provides dynamic query-based data access through their query interfaces

Discovering Deep Web entries is the first significant step in integrating Deep Web data, in order to assist users accessing Deep Web, recent efforts have focused on two kinds of approaches to discover Deep Web entries automatically: Pre-Query and Post-Query[4].

Pre-Query identifies web databases by analyzing the wide variation in content and structure of forms. In 2005, Barbosa L and Freire J.[5] propose a crawling framework FFC to automatically locate Deep Web databases by focusing the search on a given topic; by learning to identify promising links; and by using appropriate stop criteria that avoid unproductive searches within individual sites. However, this method has some limitations: it requires substantial manual tuning and the form set retrieved by FFC is very heterogeneous. After two years, Barbosa L and Freire J.[6][7][8] present again a new framework ACHE that addresses these limitations, which automatically and accurately classifies online databases based on features that can be easily extracted from web forms. Manuel Alvarez et al.[9] provide the architecture of DeepBot, a prototype of hidden-web focused crawler able to access Deep Web content. Their approach is based on a set of domain definitions, each one describing a data-collecting task. From the domain definition, the system uses several heuristics to automatically identifying relevant query forms. Hui Wang and Wanli Zuo[10] propose a three-step framework to automatically identify domain-specific hidden Web entries. With those obtained query interfaces, they can be integrated to obtain a unified interface which is given to query for users. Li Yingjun et al.[11] propose a Domain-Oriented Deep Web data source Discovery method (DO-DWD) and a novel Domain Identification strategy of Deep Web data sources (DIDW). In the discovery stage, using machine learning algorithms and some heuristic rules to find query interfaces of the data sources; In the identification stage, identifying Deep Web data sources associated with the domain by calculating the relevance between a query interface and the domain based on semantic similarity. Pengyi Zhang et al.[12] propose a novel hybrid approach to construct a collection of government Deep Web resources. It combines automatic computation power and human intelligence through social computing. This approach presents the opportunity of building information structures on deep web portals in a scalable and sustainable manner. However, most of the above approaches do not consider applying background knowledge, which is important to understand problems and situations.

Post-Query approach identifies web databases from the retrieved results by submitting probing queries to the forms. In 2003, Luis Gravano and Panagiotis G.Ipeirotis[13] introduce QProber, a modular system that automates the classification process by using a small number of query probes, generated by document classifiers. However, this approach relies on a pre-learned set of queries for database classification. Additionally, if new categories are added or old categories removed from the hierarchy, new probes must be learned and each source re-probed. After five years, Luis Gravano and Panagiotis G.Ipeirotis[14] present a novel “focused-probing” sampling algorithm that detects the topics covered in a database and adaptively extracts documents that are representative of the topic coverage of the database. However, if the topic is not self-contained, then it will affect the database selection. Victor Z.Liu, et al.[15] develop a probabilistic approach to use dynamic probing(issuing the user query to the databases on the fly) in a systematic way, so that the correctness of database selection is significantly improved while the meta-searcher contacts the minimum number of databases. However, when the user does not care about the answer’s correctness, the method will not applicable. Lu Jiang et al.[16] propose a novel Deep Web crawling method with Diverse Features. They thought that the key to Deep Web crawling was to submit promising keywords to query form and retrieve Deep Web content efficiently. Keywords are encoded as a tuple by its linguistic, statistic and HTML features so that a harvest rate evaluation model can be learned from the issued keywords for the un-issued in future. One year later, Lu Jiang et al.[17] propose a novel Deep Web crawling framework based on reinforcement learning, in which the crawler is regarded as an agent and deep web database as the environment. The agent perceives its current state and selects an action (query) to submit to the environment according to Qvalue. The framework not only enables crawlers to learn a promising crawling strategy from its own experience, but also allows for utilizing diverse features of query keywords. However, it is some of wasting network and server resources by submitting a large number of queries only for the purpose of classification.

From the analysis above, Post-Query approach cannot be adapted to structured multi-attribute forms[18], so it is difficult for Post-Query approach to obtain better classification effects. Therefore, the method of Pre-Query which depends on visual features of searchable forms, namely, attribute labels and other available resources, are able to deal with highly heterogeneous form sets and usually used to indicative the database domain. That is to say, the discovery of Deep Web entries can be translated into the issue of distinguishing query forms. In this paper, we apply the Pre-Query approach for automatically classifying Domain-Specific forms by importing focused crawling and ontology technique. The paper is organized as follows: The section 2 presents the overview of discovering Deep Web entries, which includes problem formulation and WFF framework. The section 3 presents the process of WFF framework during discovering Deep Web entries. The section 4 presents the experiment results of WFF framework. Finally, in section 5, conclusions are drawn and future work is considered.

2. Overview

2.1. Problem Formulation

***Definition1*.** Deep Web Database: a Deep Web database is a web site, which contains searchable forms and a back-end database. Each database has specific searchable forms and result pages, generally, each searchable form is also known as "Input Schema", and result pages are known as "Output Schema", therefore, a database can be described as a triple-tuple (ds, IS, OS):

(1) ds denotes the back-end database behind a web site, which runs on web server.

(2) IS denotes a searchable form schema of web database, $IS = \{a_1, a_2, ...a_n\}$, where $a_i (0 \le i \le n)$ denotes a semantic attribute.

(3) OS denotes the result pages which are obtained by submitting requests from searchable forms.

***Definition2*.** Domain-Specific Database Discovery: It is used to judge whether a target database is relevant to the source database. Given a Deep Web source set $DS = \{ds_1, ds_i, ...ds_n\}$ and a category set

$C = \{C_1, C_2, ..., C_m\}$. Domain-Specific database discovery can be regarded as a mapping function from relational databases to the “best” category, namely formula (1):

$$f : DS \rightarrow C \tag{1}$$

The mapping function can make each database $ds_i \ (1 \le i \le n)$ from DS assign to a specific category $C_j \ (1 \le j \le m)$.

The fact that Deep Web sources are sparsely distributed makes especially challenging on locating them according to different domains[19]. There are mainly four questions:

Qustion1. How to find “entries” to Deep Web databases? The entry of each Deep Web database is the query interface(searchable form). To access a web database, we must firstly find its searchable form.

Qustion2. Which depth does each searchable form locate in a site? The depth of each searchable form is the minimum number of hops from the root page to the page which contains the searchable form.

Qustion3. How to recognize the searchable forms of Deep Web databases? Accessing to databases is provided only through restricted forms, not all the HTML forms are interfaces of Deep Web sites. HTML forms can be divided into searchable forms and non-searchable forms, searchable forms are query interfaces.

Qustion4. How to distribute the subject of web databases? There are great subject diversities among web databases, it is important to locate Domain-Specific databases.

Therefore, discovering topic relevant Deep Web entries accurately is one of the critical steps toward the integration of heterogeneous Deep Web sources.

2.2. WFF Framework

Since ontology is a well-formed knowledge representation, to access Deep Web effectively, we present a novel framework WFF for effectively locating Deep Web entry points based on focused crawling and ontology technique. WFF framework given in Fig. 2 consists of three main components: Web Page Classifier(WPC), Form Structure Classifier(FSC) and Form Content Classifier(FCC).

Firstly, WPC discovers potentially interesting pages based on ontology-assisted focused crawler. Then, FSC analyzes these interesting pages and determines whether these pages subsume searchable forms based on structural characteristics. Lastly, FCC identifies searchable forms that belong to a given domain in the semantic level, and stores these URLs of Domain-Specific searchable forms to a database. Discovering Deep Web entries is simplified by combining three hierarchical classifiers, which makes the overall classification process more accurate and robust.

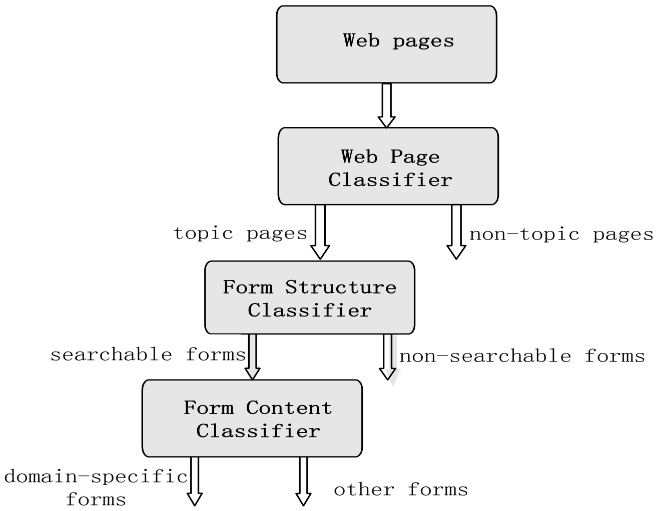

Fig. 2. WFF framework for discovering Deep Web entries, which contains Web Page Classifier, Form Structure Classifier and Form Content Classifier.

3. WFF Framework for Discovering Deep Web Entries

3.1. Ontology

Ontology as the foundation of knowledge processing, a concept model describing information system in semantic and knowledge level, user's queries and relevant data can be mapped to ontology, in this way, ontology can be seen as a knowledge system which describes concepts and relationships[20].

Definition3. Domain Ontology Concept Model(DOCM): DOCM is a data model that describes a set of concepts and relationships that may appear in a specific domain. It should be understandable by machine so that it can be used to reason about these objects within that domain. Each object can be denoted as $Class = \{CM, DT, \{S_i\}, \{CA_i\}, \{SC_i\}\}$, which describes the relevant information of object.

CM : The main class of object, which is universal and easy to understand for users. It can be seen as the keyword of object.

DT : The data type of object, such as “string”, “numerical” and so on.

$\{S_i\}$: The synonymous set of CM , namely, the concept aliases.

$\{CA_i\}$: The condition property set of object, which is "Part-Of" relationship to CM.

$\{SC_i\}$: The sub class set of CM, which is "Is-A" relationship to CM.

DOCM has a good organizational structure, which represents high-level background knowledge with concepts and relationships[21]. In this paper, the concepts and relationships of DOCM are extracted from searchable forms and result pages, and the ontology is implemented by Protégé API and represented in the Web Ontology Language(OWL)[22]. To operate ontology is equivalent to operate the OWL file.

An example of Book-Domain ontology is shown in Fig. 3.

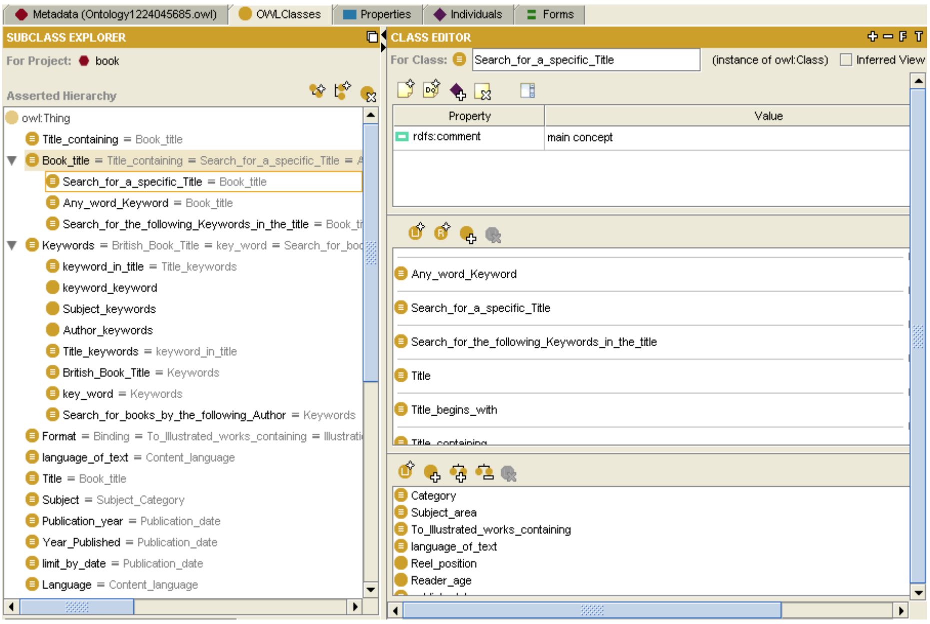

Fig. 3. An example of Book-Domain ontology, which describes the concepts and the logical relationships using a hierarchical tree structure.

3.2. WPC

WPC, namely, ontology-based focused crawling, which is used to guide the crawler and focus the search on interesting pages by analyzing features of web pages[23]. K. C.-C. Chang et al.[24] point out that the depth of Deep Web searchable form is less than 5, 94% of the searchable form depth is less than 3. Therefore, when locating an interesting page, the crawler will comply with two strategies:

Strategy1 The ontology-based crawler follows the hyperlinks from the page which is classified as being on topic.

Strategy2 The ontology-based crawler follows hyperlinks only to specific levels of depth.

Definition4. Page Similarity: Suppose $\vec{D}$ is a page feature vector containing m feature terms, $\vec{D}=\{(k_{1,d},w_{1,d}),(k_{2,d},w_{2,d}),...(k_{m,d},w_{m,d})\}$, $\vec{q}$ is a topic vector containing n feature terms, $\vec{q}=\{(t_{1,q},w_{1,q}),(t_{2,q},w_{2,q}),...(t_{n,q},w_{n,q})\}$. If these terms in page feature vector and topic vector can be found in ontology, then finding these corresponding concepts of terms from ontology, and replacing these terms with their corresponding concepts. These terms in page feature vector and topic vector can not be found from ontology called unlogin terms. After replacing these terms, page feature vector $\vec{D}$ can be divided into page concept vector $\vec{PCV}$ and page unlogin term vector $\vec{PUV}$, topic vector $\vec{q}$ can be divided into topic concept vector $\vec{TCV}$ and topic unlogin term vector $\vec{TUV}$.

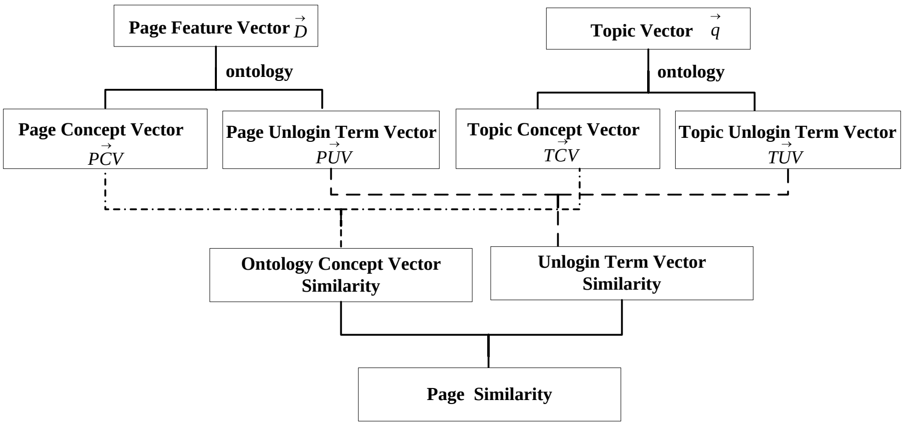

Fig. 4. The structure of page similarity computation, which contains ontology concept vector similarity and unlogin term vector similarity.

If several terms are matched with the same ontology concept, then replacing these terms with this concept, and summing these weights of several terms as the corresponding concept weight. The similarity between page feature vector $\vec{D}$ and topic vector $\vec{q}$ can be calculated in formula(2):

$$Sim(\vec{D},\vec{q})=\alpha\cdot Sim_{st\_ontology}(\vec{PCV},\vec{TCV})+(1-\alpha)\cdot Sim_{unlogin}(\vec{PUV},\vec{TUV}) \quad (2)$$

Where α is an impact factor, whose role is to adjust the impact to similarity between page concept vector $\overrightarrow{PCV}$ and page unlogin term vector $\overrightarrow{PUV}$. The structure of page similarity computation is shown in Fig. 4.

If a page which contains hyperlinks is topic relevant by page similarity algorithm, then we need to extract hyperlinks from the page and analyze the topic relevance of these hyperlinks, else, abandoning these hyperlinks.

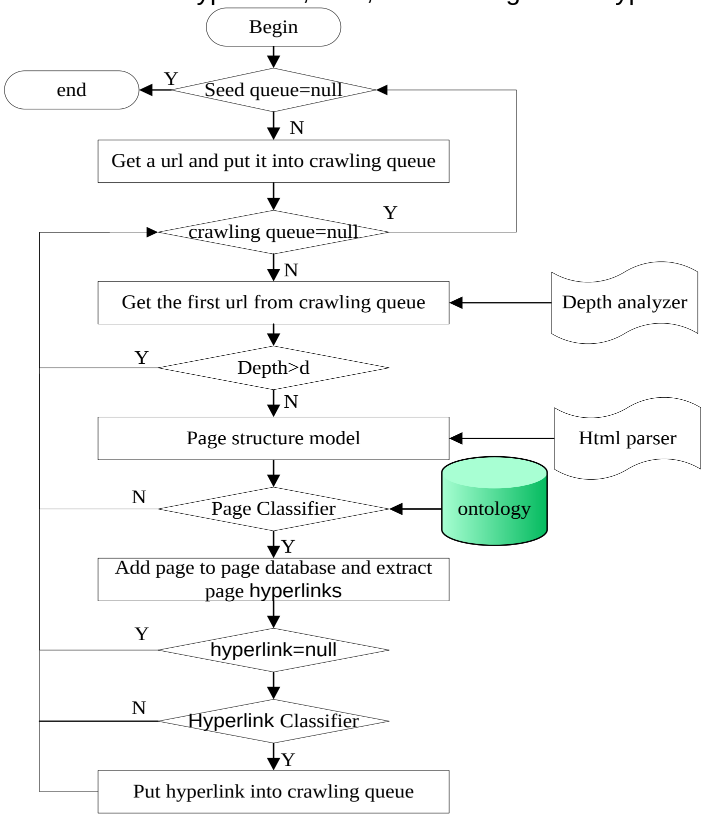

Fig. 5. WPC executive process: WPC receives as input a set of “seed” pages and recursively obtain new ones by following hyper-links in the standard depth-first traversal, lastly, recording interesting pages into repository and calling FSC.

Definition5. Hyperlink similarity: Extracting the anchor from topic page $\vec{D}$ to generate hyperlink anchor vector $\overrightarrow{Anchor} = \{(l_{1,link}, w_{1,link}), (l_2, w_{2,link}) \ldots (l_{k,link}, w_{k,link})\}$, and then calculating the anchor similarity $Sim(\overrightarrow{Anchor}, \vec{q})$ between anchor vector $\overrightarrow{Anchor}$ and topic vector $\vec{q}$ by page similarity method. The final hyperlink similarity can be calculated in formula(3):

$$Sim_{link}(\vec{Anchor}, \vec{q}) = \beta Sim(\vec{D}, \vec{q}) + (1-\beta) Sim(\vec{Anchor}, \vec{q}) \quad (3)$$

Where β is an impact factor, whose role is to adjust the impact to similarity between page feature vector $\vec{D}$ and anchor vector $\vec{Anchor}$.

The process of WPC is shown in Fig. 5.

3.3. FSC

***Definition6*.** Searchable form: The form characterized by its capacity of submitting a query to an online database. When a user submits queries in the searchable form, the queries will be issued against the database and return the results of query execution.

***Definition7*.** Non-searchable form: The form which does not represent database queries, for example, login forms, registration, mailing list subscriptions forms, email forms and so on.

FSC uses decision tree classifier which is proved to have lowest error rate[25]. Decision Tree algorithm is used to build the classifier of form structure for filtering out non-searchable forms and ensures only searchable forms that can be added to the form database.

***Definition8*.** Decision Tree: A Decision Tree is a decision support tool which uses a tree-like graph or model of decisions and their possible consequences. Each internal node tests an attribute, each branch corresponds to attribute value, and each leaf node assigns a classification[26][27].

C4.5 is an algorithm used to generate a Decision Tree developed by Ross Quinlan[28]. At each node of the tree, C4.5 chooses one attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. Its criterion is the normalized information gain that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists[29]. The information gain of attribute A_i is calculated with formula(4):

$$Gain(D, A_i) = Entropy(D) - Entropy_{A_i}(D) \quad (4)$$

Where D is the training examples, A_i is the splitting attribute. The information gain is based on entropy function from information theory, which is denoted in formula (5):

$$Entropy(D) = -\sum_{j=1}^{|C|} \Pr(c_j)\log_2 \Pr(c_j) \quad (5)$$

Where $\Pr(c_j)$ is the probability of class c_j in training examples D, which is the number of examples of class c_j in D divided by the total number of examples in D , $\sum_{j=1}^{|C|} \Pr(c_j) = 1$. If the number of possible values of the attribute A_i is v , and using A_i to partition the data D , we will divide D into v disjoint subsets $D_1, D_2 ... D_v$. The entropy after the partition by attribute A_i is shown in formula (6)[30]:

$$Entropy_{A_i}(D) = \sum_{j=1}^{|v|} \frac{|D_j|}{|D|} \times Entropy(D_j) \tag{6}$$

C4.5 Decision Tree algorithm is as follows:

C4.5 Decision Tree algorithm

Input: Training_examples D , attribute_list
Output: decision_tree
BEGIN
Generate_decision_tree(D , attribute_list)
1. Initialize()
2. creatNode(N)
3. if(Training_examples=null)
4.   return N="failure"
5. if(Training_examples $\in$ C)
6.   return leafNode(N)=C
7. if(attribute_list=null)
8.   return leafNode(N)=M(C)
9. for(each $A_i \in$ attribute_list)
10.   if(A_i is continuous)
11.   splitting(A_i)
12.   GrainRatio=compute(A_i)
13. selectMaxGrainRatio(A_i)
14. leafNode(N)= A_i
15. for each value d of A_i
16.   addCondition($A_i = d$)
17.   if($D_i = \phi$) // D_i is the subset of D based on the d value of A_i
18.   addLeafNode(N')=M(C)
19.   else
20. return Generate_decision_tree(D_i ,attribute_list)
END

The generated Decision Tree is shown in Fig. 6. Decision Tree builds an interpretable model that represents a set of rules.

```
decision_tree:
depth = 1
| isExist Form = No : Non-Searchable form
depth = 1
| isExist Form = Yes
| | depth = 2
| | | attribute-type isExist in AttributeTypeSet = No : Non-Searchable form
| | depth = 2
| | | attribute-type isExist in AttributeTypeSet = Yes
| | | | depth = 3
| | | | | special-attribute-numbers >= 3 = No : Non-Searchable form
| | | | depth = 3
| | | | | special-attribute-numbers >= 3 = Yes
| | | | | | depth = 4
| | | | | | | isExist SubmitButton = No : Non-Searchable form
| | | | | | depth = 4
| | | | | | | isExist SubmitButton = Yes
| | | | | | | | depth = 5
| | | | | | | | | is ButtonTypeSubmit = No : Non-Searchable form
| | | | | | | | depth = 5
| | | | | | | | | is ButtonTypeSubmit = Yes
| | | | | | | | | | depth = 6
| | | | | | | | | | | isExist QueryKeywordSet in {name,value} = Yes : Searchable form
| | | | | | | | | | | isExist QueryKeywordSet in {name,value} = N0 : Non-Searchable form
| | | | | | | | | is ImageTypeSubmit = Yes
| | | | | | | | | | depth = 6
| | | | | | | | | | | isExist QueryKeywordSet in {name,value,alt,src} = Yes : Searchable form
| | | | | | | | | | | isExist QueryKeywordSet in {name,value,alt,src} = N0 : Non-Searchable form
```

Fig. 6. From the Decision Tree, we can obtain the rules for classifying searchable forms and non-searchable forms.

The rules extracted from Decision Tree are as follows:

Rule1: If there is no <Form> tag in a page, then this page is non-searchable form.

Rule2: If there exists <Form> tag, then extracting attribute types between <Form> and </Form>. If each attribute type does not exist in “Attribute Type Set”, then this page is non-searchable form.

Rule3: If there exists <Form> tag, and there are attribute types in “Attribute Type Set”. If “Attribute Number” is less than 3, then this page is non-searchable form.

Rule4: If there exists <Form> tag, and there are attribute types in “Attribute Type Set”, “Attribute Number” is more than 3, but there is no submit button, then this page is non-searchable form.

Rule5: If there exists <Form> tag, there are attribute types in “Attribute Type Set”, “Attribute Number” is more than 3, and there exists submit button with “submit” type, but “Button Marker” does not exist in “Search Word Set”, then this page is non-searchable form.

Rule6: If there exists <Form> tag, there are attribute types in “Attribute Type Set”, “Attribute Number” is more than 3, and there exists submit button with “image” type, but “Image Marker” does not exist in “Search Word Set”, then this page is non-searchable form.

Rule7: If there exists <Form> tag, there are attribute types in “Attribute Type Set”, “Attribute Number” is more than 3, there exists submit button with “submit” type, and “Button Marker” is in “Search Word Set”, then this page is searchable form.

Rule8: If there exists <Form> tag, there are attribute types in “Attribute Type Set”, “Attribute Number” is more than 3, there exists submit button with “image” type, and “Image Marker” is in “Search Word Set”, then this page is searchable form.

FSC based on Decision Tree classifies the searchable forms and non-searchable forms by the above rules.

3.4. FCC

Though FSC, we can find that the topic relevant page contains a searchable form, however, the form content retrieved may belong to a different domain. Therefore, a novel method of ontology-assisted FCC is proposed to identify Domain-Specific databases by analyzing Domain-Specific form content[31][32][33].

Definition9. Ontology assisted FCC: Suppose $\vec{F}=\{(f_{1,d},w_{1,f}),(f_{2,d},w_{2,f}),...(f_{m,d},w_{m,f})\}$ is a form feature vector containing m form feature terms, where $(f_{i,f},w_{i,f})$ $(1\le i\le m)$ denotes a form feature term and its corresponding weight. $\vec{q}$ is the topic vector containing n feature terms $q=\{(t_{1,q},w_{1,q}),(t_{2,q},w_{2,q}),...(t_{n,q},w_{n,q}),\}$, where $(t_{j,q},w_{j,q})$ $(1\le j\le n)$ denotes a topic term and its corresponding weight. Generally, the vocabularies of searchable form are restricted and not duplicated, therefore, we set the weight $w_{i,d}$ of each feature term $t_{i,f}$ as $1/m$. For each feature term $t_{i,d}$ in form d , there are three cases:

Case1 If $t_{i,d}\in DOCM$, then, setting $Sim_i(t_{i,d},\vec{q})=1$.

Case2 If $t_{i,d}\notin DOCM$ and $t_{i,d}\notin\vec{q}$, then, $Sim_i(t_{i,d},\vec{q})=0$.

Case3 If $t_{j,q}\in\vec{q}$, and $t_{j,q}=t_{i,d}$, then, $Sim_i(t_{i,d},\vec{q})=\dfrac{w_{i,d}+w_{j,q}}{2}$.

The final similarity between form feature vector $\vec{F}$ and topic vector $\vec{q}$ can be calculated in formula(7):

$$Sim(\vec{F},\vec{q})=\frac{\sum_{i=1}^{m}Sim_i(t_{i,d},\vec{q})}{m} \tag{7}$$

The process of FCC is shown in Fig. 7:

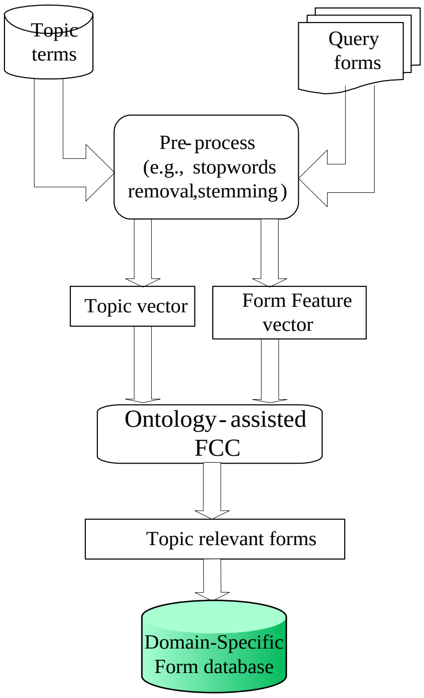

Fig. 7. Ontology plays an important role in recognizing Deep Web entry forms. Therefore, an ontology assisted FCC algorithm was proposed to locate Domain-Specific query interfaces.

4. Experiments

Though the above analysis, we implement the graphical interface for discovering Deep Web entries which is shown in Fig.8.

We evaluate our method with four experiments, respectively, WPC, FSC, FCC and WFF.

Experiment 1 **WPC:** Harvest is usually used to evaluate focused crawling, and it means the fraction of web pages crawled which satisfy the crawling target among the crawled pages. The harvest is shown in formula (8):

$$harvest = \frac{\sum_{p \in P} rel(p)}{|P|} \qquad (8)$$

Where $|P|$ denotes the number of web pages crawled, $rel(p)$ denotes the number of specific topic pages. The initial URLs for the crawler are 100 Book-Domain URLs, which are managed by a manual directory.

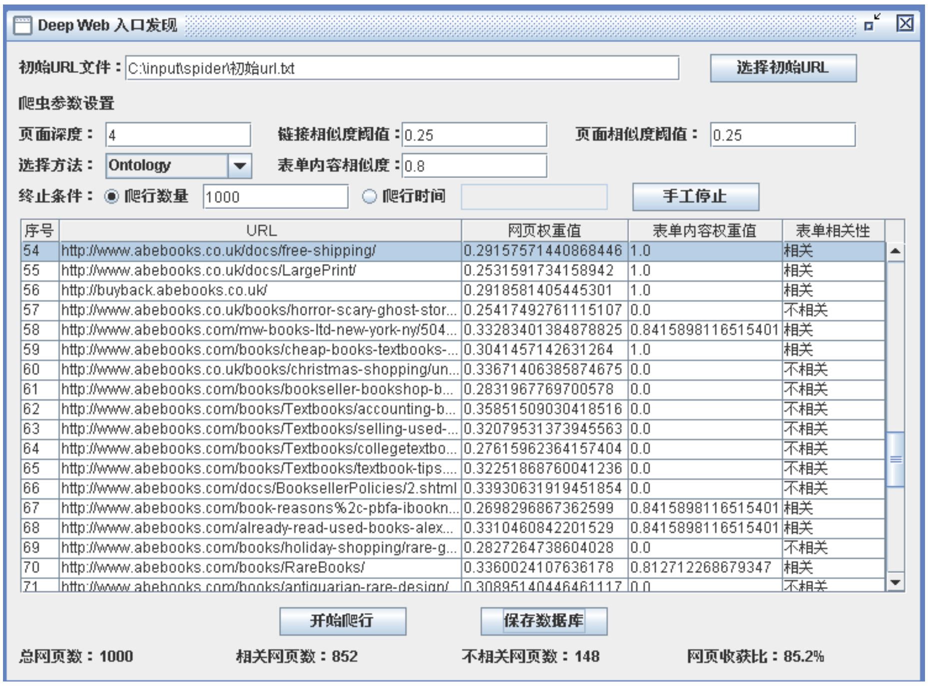

Fig. 8 The graphical interface for discovering Deep Web entries

If the impact factor α is set 0.5 in formula(11), namely, they share the same proportion for page concept vector $\overrightarrow{PCV}$ and unlogin term vector $\overrightarrow{PUV}$, then, though analyzing these 100 Book-Domain pages, the similarity distribution is that 78% pages is more than 0.3, 96% pages is more than 0.25, and 4% pages is less than 0.25, therefore, in most cases, it is more reasonable for setting page similarity threshold(PS) to 0.25 or 0.3. Similarly, the impact factor β is set 0.7 in formula(12), that is to say, we think page similarity is more important than anchor similarity, though analyzing 100 Book-Domain hyperlinks, the similarity distribution shows that 94% hyperlinks is more than 0.25, 97% hyperlinks is more than 0.2, and 3% pages is less than 0.2, therefore, in most cases, it is more reasonable for setting hyperlink similarity threshold(HS) to 0.25. Simultaneously, setting another two parameters: page depth d=4, the maximum number of crawling pages $N = 2000$. We study the performance of WPC by two crawlers with distinct focus strategies: ontology-based focused crawler(OFC) and Best-First focused crawler(BFC)[34]. Best-First focused crawler is based on TF-IDF weight model, though analyzing Book-Domain pages and hyperlinks, in most cases, it is more reasonable for setting page similarity threshold(PS) based on Best-First method to 0.5, and hyperlink similarity threshold(HS) to 0.35. Fig.9 illuminates the performance for OFC and BFC.

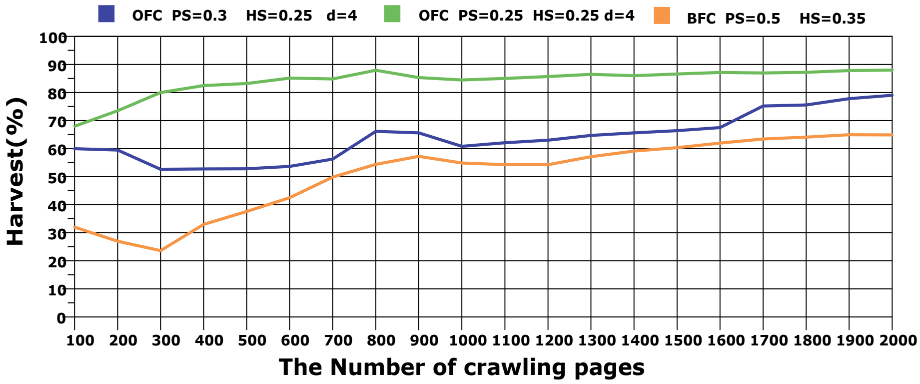

Fig. 9. The result of Web Page Classifier. From the results of WPC, when PS=0.25, it has a higher harvest ratio than PS=0.3. Because that the page similarity for 78% pages is more than 0.3, and 96% pages is more than 0.25, if PS=0.3, it will miss some Domain-Specific pages, so the harvest for PS=0.25 is higher than PS=0.3. whatever page similarity is set 0.25 or 0.3, OFC is performing better with respect to harvest ratio than BFC as the crawling progresses, the substantial increases in harvest ratio is obtained because that OFC relates the crawling topics to the background knowledge base in order to filter out irrelevant web pages.

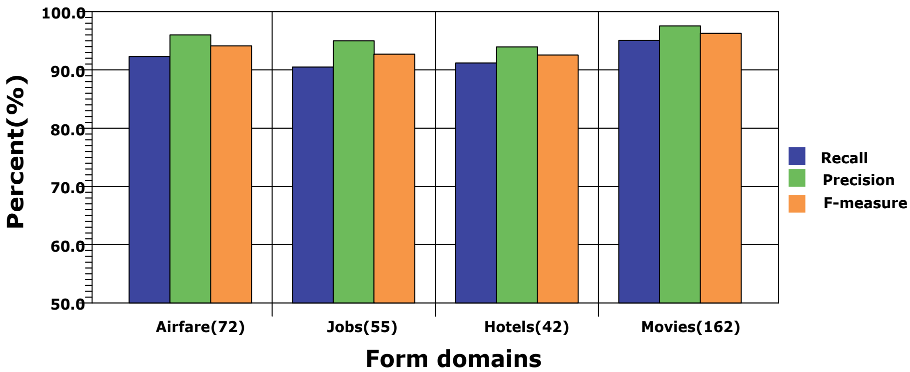

Fig. 10. The results of FSC in different domains, we can see that FSC based on Decision Tree can obtain satisfied accuracy in different domains.

Experiment 2 **FSC:** The evaluation metric for Form Structure Classifier is called Precision, Recall and F-measure. Precision is the percentage of correctly identified searchable forms over all the identified searchable forms by Form Structure Classifier. Recall is the percentage of correctly identified searchable forms over all the searchable forms. F-measure denotes a harmonic mean between precision and recall. In this study, FSC based on Decision Tree is domain-independent, and it is general and can be applied to many different domains. In order to validate FSC, we select four domains from UIUC data set: Airfare, Jobs, Hotels, Movies. The results are shown in Fig.10.

FSC based on Decision Tree can obtain satisfied accuracy. Therefore, the method of FSC based on Decision Tree is feasible.

Experiment 3 **FCC:** The evaluation metric for FCC is also Precision, Recall and F-measure. Precision is the percentage of correctly identified Domain-Specific forms over all the identified Domain-Specific forms by FCC algorithm. Recall is the percentage of correctly identified Domain-Specific forms over all the Domain-Specific forms. F-measure denotes a harmonic mean between Precision and Recall. Similarity threshold setting is a critical step for searchable form classification. There are different results on Recall, Precision and F-measure with different threshold. The threshold is not as small as possible, or the greater the good. In order to better understand the three evaluation metrics, we are on to experiment with different thresholds, which are 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9. The number of selected forms is 160 Book forms. FCC correctness ratio is shown in Fig.11:

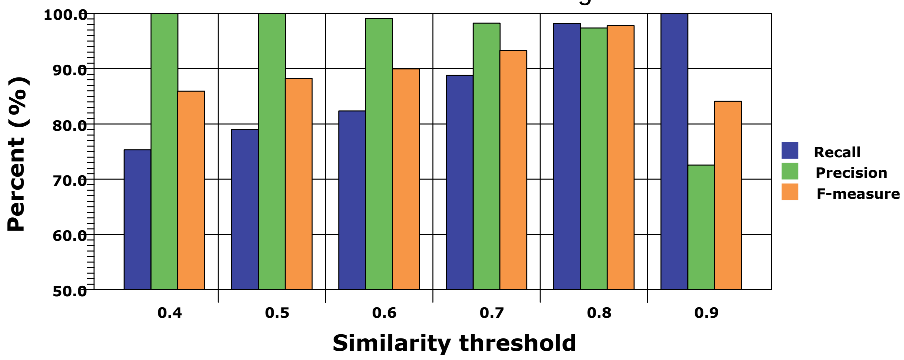

Fig. 11. From the results of FCC, we can see that when the similarity threshold is set low, the results contain most relevant pages, and mistake a lot of irrelevant pages relevant, so Precision is low and Recall is high. When the similarity threshold is set high, it will ignore most relevant pages, so Precision is high and Recall is low. When $\theta = 0.8$, there is a higher accuracy for Recall, Precision and F-measure, therefore, it is more reasonable for $\theta = 0.8$. It also proves that the method of ontology-assisted FCC can identify Domain-Specific forms with high accuracy.

Experiment 4 WFF: If the maximum number of pages for crawler $N = 10000$ and FCC threshold $\theta = 0.8$, then, with the increase of crawling pages, the changes for Domain-Specific forms by OFC and BFC are shown in Fig.12.

Through the detailed analysis above, it indicates that the WFF framework is a scalable alternative to efficiently locate Deep Web entry points based on focused crawling and ontology technique.

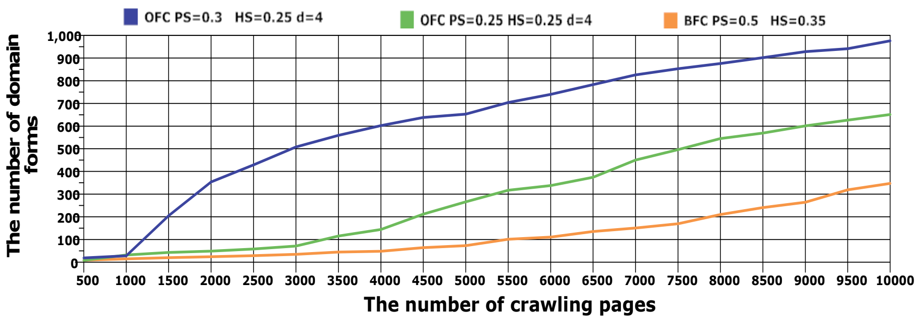

Fig. 12 The number of crawling domain forms for OFC and BFC. From the results of WFF, when PS=0.25, OFC will mistake some irrelevant pages relevant, in this way, it will crawl some useless pages. Therefore, the number of crawling domain forms for PS=0.3 are more than PS=0.25. Compared with BFC, OFC can obtain more Domain-Specific forms than BFC, because that BFC does not consider the page depth, when BFC obtain a page whose page similarity is more than threshold, it will parse the page, however, 94% of the searchable form depth is less than 3. Therefore, BFC has crawled a large number of pages without domain forms.

5. Conclusion

In this paper, we have presented a framework WFF for identifying Deep Web entries based on ontology and focused crawling automatically. Our approach composes three classifiers by partitioning the process into three modules: WPC, FSC and FCC. In the future work, we will conduct further research to improve our work in the following ways: Firstly, we will enrich the ontology, because that the classification accuracy to a large extent depends on the complete ontology knowledge base. Secondly, we will study an effective way of analyzing the hyperlinks in the visited pages to filter the irrelevant pages more efficiently. Finally, we will explore the more effective method to improve the classification accuracy in more depth.

Acknowledgment. This work is supported by the National Natural Science Foundation of China under Grant No.60973040; the National Natural Science Foundation of China under Grant No.60903098; the Science and Technology Development Program of Jilin Province of China under Grant No. 20070533; the Specialized Research Foundation for the Doctoral Program of Higher Education of China under Grant No.200801830021; the Basic Scientific Research Foundation for the Interdisciplinary Research and Innovation Project of Jilin University under Grant No.450060445161;.the Basic Scientific Research Foundation for Young Teachers Innovation Project of Jilin University under Grant No.450060441075.

References

1. Denis Shestakov, Sourav S. Bhowmick, Ee-Peng Lim: DEQUE: Querying the Deep Web. Data & Knowledge Engineering, Vol. 52, 273–311. (2005)
2. Jianguo Lu, Dingding Li: Estimating Deep Web Data Source Size by Capture–Recapture Method. Journal of Information Retrieval, 70-95. (2010)
3. Ritu Khare, Yuan An, Il-Yeol Song: Understanding Deep Web Search Interfaces: A Survey. SIGMOD, Vol. 39, No. 1, 33-40. (2010)
4. Y.Ru and E.Horowitz: Indexing the Invisible Web: A Survey. Online Information Review, Vol. 29, No. 3, 249-265. (2005)
5. Barbosa L and Freire J: Searching for Hidden-Web Databases. In Proceedings of WebDB, 1-6. (2005)
6. Barbosa L, Freire J: Combining Classifiers to Identify Online Databases. In Proceedings of the World Wide Web Conference(WWW), 431-440. (2007)
7. Barbosa L, Freire J: An Adaptive Crawler for Locating Hidden-Web Entry Points. In Proceedings of the World Wide Web Conference(WWW), 441-450. (2007)
8. Luciano Barbosa, Hoa Nguyen, Thanh Nguyen: Creating and Exploring Web Form Repositories. SIGMOD, 1175-1177. (2010)
9. Manuel Alvarez, Juan Raposo, Alberto Pan, Fdel Cacheda, Victor Carneiro: Deep-Bot: A Focused Crawler for Accessing Hidden Web Content. In Proceedings of DEECS, 18-25. (2007)
10. Hui Wang, Yanwei Liu, Wanli Zuo: Using Classifiers to Find Domain-Specific Online Databases Automatically. Journal of Software, Vol. 19, No. 2, 246-256. (2008)
11. Li Yingjun, Nie Tiezheng, Shen Derong, Yu Ge: Domain-oriented Deep Web Data Sources Discovery and Identification. In Proceedings of Asia Pacific Web Conference, 464-467. (2010)
12. Pengyi Zhang, Yan Qu, Chen Huang, Paul T.Jaeger, John Wells, W.Scott Hayes, James E.Hayes Xin Jin: Collaborative Identification and Annotation of Government Deep Web Resources: A Hybrid Approach. 285-286. (2010)
13. Luis Gravano, Panagiotis G.Ipeirotis: QProber: A System for Automatic Classification of Hidden Web Databases. ACM Transactions on Information Systems, Vol. 21, No. 1, 1-41. (2003)
14. Panagiotis G.Ipeirotis, Luis Gravano: Classification-Aware Hidden-Web Text Database Selection. ACM Transactions on Informaion Systems, Vol. 26, No. 2. (2008)
15. Victor Z.Liu, Richard C.Luo, Junghoo Cho, Wesley W. Chu: D_Pro: A Probabilistic Approach for Hidden Web Database Selection Using Dynamic Probing. ICDE, 1-12. (2004)
16. Lu Jiang, Zhaohui Wu, Qinghua Zheng and Jun Liu: Learning Deep Web Crawling with Diverse Features. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 572-575. (2009)
17. Lu Jiang, Zhaohui Wu, Qian Feng, Jun Liu, and Qinghua Zheng: Efficient Deep Web Crawling Using Reinforcement Learning. PAKDD, 428–439. (2010)
18. Pierre Senellart, Avin Mittal, Daniel Muschick, Remi Gilleron, Marc Tommasi: Automatic Wrapper Induction from Hidden-web Sources with Domain Knowledge. WIDM, 9-16. (2008)
19. Ping Wu, Ji-Rong Wen, Huan Liu, Wei-Ying Ma: Query Selection Techniques for Efficient Crawling of Structured Web Sources. In Proceedings of ICDE, pp47-57. (2006)

20. Elena Simperl: Reusing ontologies on the Semantic Web: A feasibility study. Data & Knowledge Engineering, Vol. 68, 905–925. (2009)
21. Weifeng Su, Jiying Wang, Frederick H.Lochovsky: ODE: Ontology-assisted Data Extraction. ACM Transactions on Database Systems, Vol. 34, No. 2, 1-35. (2009)
22. Matthew Horridge, Bijan Parsia, Ulrike Sattler: Explanation of OWL Entailments in Protege4. In Proceedings of International Semantic Web Conference. (2008)
23. Gechao Lu, Wanli Zuo, Aiqi Zhang, Ying Wang, Wenyan Ji: Ontology-Based Focused Crawler. Journal of Information&Computational Science, 577-584. (2010)
24. K. C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang: Structured Databases on the Web: Observations and Implications. SIGMOD Record, Vol. 33, No. 3, 61-70. (2004)
25. Barbosa, L., Freire, J.: Searching for Hidden-web Databases. In: Eighth Intl. workshop on the web and Databases. (2005)
26. Tom M. Mitchell, McGraw Hill. Machine Learning. (1997)
27. S. Sivakumari, R. Praveena Priyadarsini, P. Amudha: Accuracy Evaluation of C4.5 and Naive Bayes Classifiers Using Attribute Ranking Method. International Journal of Computational Intelligence Systems, Vol. 2, No. 1, 60-68. (2009)
28. Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers. (1993)
29. http://en.wikipedia.org/wiki/C4.5_algorithm
30. Bing Liu. Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Springer. (2007)
31. Huilan Zhao: Research on Deep Web Sources Classification Technology. In Proceedings of the 2nd International Conference on Future Computer and Communication, 169-171. (2010)
32. Hexiang Xu, Xiulan Hao, Shuyun Wang, Yunfa Hu: A Method of Deep Web Classification. In Proceedings of Machine Learning and Cybernetics, 4009-4014. (2007)
33. Lau A, Tsui E, Lee W.B: An Ontology-based Similarity Measurement for Problem-based Case Reasoning. Expert Systems with Applications, Vol. 36, No. 3, 6574-6579. (2009)
34. S.Chakrabartim, M.van den Berg and B.Dom: Focused Crawling: A New Approach to Topic-specific Web Resource Discovery. Computer Networks, Vol. 31, No. 11-16, 1623-1640. (1999)

Ying Wang was born in 1981. She is a lecturer at the Jilin University and a CCF member. She received her Ph.D. degree from Jilin University. Her research area is Web Information Mining, Ontology and Web search engine.

Huilai Li was born in 1962. He is a professor and doctoral supervisor at the Jilin University. His research areas are partial Differential Equations.

Wanli Zuo was born in 1957. He is a professor and doctoral supervisor at the Jilin University and a CCF senior member. His research areas are database, data mining and Web search engine.

Fengling He was born in 1962. He is a professor at the Jilin University and a CCF senior member. His research areas are database, data mining and Web search engine.

Xin Wang was born in 1981. He is a lecturer at the Changchun Institute of Technology. His research area is Web Information Mining and Ontology.

Kerui Chen was born in 1983. She received her Ph.D. degree from Jilin University. Her research area is Web Information Mining, Ontology and Web search engine.

Received: March 22, 2010; Accepted: January 13, 2011.