Special issue: Leading ICT technologies in the Information Explosion

Research Paper

Building web page collections efficiently exploiting local surrounding pages

Yuxin WANG[1,*1] and Keizo OYAMA[2]
[1] *Information Technology Center, University of Tokyo*
[2] *National Institute of Informatics*
[2] *The Graduate School for Advanced Studies (SOKENDAI)*

ABSTRACT
This paper describes a method for building a high-quality web page collection with a reduced manual assessment cost that exploits local surrounding pages. Effectiveness of the method is shown through experiments using a researcher's homepage as an example of the target categories. The method consists of two processes: rough filtering and accurate classification. In both processes, we introduce a logical page group structure concept that is represented by the relation between an entry page and its surrounding pages based on their connection type and relative URL directory level, and use the contents of local surrounding pages according to that concept. For the first process, we propose a very efficient method for comprehensively gathering all potential researchers' homepages from the web using property-based keyword lists. Four kinds of page group models (PGMs) based on the page group structure were used for merging the keywords from the surrounding pages. Although a lot of noise pages are included if we use keywords in the surrounding pages without considering the page group structure, the experimental results show that our method can reduce the increase of noise pages to an allowable level and can gather a significant number of the positive pages that could not be gathered using a single-page-based method. For the second process, we propose composing a three-grade classifier using two base classifiers: precision-assured and recall-assured. It classifies the input to assured positive, assured negative, and uncertain pages, where the uncertain pages need a manual assessment, so that the collection quality required by an application can be assured. Each of the base classifiers is further composed of a surrounding page classifier (SC) and an entry page classifier (EC). The SC selects likely component pages and the EC classifies the entry pages using information from both the entry page and the likely component pages. An evident performance improvement of the base classifiers by the introduction of the SC is shown through experiments. Then, the reduction of the number of uncertain pages is evaluated and the effectiveness of the proposed method is shown.

KEYWORDS
Web page collections, page group model, logical page group structure, three-grade classifier, quality assurance, precision and recall

1 Introduction

High quality scholarly information services used to be maintained through a lot of human work, but with much less time today, thanks to electronic publishing technology. However, the Web is becoming more and

Received September 17, 2008; Revised January 15, 2009; Accepted January 18, 2009.

[*1] The work presented in this paper was partly done when Ms. Wang was with the National Institute of Informatics. She is currently with Team Lab Corp.

[2)] *oyama@nii.ac.jp*

DOI: 10.2201/NiiPi.2009.6.4

more important as a potential information source that can add value to such services. Then, what is required first is to create a web page collection with assured recall and precision (called collection quality, in this paper) set by the service requirements. However, the collection quality cannot be reached using only automatic processing, even with state-of-the-art classification technology, and manual assessment is still indispensable. The recall requirement has an especially large effect on the assessment cost.

Many researchers have investigated search and classification techniques for web pages, etc. However, most of them are of the best-effort type and pay no attention to quality assurance. Thus, we tried to create a classification method to efficiently build a web page collection that can assure both a given high recall and a given high precision. As an example of such collections, we focused on a researcher's homepage in this paper.

Some research works on web page classification have shown that it is generally effective to use features that exploit link, directory, and document tag structures. However, these techniques are not expected to be effective for improving the performance at a very high recall.

However, taking into account that a set of related information is often presented on a set of interconnected pages (namely, a logical page group), the contents in surrounding pages must be considered in addition to the content in each entry page. This approach is expected to be potentially effective for improving the performance at very high recall, although no previous work has actually presented such promising results. Therefore, we have been studying a method to exploit the logical page group structures [1]–[4].

In the current work, we focus on the surrounding pages placed in the same URL directory structure and that are either explicitly or implicitly linked. When we use classification techniques, we regard each web page as a unit. We gather information from its surrounding pages taking the local link connections and directory levels into account, and use it together with the information on the page itself.

Since the amount of web pages is very large, one problem arising from this method is a high computing cost for the feature extraction. Therefore, we split the whole process into two sequential processes: (1) rough filtering for efficiently narrowing down the candidate page amount with a very high recall, and (2) an accurate classification for classifying the candidate target pages into three grades.

We exploit the information in the surrounding pages in both processes so that we achieve a high classification performance, especially in terms of recall. However, the usage styles slightly differ between the two processes, and the formal definitions are given in each section.

The rest of this paper is organized as follows. The related works are introduced in Sec. 2. Section 3 describes the overall scheme of the proposed method, consisting of rough filtering and accurate classification. The details and experimental results of the rough filtering and the accurate classification are presented in Secs. 4 and 5, respectively. Section 6 presents the evaluation on the reduction of the manual assessment cost with the proposed scheme using the results from the previous sections. Finally, we conclude our work in Sec. 7.

2 Related works

The method proposed in this paper belongs to the web page classification domain, and is closely related to the web page search and clustering domains. In these domains, what information sources to use and how to use them are two major problems.

The previous works have tried to exploit the textual contents in each page, and also various web-related information sources [5], such as html tags [6]–[9], URLs [7] [9]–[11], subgraphs of web pages [12] [13], directory structures [13] [14], anchor texts [6]–[8], contents of globally link-related pages [8] [15]–[17], and contents of local surrounding pages [7] [12]–[14].

All of these information sources except for the last one, are used to capture the features that are characteristic to the targeted pages, and are effective at emphasizing the highly probable pages. The last one, contrarily, is used to collect information dispersed over a logical page group, and is effective for comprehensively gathering potential pages. However, this last one tends to increase the amount of noise, and no clear performance improvement has been obtained by the previous works.

Nevertheless, since comprehensiveness (or recall) is the key factor for the quality assurance of a web page collection, we are mainly looking to exploit the last one, i.e., the surrounding pages as information sources.

In the rest of this section, we will introduce several previous works that have contrived to exploit them.

Sun et al. [13] proposed a method to first classify each page based on its content and then to iteratively classify web page subtrees by combining the previous results taking into consideration other information sources, such as the link and directory structures. Although they achieved rather good results, their methods require extra training data of support pages in addition to the main pages. In addition, when an entry page contains no textual information but only hyperlinks, their approach will not work.

Masada et al. [12] proposed a method to first cluster web pages based on their link structures, etc., and then to merge the score (or weight) of each content word to generate the document vector. However, the effectiveness of this proposal was limited, probably because it also merges many irrelevant words from the surrounding pages.

Yang et al. [14] defined five hypertext regularities and tested how their presence influenced the classification performance. Using the co-referencing regularity among the others, they treated all the content words of the surrounding pages together using Naïve Bayes and *k*-nearest neighbor, while they used First Order Inductive Learner (FOIL) to treat the content words separately for each surrounding page. However, the performance suffered for all cases.

Craven et al. [7] compared the FOIL with Predicate Invention for Large Feature Spaces (FOIL-PILFS) to a FOIL using much richer features and proved the algorithm's effectiveness. However, since they made no comparison between the different features, the effectiveness of the contents of the surrounding pages is unknown.

Many other works also tested the features using the contents from surrounding pages, but the results showed no clear performance improvement or even showed performance degradation.

We, too, exploited the contents in the surrounding pages while taking the local link structures into account, but using a different approach [1] [2]. We introduced a model of a logical page group structure (namely, page group model) that was based on the combination of a local link connection and the relative directory level. The contents are gathered and combined according to the page group model so that the features of each page, whether presented as a single page or as a logical page group, can be properly represented.

In addition, we introduced, for the first time, a mechanism to explicitly filter noisy surrounding pages that took into consideration the various features representing the relationships between an entry page and a surrounding page [4], and then we use the contents of the remaining pages (namely, likely component pages). No previous work has proposed such a mechanism.

In addition, we proposed a framework to assure the high quality required by practical applications [3]. We approached this problem by combining multiple filtering and classification components. No previous work has explicitly dealt with the issue of quality assurance for a document collection.

In this paper, we show the effects of the scheme introduced in [3] when combined with our methods shown in [1] [2] and [4].

3 Overall scheme

The overall scheme of the proposed framework is shown in Fig. 1. It contains two processes: rough filtering [1] and accurate classification [4].

The rough filtering process is used for efficiently narrowing down the amount of candidate pages with a very high recall from the Web [1]. The input is whole web pages and the output is only the candidate pages satisfying the recall required by its application with a sufficient margin. We set the performance required for the rough filtering to at least 98% in the current work. Precision does not matter that much, but a smaller amount of output pages is desirable under the constraints put on the recall.

Although we use a static web data corpus for our experiments in this work, the rough filtering process can be merged with a web crawler (or robot) for real applications. So, this might seem like it is similar to focused crawling, but it differs in several aspects. First, focused crawlers predict the relevance of each page before fetching while rough filtering does not. Consequently, it works only with comprehensive crawlers. Second, focused crawlers can handle only one category at a time while rough filtering can handle virtually as many categories as you like at a time.

The accurate classification is for classifying the candidate pages [4], i.e., the output from the rough filtering, into three grades: "assured positive", "assured negative", and "uncertain". For an example of the quality requirement, we set the recall to at least 95% (regarding both the assured positive and uncertain as positive) and

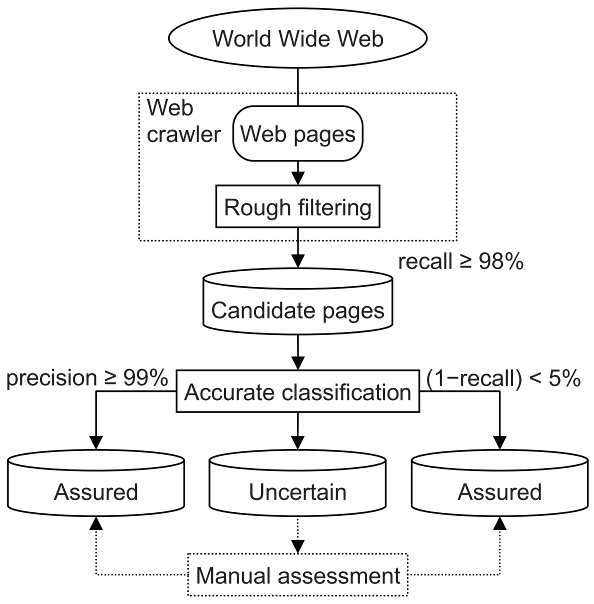

Fig. 1 Overall scheme of proposed method.

the precision to at least 99% (regarding only the assured positive as positive) in the current work.

Even when using the state-of-the-art classification techniques, it is impossible to make the "uncertain" part completely empty solely by automatic computer processing. Therefore, manual assessment is indispensable to assuring collection quality, and a relatively high computer processing cost is allowed for the accurate classification in order to reduce the number of uncertain pages.

4 Rough filtering

4.1 Structure of rough filtering

Rough filtering uses a set of property-based keyword lists (described in Subsec. 4.2) and several kinds of page group models (described in Subsec. 4.3). Fig. 2 illustrates its conceptual structure.

For the first, a document vector is generated from every web page. A document vector consists of binary values, each of which corresponds to a keyword list and represents if any of the keywords in the keyword list is present in the web page. Next, for each of the page group models, the document vectors are merged by making a logical sum of each vector element. In this process, only the elements corresponding to the keyword lists of the suitable types for each page group model are used (in the figure, the ignored elements are indicated with 'x'). They are further merged with the entry page's document vector to compose a final document vector (namely, virtual document vector). Here, a conceptual document represented by the virtual document vector is called a virtual entry page, and the process to merge the document vectors is called keyword propagation. Finally, scores of virtual entry pages are obtained by counting the number of 1's in the virtual document vector (or, L1 norm), and those that scored more than or equal to a threshold score are output. The threshold score will be selected taking into account the evaluation results so that the recall satisfies the requirement (98% in the current work), and the output amount is reasonable.

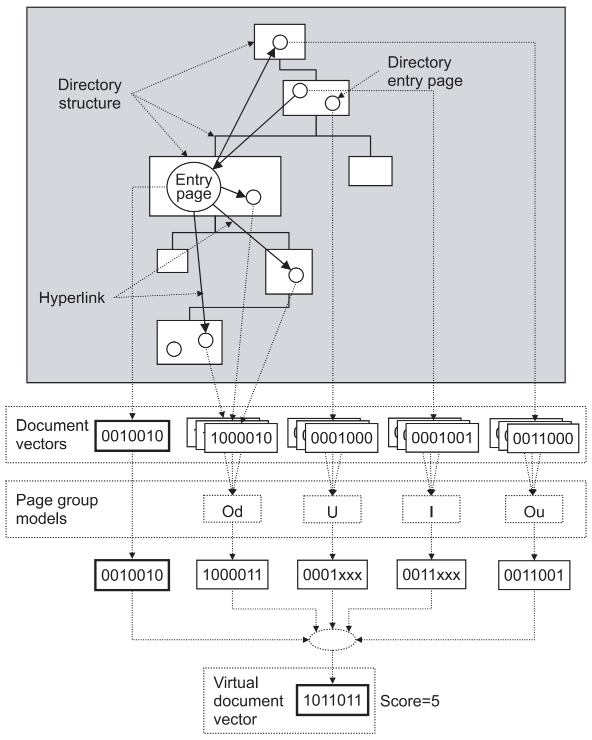

Fig. 2 Structure of the rough filtering.

4.2 Property-based keyword lists

Although the styles and structures of homepages tend to greatly differ and the presentations are very diverse, they usually contain several basic information elements that are common to homepages in the same category. Therefore, we introduce property-based keyword lists representing the common properties in a category ("researcher" in the current work), expecting that a certain number (not necessarily all) of them are included in each entry page or in its surrounding pages.

We manually gathered keywords, categorized them, and eventually obtained 12 keyword lists containing 86 keywords for the researcher's homepage category in our previous work. We mainly used property-name-related terms. Property-value-related terms are used only when they can be enumerated within a small number; otherwise their maintenance would require a lot of effort.

Each of the keyword lists is then assigned a type, either organization-related or non-organization-related. Keyword lists corresponding to the properties common to the members in the same organization are designated organization-related, while keyword lists corresponding to individual researcher's properties are designated non-organization-related. The types and meanings of these 12 keyword lists are listed in Table 1 along with some keyword examples. Note that the actual keywords are in Japanese.

4.3 Page group model

Taking into account the logical page group structure within a site, we propose four simple page group models (PGMs) corresponding to four kinds of link and directory structures. The definitions of the surrounding pages and PGMs are given in Tables 2 and 3, respectively.

A single page model (SPM) and a single site model (SSM) are used as the baselines for evaluating the effectiveness of the proposed PGMs. **Od**, **Ou**, **I**, and **U**

Table 1 Property-based keyword lists and keyword samples.

Type	Keyword lists	Keyword samples*
Non-organization-related	general word	research, study
	research topic	research topic, theme
	title	doctor, professor
	position	present position, duty
	history	biography, personal history
	achievement	paper, bibliography
	lecture	course, seminar
	academic society	academic society, regular member
Organization-related	major	major, specialty, research field
	member	staff, member
	organization	university, institute, school
	section	section, department

* Original keywords are in Japanese.

Table 2 Definition of surrounding pages.

Notations	Definition
$P_{out}(r)$	Set of pages having link from r in same site (r's out-linked pages)
$P_{in}(r)$	Set of pages having link to r in same site (r's in-linked pages)
$P_{lower}(r)$	Set of pages in directories lower in directory subtree of r (r's lower pages)
$P_{upper}(r)$	Set of pages in directories upper in directory path of r (r's upper pages)
$P_{dent}(r)$	Set of directory entry pages in r's directory path
$P_{top}(r)$	Set of site top page(s) of r

Table 3 Definition of page group models.

Models		Description	Propagated pages
SPM (baseline)		Single page model; no keyword propagation is used.	-
SSM (baseline)		Reference page group model; all out-linked pages in same site are used.	$P_{out}(r)$
Simple PGM	Od	PGM based on out-links downward; out-linked pages in directory subtree are used.	$P_{out}(r) \cap P_{lower}(r)$
	Ou	PGM based on out-links upward; out-linked pages in directory path are used.	$P_{out}(r) \cap P_{upper}(r)$
	I	PGM based on in-links upward; in-linked pages in directory path are used.	$P_{in}(r) \cap P_{upper}(r)$
	U	PGM based on directory entry pages; site top pages and entry pages of directory path are used.	$P_{dent}(r) \cup P_{top}(r)$
Modified PGM	Od@θ	Od with additional conditions on number of out-links; if there are too many out-links, Od is not used.	If $N_{Lod}(r) \leq \theta$, same as Od; otherwise, same as SPM.*
	Ou#, I#,U#	Ou, I and U, each propagating organization-related keywords only	Same as Ou, I, and U for organization-related keywords; for others, same as SPM.

* $N_{Lod}(r)$ denotes the number of links from page r to the pages in the same directory and in directories lower in the directory subtree.

are four simple PGMs. **Od** is intended to propagate all kinds of keywords from out-linked component pages in the directory subtree (actually, we ignore the directories that are three or more levels lower based on a preliminary experiment). **Ou**, **I**, and **U** are intended to propagate all kinds of keywords from out-linked pages, in-linked pages, and directory entry pages, respectively, in the upper directory path (actually, we ignore the directories in the four or more levels higher too).

Since simple PGMs usually propagate many irrelevant keywords and consequently include many noise pages, we devised modified PGMs to reduce such noises, whereas keeping useful keywords propagated. **Od**@θ is a modified PGM derived from **Od** with the intention of excluding irrelevant pages, based on the observation that one of the noise sources is large groups of mutually linked pages within a directory, and that an entry page having many out-links always contains sufficient keywords in itself.

Ou#, **I#**, and **U#** are modified PGMs derived from respective simple PGMs with the intention of excluding irrelevant keywords based on the observation that non-organization-related keywords related to the researcher rarely exist in the upper directory hierarchies. Therefore, only organization-related keywords are propagated with these PGMs. Since any single PGM can use only a part of the available component pages and cannot collect sufficient information, we use all of the modified PGMs combined.

4.4 Experiments

4.4.1 Data set

For the experiments, we used NW100G-01, a corpus of 100 GB web document data containing 11,038,720 web pages gathered from the '.jp' domain for the WEB Tasks of the Third and Fourth NTCIR Workshops [18] [19].

A sample data set used for the rough filtering was prepared from NW100G-01 by the authors. We first collected 113,380 pages containing some typical Japanese family names and randomly selected 11,338 pages, 10% from the total. Each of the pages was then manually assessed by the authors according to its content and, if necessary, the contents of the surrounding pages. Consequently, we obtained 426 positive samples and 10,912 negative samples. We call this set of 11,338 pages ResJ-1, hereinafter.

4.4.2 Filtering experiment

We first experimented on individual simple PGMs. The results show that all the simple PGMs increase the page amounts more than the SPMs. Then, we experimented on the modified PGMs and confirmed their effectiveness at reducing the page amount without degrading the recall. Finally, we experimented on combinations of PGMs.

Three results (converted to the case where entire corpus is processed) are shown in Fig. 3 together with SPM and SSM. For each point, the x-value is the number of pages in the corpus that scored at least a threshold score i. The y-value is the recall defined by np(i)/Np, where Np is the total number of positive sample data and np(i) is the number of positive sample data that scored at least i ($1 \leq i \leq 12$). For each data sequence, the most up and right data corresponds to $i = 1$, and every next one corresponds to a threshold score incremented by 1. In general, a higher recall and less page amount indicate a better performance; however, we put priority on recall.

The three PGM combinations shown in Fig. 3 are:

C1 : **Od**@5,**Ou#**,**I#**,**U#**,

C2 : **Od**@10,**Ou#**,**I#**,**U#**,

C3 : **Od**@20,**Ou#**,**I#**,**U#**.

All of them use all four modified PGMs with $\theta = 5$, 10, and 20 for **Od**@θ, respectively.

The results show that even the best performing PGM combination **C1** is inferior to SPM in all the recall ranges except for 100%. We will investigate into this issue in the next subsection.

In addition, the proposed method reduced the number of pages to nearly half compared to SSM, a naive method to propagate keywords from the surrounding pages. This demonstrates the effectiveness of using PGMs.

4.4.3 Additional experiments and discussions

(1) Overlooked positive pages

Since we use test data in which the assessment was manually done by the authors, we executed the assessment prior to the experiments so that any arbitrariness in assessment could be excluded and then as much of the subjectivity and fairness as possible could then be secured. However, while we were conducting the failure analysis, we found a considerable number of samples that were in truth positive were assessed negative due to some error. After analyzing this problem, we found that the main reason for the error was that the assessor overlooked pages that contained too little information although rich information is presented in its component pages. Therefore, we decided to obtain a final evaluation result by compensating the results shown in Subsubsec. 4.4.2 taking into account the additional assessment as described below.

We additionally assessed the pages in the ResJ-1 data that scored less than a three using SPM and no less than a four using **C3**, and found 13 new positive data, which were overlooked in the manual assessment. We thoroughly checked them and the results showed the ability of the proposed method to find the positive pages that cannot be gathered by using SPM.

Taking into account the 13 newly-found positive data, the recall of the SPM at threshold scores of two and three should be corrected from 99.8% to 97.0% and from 99.1% to 96.1%, respectively. Based on the corrected results, the proposed methods outperformed the SPM with a 5% significance.

In addition, four of the newly-found positive pages cannot be gathered using SPM even if the threshold score is set to one. This means SPM can hardly achieve the recall goal for any feasible page amount.

Furthermore, a failure analysis on all three positive

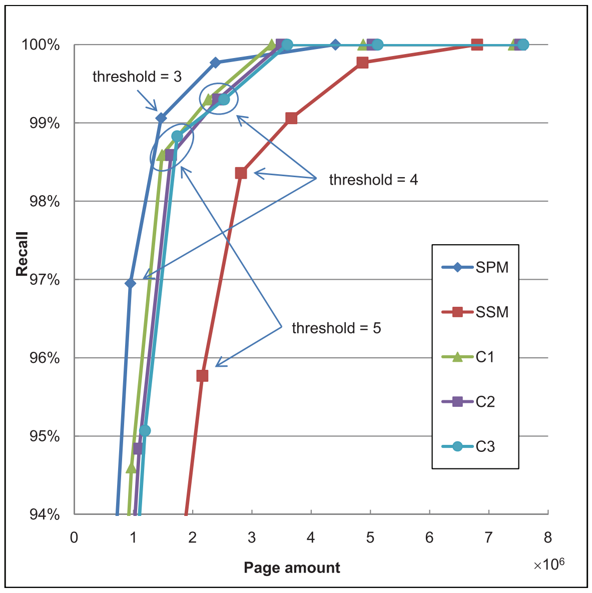

Fig. 3 Performance of well-performing PGM combinations.

Table 4 Comparison of pages output from rough filtering of two data sets.

Data set	Total amount	Output amount	Output proportion
NW100G-01	11,038,720	2,530,850	22.9%
NW1000G-04	95,870,352	14,128,826	14.7%

pages that scored only a three using **C1** to **C3** revealed that they are of a similar pattern. Although they have hyperlinks to the researchers' personal homepages, our method cannot exploit them because they exist on separate sites. This fact supports that for applications where only an informative homepage suffices when multiple homepages exist for a researcher, the proposed method works if their personal homepages are crawled.

Since we are to guarantee a 98% recall in the current work, we can use any of **C1** to **C3** with a threshold score of four. We will eventually select **C2** as the most appropriate one for the current goal, because the recalls of **C2** and **C3** at a threshold score of four are the same (with less pages for **C2** than for **C3**) and are slightly higher than that of **C1**.

(2) Applicability to larger data set

We used another corpus for evaluating the effectiveness of the rough filtering: NW1000G-04, 1.36 TB (1.5×10^{12} byte) web document data containing 95,870,352 web pages, which was created for the WEB Task at the Fifth NTCIR Workshop [20].

We used the PGM combination **C3** and the candidate pages are gathered using a threshold score of four. Table 4 lists the comparison of experiment results.

When comparing the proportions of the pages output from the rough filtering, it is shown that the output pages can be reduced more for larger data sets (less than 15% of the corpus). Therefore, rough filtering is not only applicable, but is also more efficient for a larger data set. Since we have not assessed the output, its effectiveness is yet to be investigated.

(3) Filtering effectiveness

To evaluate the effectiveness of rough filtering, 20,846 pages were randomly sampled from the output, and then each page was manually assessed based on its content (and the content of its surrounding pages, if necessary). Consequently, we obtained 480 positive samples and 20,366 negative samples. When a researcher's information is provided in a set of more than one interlinked pages in a site, only the entry page is labeled positive and the others are labeled negative.

Then, its precision is calculated as 480/20846= 2.3%, which is rather low for a typical document filter. Although efficiency is given priority, some more powerful (but still efficient) filtering methods should be investigated in the future. To keep the number of keywords reasonable for the constraint described in Subsec. 4.5, a rule based machine learning method (e.g., FOIL) that can handle a relation between keywords and page groups is promising.

4.5 Implementation issues

Rough filtering seems to require a large computational cost because of the keyword propagation processing. However, it can be implemented efficiently in the following way.

First, keyword inclusion in a web page can be judged very efficiently. Since keywords are fixed in advance, a very efficient multiple string matching algorithm can be used. No natural language processing such as segmentation or POS tagging is necessary. Since the web robot needs to scan each fetched page for extracting the hyperlinks to find new pages, keyword matching can be easily and very efficiently embedded in the robot process.

Next, since keyword propagation is limited to within each site, the generation of virtual document vectors is processed in parallel site by site. This processing can be embedded in the robot process or can be implemented as a separate process. However, taking into consideration that keyword propagation along an out-link (i.e., backward keyword propagation) must be prolonged until the out-linked page is fetched and that the processing requires rather heavy database operations if embedded in the robot process, it is reasonable to implement it as a separate batch process that is run after a site has been crawled.

Web robots generally use database management systems to maintain records of web pages corresponding to URLs already found. Each record holds the URL, the time of the last crawl, the time of the next scheduled crawl, etc. When a page has been fetched, the page content is scanned to extract the hyperlinks, and the corresponding record is inserted if it is not stored yet.

For the current work, in addition to the usual output, the robot should output, for each fetched web page p, its document identifier, document vector d_p, which is the result of the keyword matching, and a list of the document identifiers of the out-linked web page set Q. For the current keyword lists, d_p can be represented with 12 bits.

The virtual document vector generation and score calculation require a look up of the document vectors corresponding to the in-links, out-links, and directory entry relations. However, since its processing is executed site by site, all the document vectors and the virtual document vectors can usually be stored on the core memory. Consequently, it can be executed in a time proportional to the total number of considered links, which is approximately proportional to the number of documents in a site.

5 Accurate classification

5.1 Surrounding page group

We use a model of surrounding page groups that is slightly different from the one used in the rough filtering, although the basic idea for categorizing the surrounding pages based on the link and directory level is the same.

Formally, a surrounding page s is a page related to the entry page r by one or more of the connection type c described below and placed in a directory in r's directory path to the site root or in r's directory subtree. As shown in Table 5, c is either an *out* (r has an out-link to s), *in* (r has an in-link from s), or *dent* (s is a directory entry page). s's relative directory level l is either of *same* (s is placed in the same directory as r), *lower* (s is placed lower in the directory subtree of r), or *upper* (s is placed higher in the directory path of r).

A surrounding page group (SPG) denoted as $G_{c,l}$ is a group of surrounding pages sharing a connection type c and a relative directory level l. Table 6 shows an overview of SPG. An "*all*" for c and l means the attribute is ignored. Note multiple c may apply to a surrounding page s and hence an s may belong to multiple $G_{c,l}$. We omit $G_{dent,lower}$ because it almost overlaps $G_{out,lower}$. The entry page r itself is assigned an independent SPG G_{cur}.

Each $G_{c,l}$ has its own potential meaning in a logical page group. For example, $G_{in,lower}$ consists of in-linked pages in lower directories, each of which might represent a component page having a back link to the entry page, and $G_{dent,upper}$ consists of directory entry pages in the upper directories, each of which might represent an entry page of the organization the researcher belongs to.

5.2 Composition of accurate classification

Fig. 4 shows the composition of the proposed method (95% recall and 99% precision are the example quality requirements in the current work). We use two base classifiers to construct a three-grade classifier. The first is a recall-assured classifier (RAC) that assures the target recall with the highest possible precision, and the second is a precision-assured classifier (PAC) that assures the target precision with the highest possible recall.

The pages output from the rough filtering (candidate pages) are first input to the RAC and its negative predictions are classified as "assured negative". The rest are then input to the PAC and its positive predictions are classified as "assured positive". The remaining pages are classified as "uncertain" and they will be manually assessed.

Each of RAC and PAC are further composed of two component classifiers, i.e., a surrounding page classifier (SC) and an entry page classifier (EC) as shown in Fig. 5. The input data of SC consists of all pairs of entry

Table 5 Summary of surrounding page categories.

Attribute	Category	Definition
Connection type c	*out*	r's out-linked pages
	in	r's in-linking pages
	dent	directory entry pages
Relative directory level l	*same*	pages in the same directory as r
	lower	pages in the lower directory subtree of r
	upper	pages in the upper directory path of r

Table 6 Overview of surrounding page groups.

Relative directory level l	Connection type c				
	r	*out*	*in*	*dent*	(merged)
same	G_{cur}	$G_{out,same}$	$G_{in,same}$	$G_{dent,same}$	$G_{all,same}$
lower	–	$G_{out,lower}$	$G_{in,lower}$	–*	$G_{all,lower}$
upper	–	$G_{out,upper}$	$G_{in,upper}$	$G_{dent,upper}$	$G_{all,upper}$
(merged)	–	$G_{out,all}$	$G_{in,all}$	$G_{dent,all}$	$G_{all,all}$

* $G_{dent,lower}$ is omitted because it almost overlaps with $G_{out,lower}$.

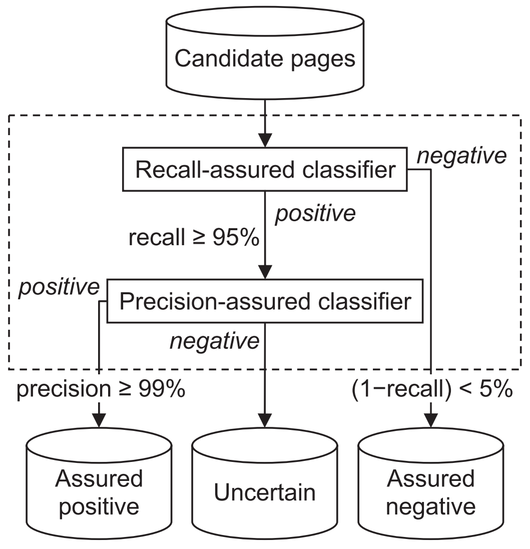

Fig. 4 Composition of the accurate classification.

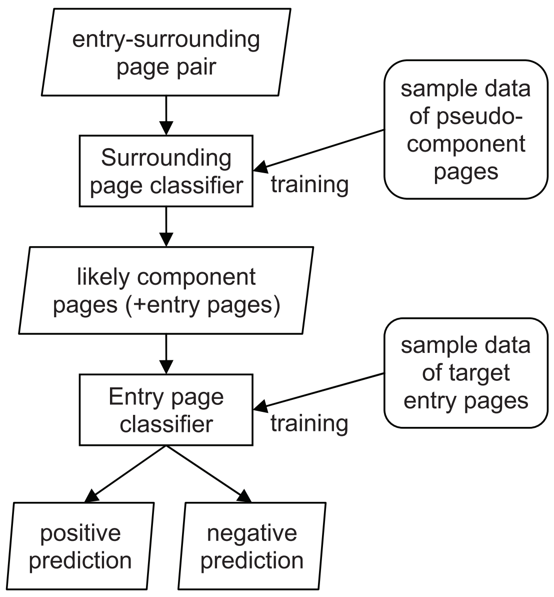
entry-surrounding page pair
Surrounding page classifier
sample data of pseudo-component pages
training
likely component pages (+entry pages)
Entry page classifier
sample data of target entry pages
training
positive prediction
negative prediction

Fig. 5 Overall configuration of base classifier.

pages and surrounding pages. The surrounding pages of its positive output are deemed as likely component pages, which will be used in EC.

We used a support vector machine (SVM) for both of the two component classifiers, since SVM is shown to be efficient and effective for text classification. An SVM^{light} package[1)] with a linear kernel is used in the current work.

5.3 Surrounding page classifier

The SC is used for selecting the likely component pages of each entry page, so that noisy surrounding pages are filtered. It classifies each pair of an entry page and a surrounding page (entry-surrounding page pair, or E-S pair).

A certain amount of labeled E-S pair data is necessary to train the SC. However, to prepare them manually is very expensive. Thus, we try to use weakly labeled data that are automatically generated from manually labeled sample entry pages. Here, all the E-S pairs of positive entry pages are deemed "component pages" (positive) and all the others are "non-component pages" (negative). We call the said "component page" a "pseudo-component page", since it is not necessarily a true "component page".

For the SC, we use the following five types of features, regarding the relations between pages of each E-S pair. They are extracted by comparing the contents and URLs of the paired pages.

Link/URL related features :
Link type (in/out-link); URL format (relative/full path; with/without host name; directory entry page; relative directory level); metrics of term co-occurrences in URL; etc.

Anchor-text-content related features :
Frequency of each anchor text term in various regions of the pages.

Shared-content-word related features :
Metrics on co-occurrences of several types of terms (person/organization names; general terms; terms in title elements) in various regions of the pages (several important tag elements; page top n words), etc.

Page-style related features :
Match/mismatch of background color/image, text color, and style sheet.

URL/anchor-text term related features :
Occurrence of each feature term in the URL and in the anchor texts, respectively, for each paired page (feature terms are extracted from the URL and from the anchor text part of all sample pages, respectively).

In addition, an SC can be composed of one or multiple component classifiers, each of which handles E-S pairs corresponding to each SPG. We tested several compositions and found that the following performs the best:

3-dir :
Uses three component classifiers for SPGs corresponding to the respective relative directory levels, i.e., $G_{all,l}, l \in \{same, lower, upper\}$.

We tune the SC so that informative pages are passed and noisy pages are filtered, based on the overall classification performance including the EC, because to train the SC, we use the sample data of the pseudo-component pages, in which a certain amount of noisy surrounding pages of positive entry pages are deemed positive, and vice versa for negative entry pages. Here, we assume the fraction of noisy pages in the pseudo-component pages (false positive) is approximately constant, and we therefore use the recall on the training data as a tuning parameter. Note the recall is easily controlled by the SVM threshold.

In addition, in order to cope with the imbalanced ratio of the positive to negative training data in SC, we set their inverse ratios to the j-option value of SVM^{light} (weight on error cost of a positive sample against that of a negative sample). For other options, default parameters are used.

[1)] SVM^{light} Support Vector Machine. http://svmlight.joachims.org/.

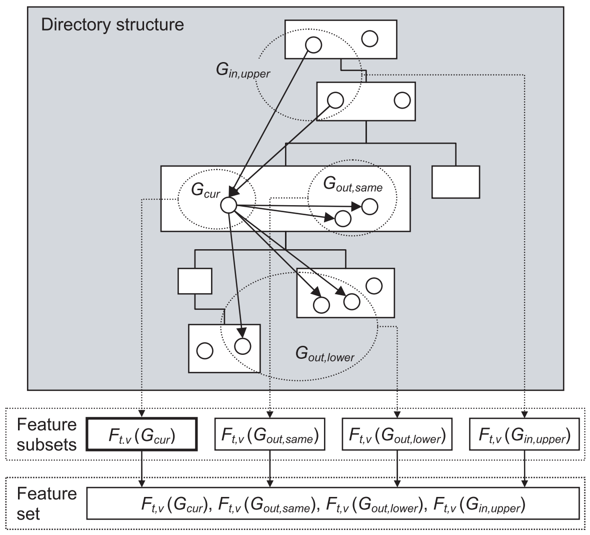

Fig. 6 Example composition of feature sets for entry page classifier.

5.4 Entry page classifier

The EC is to classify each entry page using the features obtained from the content of the entry page itself and its likely component pages selected by the SC.

The feature set used by EC is generated from the likely component pages, as shown in Fig. 6. The likely component pages are grouped into SPGs, and from each of them, we extract a feature subset $F_{t,v}(g) = \{f_{t,v}(g, w_t)|w_t \in W_t\}$, where $f_{t,v}(g, w_t)$ denotes a textual feature, t indicates a text type *plain* (plain-text-based) or *tagged* (tagged-text-based), v indicates a value type *binary* or *real*, g denotes an SPG, and $w_t \in W_t$ denotes a feature word. Then, the feature subsets are concatenated to compose the final feature set.

For text type *plain*, a feature is extracted from the textual content excluding the tags, scripts, comments, etc. We use the top 2,000 feature words ranked by mutual information. For text type *tagged*, a feature is extracted from "*text*" segments that match either pattern "> *text* <" or "<img. . . alt= "*text*". . . >" and that are not more than 16 bytes long with the spaces omitted. The obtained words with not less than 1% file frequency are used as feature words. The suffix "**_t**" of the EC name used in Subsec. 5.5 indicates the additional use of *tagged*.

For a *binary* value type, each feature represents the presence of the feature word in g, while for a *real* value type, each feature represents the proportion of the pages containing the feature word within g. The suffix "**_r**" of the EC name used in Subsec. 5.5 indicates the use of *real*; otherwise *binary* is used.

We tune the EC independently from the SC in the final stage with the two cost related parameters of SVM^{light}, i.e., c (trade-off between training error and margin) and j (mentioned in Subsec. 5.3).

5.5 Experiments

In the experiments, we used SVM^{light} with linear kernel for both SC and EC. In order to better understand the effectiveness of SC, the performance of the proposed composition is compared with compositions without the SC. Five-fold cross validation is applied for all experiments.

5.5.1 Sample data

We prepared the sample data set named ResJ-2 from NW100G-01, using the 20,846 pages described in Subsec. 4.4.3 (3), which were randomly sampled from the rough filtering output and manually assessed (eventually, 480 positive and 20,366 negative samples) and the 426 positive samples described in Subsec. 4.4.1, which were used in the experiments of the rough filtering, altogether (906 positive, 20,366 negative and 21,272 in all).

The weakly labeled E-S pairs were automatically derived from the ResJ-2 data set, obtaining 4,200 positive and 164,075 negative samples. This means that in total, 906 positive entry pages have 4,200 surrounding pages and 20,366 negative entry pages have 164,075 surrounding pages, including any overlaps.

5.5.2 Experimental results

We tested various SC and EC compositions. However, only the top performing one and some typical ones for comparison are shown: **baseline**, **U_t**, **A_tr**, and **A_t+3-dir**. For EC, the **baseline** uses a *plain* and *binary* feature subset from only G_{cur}, **U** stands for using the feature subsets from G_{cur}, $G_{out,upper}$, $G_{in,upper}$, and $G_{dent,upper}$ together, and **A** stands for using those from G_{cur} and all $G_{c,l}$, $c \in \{out, in, dent\}$, $l \in \{upper, same, lower\}$ together. As described in Subsec. 5.4, suffixes "**_t**" and "**_r**" indicate the additional use of *tagged* features and the alternative use of *real* feature values, respectively. For SC, **3-dir** indicates the use of the proposed composition shown in Subsec. 5.3; otherwise no SC is used.

For all the experiments, the performance of each classifier is evaluated by precision, recall, or F-measure defined as:

$$\text{precision} = \frac{|P_P \cap P_T|}{|P_P|}$$

$$\text{recall} = \frac{|P_P \cap P_T|}{|P_T|}$$

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

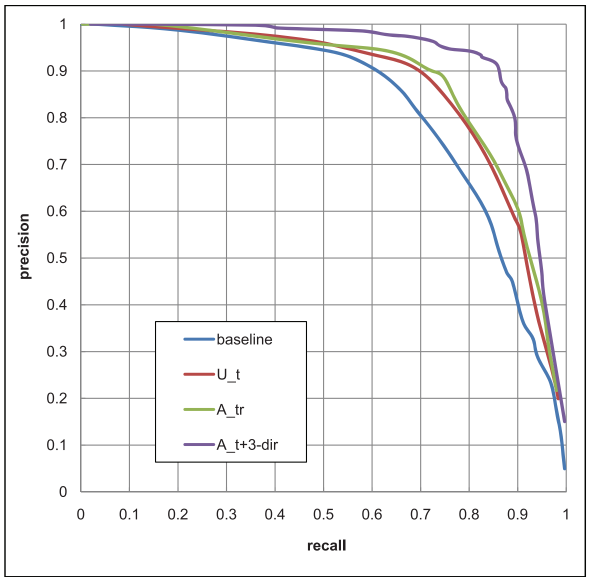

Fig. 7 Performances of base classifier.

Table 7 Performance of well performed classifier composition.

Classifier composition	F-measure	Recall at precision p			Precision at recall r		
		p =.98	.99	.995	r =.90	.95	.98
baseline	.748	.255	.167	.099	.406	.270	.173
U_t	.799	.326	.206	.147	.570	.338	.215
A_tr	.811	.310	.230	.190	.604	.404	.218
A_t+3-dir	.887	.614	.452	.386	.745	.466	.255

where P_P is positive predictions and P_T is positive samples.

The overall performance is shown in Fig. 7 and their performance measures are shown in Table 7. Each curve in Fig. 7 is drawn by connecting the results of the well-performing tuning parameters. Note that the baseline is a simple SVM classifier that doesn't use SC or SPGs, **U_t** and **A_tr** use SPGs for EC but no SC, and **A_t+3-dir** uses all SPGs for EC and 3-dir composition for SC.

The results show that, in all recall-precision ranges, the proposed base classifier (**A_t+3-dir**) greatly outperformed the baseline. In addition, comparing its result with the ones not using SC (**U_t** and **A_tr**), we found that recall at very high precision is substantially improved.

The effects of this performance improvement will be further discussed in Sec. 6 in terms of the cost of the manual assessment.

5.6 Implementation issue

SC requires feature extraction and classification on every E-S pair. EC also requires feature extraction from all the entry pages and surrounding pages. According to our experience, an entry page has five to fifteen surrounding pages on average depending on the data collections. Even though the number of candidate pages is much smaller than that of a whole web collection, the processing cost is still huge. On the other hand, for SVM, learning is a computation intensive process that needs to be executed only once, and classification is a straight forward process and is very efficient. Therefore, feature extraction is the dominant part of the processing cost.

The feature extraction needs natural language processing such as segmentation and POS tagging, which is the most time consuming. However, each page can be processed in parallel and can be easily executed with PC clusters that are now widely available.

For feature generation based on SPGs, as in the case of the rough filtering, each site can be processed in parallel. Each data may be processed in parallel, but a careful analysis on the data access pattern is required. For most cases, site level parallelism is sufficient. If necessary, a site might be divided into several directory subtrees and processed in parallel.

6 Evaluation of overall system

In order to better understand the effect of the proposed method in reducing the total amount of pages that require manual assessment (i.e., the size of the uncertain class), we compared the best performing three-grade classifier composition to the baseline. Table 8 shows the estimated page numbers of the classification output from the three-grade classifiers for some quality requirements. When comparing the "uncertain" class sizes, the proposed method reduces the amount from 62.1% to 29.7% depending on the required quality. The effect is especially large when the required quality is relaxed. Considering the large cost for the manual assessment, this reduction is quite important.

In addition, if we are to guarantee a 99% precision without manual assessment, only 16.7% of all positive data can be collected with the baseline, while with the proposed method, 45.2% can be collected, as seen in Table 7. If we can allow as low as a 90% precision, more than 85% of the positive data can be collected with the proposed method, whereas only 60% can be collected with the baseline, as seen in Fig. 7. There may be some applications for which this performance is sufficient.

Although the performance improvement viewed with these measures may seem only slight, its actual effect on real applications is rather large.

Table 8 Estimated page numbers of classification output from NW100G-01.

		Requirement (precision/recall)		
		0.995/0.98	0.99/0.95	0.98/0.90
baseline	assured positive	4,776	8,097	12,490
	uncertain U_B	267,132	160,792	93,914
	assured negative	10,766,812	10,869,831	10,932,316
A_t+3-dir	assured positive	18,621	21,915	30,073
	uncertain U_A	165,849	75,939	27,913
	assured negative	10,854,249	10,940,866	10,980,733
Reduction ratio $\|U_A\|/\|U_B\|$		0.621	0.472	0.297

7 Conclusion

We proposed a realistic framework for assuring the quality of web page collections using two processes: rough filtering and accurate classification. In both processes, we introduced the concept of logical page group structure and demonstrated its effective uses for filtering and classifying web pages, where researcher's homepages are used as an example.

In the rough filtering, we described the method for comprehensively gathering all the potential researchers' homepages from the web very efficiently. We proposed a method of using property-based keyword lists combined with four page group models. Two key techniques were used to reduce the irrelevant keywords being propagated by considering the relations on the content and the structure between pages in each logical page group. Consequently, the page amount could be reduced down to approximately 23% of the entire corpus. The implementation issues were discussed taking into consideration how it can be integrated in a web crawler, and the future issues on performance (or precision) improvement are also presented.

In the accurate classification, we proposed a method for classifying web pages into three grades by combining two base classifiers: a precision-assured classifier and a recall-assured classifier. We also used the concept of a page group model for the base classifiers. We composed them using two component classifiers: a surrounding page classifier and an entry page classifier. Compared to a baseline, the classification performance throughout the recall-precision range was significantly improved.

By combining the rough filtering and the accurate classification, we can efficiently process vast amounts of web pages and reduce the manual assessment cost for building web page collections with assured collection quality.

We have also applied the composition of the base classifier to another commonly used test data named "Web-KB" in four categories (not shown in this paper), and have verified its effectiveness. Thus, we expect it to be effective also in many other categories, such as shopping, product catalogs, and so on, although this has yet to be verified. On the other hand, the rough filtering currently requires creating keyword lists manually, and in addition, its applicability to other domains is still unknown. However, in terms of information services, we hope high-quality web page collections could successfully be built using our method, and that they would be applied to various domain specific search engines with an assured high quality.

In conclusion, to tackle the diversity of web data by exploiting their rich web-based features in pursuit of a very high performance at less processing cost is a challenging problem. The method presented in this paper is considered to be a general framework for solving the same kind of problems and we hope that the method will be a benefit and contribute to related research on web information utilization.

Acknowledgement

This study was partially supported by a Grant-in-Aid for Scientific Research B (No. 18300037) from JSPS. We used the NW100G-01 and NW1000G-04 document data sets under permission from the National Institute of Informatics.

References

[1] Yuxin Wang and Keizo Oyama, "Combining page group structure and content for roughly filtering researchers' homepages with high recall," *IPSJ Transactions on Databases (IPSJ TOD)*, vol.47, no.SIG 8, pp.11–23, 2006.

[2] Yuxin Wang and Keizo Oyama, "Web page classification considering page group structure for building a high-quality homepage collection," *Proc. of Fourth International Conference on Web Information Systems and Technologies (WEBIST 2007)*, March 3–6, 2007, Barcelona, Spain, vol.WIA, pp.170–175, 2007.

[3] Yuxin Wang and Keizo Oyama, "Framework for building a high-quality web page collection considering page group structure," *Proc. of Joint 9th Asia-Pacific Web Conference (APWeb 2007) and 8th International Conference on Web-Age Information Management (WAIM 2007)*, Jun. 16–18, 2007, HuangShan, China, LNCS 4505, pp.95–107, 2007.

[4] Yuxin Wang and Keizo Oyama, "Web page classification exploiting surrounding pages with noisy page filtering," *Proc. of the 2008 International Conference on Data Mining (DMIN2008)*, Jul. 14–17, 2008, Las Vegas, Nevada, USA, pp.626–632, 2008.

[5] S. Chakrabarti, "Data mining for hypertext: a tutorial survey," *ACM SIGKDD Explorations*, vol.1, no.2, pp.1–

11, 2000.

[6] A. Sun, E.-P. Lim and W.-K. Ng, "Web classification using support vector machine," *Proc. of the fourth international workshop on web information and data management*, McLean, Virginia, USA, ACM Press, pp.96–99, 2002.

[7] M. Craven and S. Slattery, "Relational Learning with Statistical Predicate Invention: Better Models for Hypertext," *Machine Learning*, Springer, vol.43, no.1-2, pp.97–119, 2001.

[8] J. Sun, B. Zhang, Z. Chen, Y. Lu, C. Shi and W. Ma, "GE-CKO: A Method to Optimize Composite Kernels for Web Page Classification," *Proc. of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI2004)*, Sep. 2004, Beijing, China, pp.299–306, 2004.

[9] L. K. Shih and D. R. Karger, "Using URLs and table layout for web classification tasks," *Proc. of 13th International Conference on World Wide Web (WWW2004)*, May 17-22, 2004, New York, NY, USA, pp.193–202, 2004.

[10] M.-Y. Kan and H. O. N. Thi, "Fast webpage classification using URL features," *Proc. of 14th ACM International Conference on Information and Knowledge Management (CIKM'05)*, Oct. 2005, Bremen, Germany, pp.325–326, 2005.

[11] M.-Y. Kan, "Web Page Categorization without the Web Page," *Proc. of 13th World Wide Web Conference on Alternate track papers & posters (WWW2004)*, May 17–22, 2004, New York, NY, USA, pp.262–263, 2004.

[12] T. Masada, A. Takasu and J. Adachi, "Improving web search performance with hyperlink information," *IPSJ Transactions on Databases (IPSJ TOD)*, vol.46, no.SIG 8, pp.48–59, 2005.

[13] A. Sun and E.-P. Lim, "Web unit mining: finding and classifying subgraphs of web pages," *Proc. of 12th International Conference on Information and Knowledge Management (CIKM2003)*, Nov. 2003, New Orleans, Louisiana, USA, pp.108–115, 2003.

[14] Y. Yang, S. Slattery and R. Ghani, "A Study of Approaches to Hypertext Categorization," *Journal of Intelligent Information Systems*, vol.18, no.2–3, pp.219–241, 2002.

[15] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced hypertext categorization using hyperlinks," *Proc. of International Conference on Management of Data (SIGMOD'98)*, 1998, Seattle, WA, USA, pp.307–318, 1998.

[16] M. Chau, "Applying web analysis in web page filtering," *Proc. of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'04)*, Jun. 2004, Tucson, Arizona, USA, p. 376 (2004).

[17] E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pen-nock and G. W. Flake, "Using web structure for classifying and describing web pages," *Proc. of the 11th International World Wide Web Conference (WWW2002)*, May 2002, Honolulu, Hawaii, USA, pp.562–569, 2002.

[18] K. Eguchi, K. Oyama, E. Ishida, N. Kando and K. Kuriyama, "Overview of the web retrieval task at the third NTCIR Workshop," *NII Technical Report*, NII-2003-002E, National Institute of Informatics, 2003.

[19] K. Eguchi, K. Oyama, E. Ishida, N. Kando and K. Kuriyama, "Evaluation methods for web retrieval tasks considering hyperlink structure," *IEICE Transactions on Information and Systems*, vol.E86-D, no.9, pp.1804–1813, 2003.

[20] Keizo Oyama, Masao Takaku, Haruko Ishikawa, Akiko Aizawa and Hayato Yamana, "Overview of the NTCIR-5 WEB Navigational Retrieval Subtask 2 (Navi-2)," *Proc. of the Fifth NTCIR Workshop Meeting on Evaluation of Information Access Technologies*, Dec. 6–9, 2005, Tokyo, Japan, pp.423–442, 2005.

Yuxin WANG

Yuxin WANG received B.E and M.E from East China Normal University, Shanghai, China in 1990 and 1993, respectively, and Ph.D. from The Graduate University of Advanced Studies (SOKENDAI), Japan in 2006. She became a project researcher at NII in 2006, a research fellow at the University of Tokyo in 2007, and joined Team Lab Corp. in 2008. Her research interests include web information utilization, information retrieval and natural language processing.

Keizo OYAMA

Keizo OYAMA received the B.E., M.E. and Dr. Eng. from the University of Tokyo in 1980, 1982 and 1985, respectively. He is a professor of National Institute of Informatics (NII), Japan and the Graduate University for Advanced Studies (SOKENDAI). His research interests are structured text processing, web retrieval systems and full-text search technologies. He is a member of IPSJ, IEICE, JSIMS and DBSJ.