SEWISE : An Ontology-based Web Information Search Engine

Georges Gardarin[1], Huaizhong KOU[1], Karine Zetourni[1], Xiaofeng Meng[2], Haiyan Wang[2]

[1]PRiSM Laboratory, University of Versailles Saint-Quentin
45 Etats-Unis Road, 78035 Versailles, France
{Firstname.Lastname}@prism.uvsq.fr
[2] Departement of computer science,
Remin University of China, Beijing
{xfmeng@mail.ruc.edu.cn, hywang_@sohu.com}

Abstract: Since the begin of the 90's, the World Wide Web (WWW) rapidly guides the world into a newly amazing electronic village, where everybody can publish everything in electronic form and find almost all required information. The volume of available information is increasing exponentially in different formats, 80% being text. It remains hard to find interesting information directly from Web sources. SEWISE is an ontology-based Web information system to support Web information description and retrieval. According to domain ontology, SEWISE can map text information from various Web sources into one uniform XML structure and make hidden semantic in text accessible to program. The textual information of interest is automatically extracted by Web Wrappers from various Web sources and then text mining techniques such as categorization and summarization are used to process retrieved text information. Finally, text descriptions are built in XML format that can be directly queried. SEWISE provides support for topic-centric Web information search. The SEWISE prototype is implemented and has been experimented using French financial Web news from several popular sites.

1. Introduction

To overcome the bottleneck of information retrieval on the Web, there are urgent needs to automate program access to semantic abstractions of information. The main barrier to information automatic access is that all existing information is represented freely by different information providers and that concepts in the same domain are very often expressed using different methods. The consequence is that the semantic of information is not understandable for search engines and that knowledge cannot be shared between data sources.

To solve the above problems, many technologies have been proposed. XML is a first syntactic step for information communication or exchange between programs. It allows application developers to describe different information elements by using domain-specific terms. Domain-specific metadata [Jeffery98] is another important techniques. The metadata can describe the semantics of the underlying information by metadata schemas. A metadata schema is comprised of both a vocabulary used for the description

of information in a domain and a set of semantic relationships to structure this information [Amann+99]. Metadata schemas can support a sharable, structural view of information with rich semantics to be communicated. The Resource Description Framework (RDF) [RDF] is a foundation for representing metadata ; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF emphasizes facilities to enable automated processing of Web resources. It supports standard mechanisms for the representation of metadata schemas as well as source descriptions. Both RDF descriptions and schemas are expressed by XML documents. Ontology has been used in several disciplines, from philosophy to knowledge engineering. It is independent of the actual data, reflects a common understanding of the semantics of discourse, and is used to share and exchange semantic information between sources [Gruber93]. They are declarative specifications of the basic concepts and roles in an application domain.

SEWISE is an ontology-based Web information system to support Web information organization and retrieval. It uses ontologies only for describing a common data model in a given application domain. According to domain ontology, SEWISE organizes the text information from various Web sources into one uniform structure and make hidden-semantic of text accessible to program. So, by ontology-based, we mean that SEWISE provides to applications a common concept model of each domain in contrast to different information at various Web source sites. In SEWISE, based on domain-specific ontologies, the Web wrapper are inductively generated in semi-automatic way. These Web wrappers automatically extract interesting text information from various Web sources. Then text mining techniques such as categorization and summarization are used to process retrieved text information. Finally, text enriched images are built in XML format. The SEWISE prototype has been experimented using French finance Web news from several popular sites.

This rest of this paper is organized as follows. In section 2, we describe the system architecture. In section 3, we present our Web wrapper generator toolkit. Section 4 reports on the text mining techniques that are integrated in our system. Section 5 focuses on the financial application that we have implemented based on our system. In conclusion, we summarize the contributions of this paper and introduce future research directions.

2. System Architecture Overview

This section gives an overview of the SEWISE architecture and describes the functions of its various components. The global architecture is represented in Figure 1. It is composed of three layers of software : the Web information extraction layer, the Web information enrichment, and the XML Repository containing the XML document base.
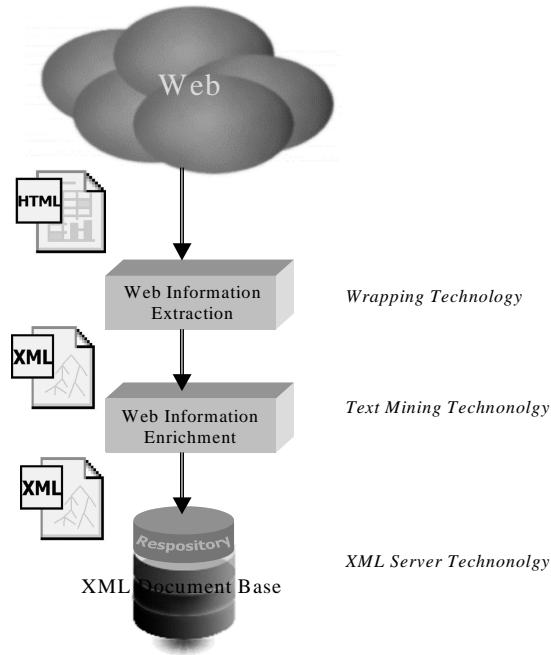
Figure 1 : Global Architecture of SEWISE

2.1 Web Information Extraction

The Web Information Extraction layer aims at extracting domain-specific information elements from various Web sites relevant to the application topics. It is composed of Web wrappers, which are specialized programs to automatically extract data from Web pages and covert the results into structured XML documents. For example, in the case of financial news, interesting information elements include the title, date, source, news content, etc. A Web wrapper takes advantage of the tag structure contained in the HTML pages to locate and extract interesting information fragments. Given a Web site, a Web wrapper retrieves corresponding HTML pages and extract interesting information elements contained in the HTML pages. Finally it maps extracted data into a common XML model accessible by applications. Web Wrappers used in SEWISE are semi-automatically generated by the SGWRAP toolkit that will be detailed in section 3.

2.2 Web Information Enrichment

The second layer semantically enriches XML documents produced by wrappers using text mining techniques. In SEWISE, applied text mining methods include indexing, categorization, summarization, and keyword extraction. Indexing is exploited to represent extracted texts by using domain-specific dictionary or vocabulary. SEWISE

uses the document vector space model [Salton83] to represent texts. Categorization helps to find the topics each text talks about while summarization gives support to find representative sentences in texts based on sentence weighting algorithms. All found sentences together are considered as the summary of the text. Keyword extraction can statistically extract specified number of keywords. Mixing these techniques, the Web enrichment layer enhances the extracted semantics of Web pages. The information produced by this layer are stored in the common domain-specific ontology model. Section 4 will furthermore present these components.

2.3 XML Repository and Document Base

The Web Document Base layer organizes XML documents obtained by the second layer in relational databases. It is composed of a relational DBMS on top of which is installed an XML Repository implementing XML to relational mappings. Both the e-XML Repository component of e-XMLMedia and the Oracle Text cartridge are integrated in SEWISE. The former is able to store XML documents in any object-relational DBMS and provides XQuery access to retrieve XML document fragments and assemble them [Gardarin+02a]. Oracle9i can directly store XML document in text tables and provide retrieval operations on different sections of XML documents.

3. A Schema-Guided Toolkit for Generating Wrappers

A wrapper is a software component that "*logically converts the underlying data objects to a common information mode"* [Chawathe+94]. In the Web environment, the purpose of wrappers should be to convert information implicitly stored as HTML document into information explicitly stored as a data-structure for further processing. To access and encode in XML information of interest presented in HTML pages, some Web wrappers around different Web sites are integrated in SEWISE applications. This section introduces the functions of Web wrappers and describes SGWRAP, an intelligent toolkit for generating XML-based wrapper.

3.1 Wrapping Web Sites in XML

The function of Web wrappers is to extract useful information elements presented in various Web sources and to map them into a common data model accessible to search engines and mediation tools. Through Web wrapper, application program can visit source Web information without understanding them. To this end, three main tasks must be executed: (i) Retrieving Web pages from source sites. (ii) Locating and extracting useful information elements. (iii) Mapping the extracted data into a common data model.

Retrieving means to fetch and clean Web pages up to reach a syntactic XML format. It can be carried out by using some existing Java tool such as TIDY [TIDY]. After retrieving, we can obtain a DOM [DOM] tree of the Web information.

Extracting is one the most difficult part of a wrapper. It relies on a set of predefined extraction rules that indicate the position of useful information elements and describe the relationships between them in the DOM tree produced by the retrieval process. With the help of the predefined extraction rules, domain-specific information elements can be identified from other information in the DOM tree and extracted. The logical information relationships described by the extraction rules are build and kept.

Mapping converts extracted data and their relationship into a common data model. The common data model constitutes a domain-specific ontology that defines a set of vocabulary for representing domain entities and the relationships between them.

3.2 The Schema-Guided Wrapper Generator

The toolkit SGWRAP was developed to semi-automatically generate Web wrapper particularly for extracting locally well-formatted data from Web HTML pages [Meng+02]. For example, both book sale data (http://www.amazon.com) and stock market data (http://biz.yahoo.com) are locally well-formatted. In order to automatically generate the wrappers capable of extracting large information elements such as news story documents from Web pages, we expanded SGWRAP. In contrast to extracting small locally well-formatted data, large text information elements in Web pages pose several new problems. The system architecture of SGWRAP is shown in Figure 2.
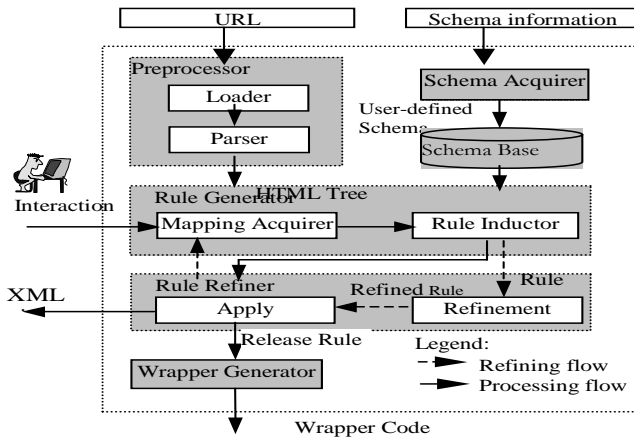

Figure 2 : Architecture of the SGWRAP Toolkit

Let us first outline the way SGWRAP works. It receives two input parameters: an URL and a predefined data schema (more precisely a DTD). The URL indicates the example Web page to wrap and the predefined data schema defines domain-specific entities and their logical relationships. Then SGWRAP communicates with the user who guides SGWRAP to obtain a set of mapping pairs between information elements of interest in the example Web page and the domain-specific entities defined by the data schema. Next SGWRAP inductively learns extraction rules from this set of mapping pairs. Finally, it

automatically generates the wrapper code in which the learned extraction rules have been coded.

We now describe the main components of SGWRAP corresponding to the grey boxes in Figure 2.

Preprocessor: It is responsible for setting up the environment for the system. Using the URL given by the user as input to SGWRAP, it fetches the corresponding Web page and then represents the fetched HTML document as a DOM tree, called the HTML DOM tree. For this, since there often exist syntax errors such as missing tags and tag mismatching in the original HTML document, the fetched HTML document must be carefully preprocessed and all detected errors must be fixed. We use TIDY [TIDY] to generate correct XHTML. In the resulting DOM tree, a data item is a leaf node and its position in the document can be described by the path from the root to the leaf. The processed XHTML document is shown to the user on the screen.

Schema Acquirer: Another input to SGWRAP is a user-defined data schema under the form of a DTD (type information could be added in a future version). It defines domain-specific entities and the logical relationships between them. Schema Acquirer reads the DTD into the system and displays it on the screen as a tree, called the data schema tree.

Mapping Acquirer: It aims to produce a set of mapping pairs between the information elements of interest in the example Web page and the entities defined by the user-predefined data schema. The user has to guide the system by clicking in both the schema tree and the XHTML page. First, the user clicks the schema tree to select one domain entity such as "title" or "author". Mapping Acquirer keeps the selected entity name and its path in the schema tree. Next the user has to highlight the information element of interest contained in the example XHTML page, which corresponds logically to the selected entity. Its path in the HTML DOM tree can be identified and recorded by Mapping Acquirer. Finally, by clicking one command button the user tells Mapping Acquirer to build the set of mapping pairs of the following form:

(HTML DOM tree path, schema tree path)

where "HTML DOM tree path" is the HTML DOM tree path of the highlighted information element and "schema tree path" is the schema tree path of the selected entity. One example of such mapping pair is as follows:

(/body/table[0]/tr[2]/td[1]/table[0]/tr[1]/td[0]/table[0]/tr[1]/td[0]/table[0]/tr[0]/td[0] /small[0]/a[0]/text()[0], financenews.category.name).

Rule Generator: It runs an inductive algorithm to generate extraction rules from the set of mapping pairs. The extracted rules are expressed in XQuery as queries of the form LET … RETURN…. For example, the following rule extracts the category name:

LET $name=

/body/table[0]/tr[2]/td[1]/table[0]/tr[1]/td[0]/table[0]/tr[1]/td[0]
/table[0]/tr[0]/td[0]/small[0]/a[0]/text()[0]

RETURN <name>$name</name>

Wrapper Generator: The algorithm recursively generates all rules for each of the element in the schema, exploring the tree in depth first. It computes one rule for each node and correctly nests all rules. In general, all objects (related to extraction rule nodes) and the relationships between them are associated to rules. The final rule is a data source specific program that can be used to extract information from other pages with similar structure. When one runs the generated wrapper, all objects and their relationships are constructed in memory.

4. Extending Wrappers with Text Mining Techniques

Text mining techniques can help unearth the knowledge hidden in various textual document. This section presents the text mining techniques that we integrated within **SEWISE**. They were first implemented in a document categorization system [Gardarin+02b], which was latter merged with the SGWRAP toolkit.

4.1 Text Indexing (TI)

The objective of text indexing is to identify significant terms[1] contained in a document and represent textual documents in a simple model for facilitating analysis and processing. The significant terms must characterize the topics of documents. They must also discriminate a given document from others. For this, a domain-specific term dictionary and an approach to estimate word importance have been developed.

Our approach starts with a domain-specific corpus of pre-labeled documents. All distinct words in the corpus are first identified and all stop words are sorted out. The Porter stemming algorithm [Porter80] is used to convert words to their stems. By using CHI Square statistic method-χ^2 [Yang+97], we calculate for every term the degree at which it characterizes various domain topics. We select some terms with high degree to create a domain-specific dictionary.

Given a document, we identify all dictionary terms contained in the document text and then use the *tf-idf* [Salton83] formula to calculate the importance of each term relatively to the document content. Finally, we use the document vector space representation model [Salton83] to map the document into a vector in the high dimension vector space of term concepts.

4.2 Text Categorization (TC)

Text categorization is the procedure that aims to discover the domain topics contained in a document and assign the document to the discovered topics. It is often considered as a problem of inductive supervised machine learning.

---

[1] Generally speaking, terms can be phrases, n-grams, words and word stems. In SEWISE, terms are word stems.

We represent a document by using the document vector space model, which takes the document as a bag of words, and implement the k-nearest neighbor (k-NN) classifier [Yang+99] to categorize the document. k-NN applied to text categorization is one of the top-performing machine learning algorithms and is simple to implement. It relies on the basic assumption that the classification of an instance is most similar to the classification of its nearby instances. A new document is categorized in the following steps:

1. Indexing – the system builds a document vector to represent the document by identifying the significant terms present in the document and estimating their weights.
2. Retrieving k nearest neighbors – the system calculates the similarities between the document vector of the new document and all document vectors of training documents. The cosine-based similarity distance [Salton83] is used for similarity calculation. Then all training documents are ranked in decreasing order of similarity, among which we select the k top documents for estimating category score at the next step.
3. Calculating category score – for every category, we calculate a category score by combining the membership weights of the selected k documents to the category. A category score indicates the relevance of the new document to the category. The independence-based weighting (IBW) [Kou+02] algorithm is used to calculate the category score.
4. Assigning category – the categories with the category score higher than a predefined threshold are assigned to the new document. The predefined threshold is determined during the learning phase for each category.

k-NN classifier is a lazy learning algorithm. In the learning phase, the k-NN categorization system only does two things: build document vectors for all training documents and determine the score threshold for every category.

How to determine the score threshold? For a pair of a category and a training example document, the score can be calculated through the first three steps above. Then given a category, all training documents can be ranked in decreasing order of the calculated score between the category and the training documents. The F1-measure [Yang99] value can be calculated for this category by going down the ranked list of training documents. We set the score as the score threshold that maximizes the F1-measure.

4.3 Summarization by SENtence EXtraction (SENEX)

Document summarization can help the users identify which documents are most relevant to their needs. But summarization is a hard problem of natural language processing. In fact, the human summarization is the process that given a document one tries to understand, interpret, abstract it and finally generate a new document as its summary. The approaches explored includes linguistic approaches [Mittal+98] [Radev+98], statistical and information-centric approaches [Strazalkowski+98] [Goldstein+98], etc. Instead of understanding and abstracting the original document, automatic text-span extraction methods are often practiced to produce document summary. By text-span extraction, we mean that some significant sentences with fixed lengths are extracted

from original document to build a summary. In SEWISE, we generate document summary by sentence extraction.

We plan to implement the approach introduced in [Goldstein+98]. But differently from [Goldstein+98] where both statistic features and linguistic features have been used, we only use statistic features. Our approach is outlined as follows:

- Generate the document vector Q (called query vector) for a given document d.

- Identify all sentences S={S_i} of the document d.

- Consider each sentence S_i as a short text document, and generate a document vector, noted as S_i.

- Calculate the score for each sentence using the formula:

$$Score(S_i) = \sum w_t Q_t S_{it}$$

  where w_t is the weight of dictionary term t, and Q_t and S_{it} are the weights of term t in the query vector Q and sentence document vector S_i respectively.

- Rank the sentences in the decreasing order of calculated scores.

- Combine in sequence order some top sentences as document summary.

4.4 Keyword EXtraction (KEX)

Keywords of textual documents can support efficient information retrieval and are widely used in search engines. In our current implementation, we only use individual words as keywords; compound words are not considered yet. To extract keywords from a document, both the relevance of terms to the domain topics and the importance in document are considered, then scores are calculated for every term. Finally some terms with highest score are selected as keywords of the document.

We describe how to calculate the domain relevance and the document importance of a term. Given a document d belonging to the category c, we assume the centroid vector $\vec{c}$ of the category c and the document vector $\vec{d}$ of d are :

$$\vec{c} = \left(w_1^c, w_2^c, ..., w_T^c \right) \text{ and } \vec{d} = \left(w_1, w_2, ..., w_T \right)$$

$\vec{c}$ is defined as the average of document vectors of training examples in the category c and can represent the category schema; T is the total number of terms in the domain-specific dictionary. We extract $w_i^c / |\vec{c}|$ as the domain relevance of the ith term t_i to the category and w_i as the importance of t_i in d. Then we define the score of t_i by the formula:

$$score(t_i) = \lambda \times w_i + (1 - \lambda) \times \frac{w_i^c}{\left|\vec{c}\right|} \quad if \; t_i \in d; \, 0 \; if \; t_i \notin d$$

where $\lambda (0<\lambda<1)$ measures the contribution of the domain relevance and the document importance of a term. Note that all terms not present in d have a score value of 0.

4.5 Wrapping up all together

We have introduced four complementary techniques for extracting semantics from texts : text indexing, text categorization, sentence extraction, and keywords extraction. As explained above, SEWISE includes wrappers which extract information from Web pages and encode them in XML following a given DTD. The wrapper generates XML documents including textual elements. We apply the four text mining techniques (TI, TC, SENEX, KEX) to text elements in order to semantically tag the XML document generated by wrappers. Thus, we obtain XML images of Web pages with semantic information and extracted information. We further add the URL, the generation and last modification dates of the page, which are useful information for queries and later for refreshment of images.

5. Experimentation with SEWISE

In this section, we introduce the application we have experimented with, including the system ontology, the source Web sites, the generated Web Wrappers, and the generated XML image files enhanced by text mining techniques. The French financial news has been chosen as domain of interest. We build the collection of example Web documents by collecting the news stories at Yahoo!Finance[2] through Web wrapper, which are used to inductively learn document categories and to construct a financial knowledge database. The financial news from some popular French Web sites (such as Boursorama[3], FirstInvest[4] and Lesecho[5]) have been integrated in the prototype. According to the system ontology, a semantic enriched XML image file is built for every Web document .

5.1 Web Financial News and Web Wrappers

We use some popular French finance news sites to experiment with SEWISE. The financial news from Yahoo!Finance are collected to induce the domain knowledge

---

[2] http://fr.biz.yahoo.com/fc/industrie.html
[3] http://www.boursorama.com/international/actu_intern_index.phtml
  http://www.boursorama.com/infos/actualites/actu_marches.phtml
[4] http://www.firstinvest.com/actualite/allthenews.asp
[5] http://www.lesechos.fr/infodirect/touslestitres.htm

database. Yahoo!Finance has organized financial news into 16 different categories with the category hierarchy shown in Figure 3.
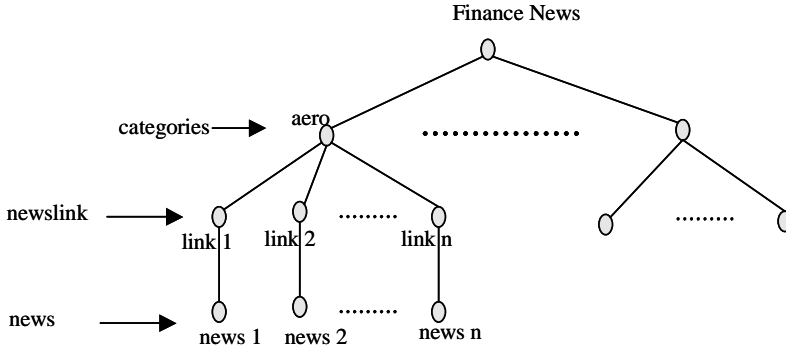

Figure 3: The hierarchy structure of Yahoo!Finance news

The finance news from the other accessed sites are not organized into any category, except FirstInvest. However, the only 5 categories of FirstInvest are very different from those of Yahoo!Finance. Thus, we do not retain FirstInvest categories. The news on all sites have title, date, source and content. But the date formats are different. They will be integrated into a common domain ontology structure defined in section 5.2.

A wrapper around each Web site is generated by SGWRAP. For this, a DTD file related to the Web site must be defined to describe information elements present in the domain data ontology and the relationships between them. When defining the DTD file, both logical relationships and Web link relationships between the information elements must be considered. For example, we define the DTD file for Yahoo!Finance given in Figure 4. Although the element "newsbody" does not belong to the domain ontology, it must be introduced to represent the Web page link relationship between categories and news as in Figure 3. Its value is the URL of some finance news.

```
<!ELEMENT financenews (category+)>
<!ELEMENT category (name, news+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT news (title, source, date, newsbody)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT source (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT newsbody (content)>
<!ELEMENT content (#PCDATA)>
```

Figure 4: DTD file for Yahoo!Finance news

5.2 An Ontology for the Finance News Domain

An ontology consists of an "explicit specification of a conceptualization" [Gruber93]. It can be used in an integration task to describe the semantics of the information sources and to make the content explicit. With respect to the integration of data sources, ontologies can be applied for the identification and association of semantically corresponding information concepts [Wache+01]. We use an ontology only for explicating content, that is, making domain semantics explicit and understandable for applications. A single ontology approach is adopted, which exploits one global ontology providing a shared vocabulary for the specification of the semantics [Wache+01].

We use RDF and Dublin Core metadata language to define our ontology structure by an XML schema file . Seven standard Dublin Core elements are adopted. We enrich it with some rdf elements, the keywords and vector elements. Altogether, the ontology structure is as follows:

- rdf:about – Web document URL.
- dc:title – title of the Web document.
- dc:source – original source of the news. For example, "reuter","lesecho".
- dc:date – date when the news is published by dc:source. Its format is "YYYY-MM-DD".
- dc:subject – domain topics that are contained in the news, discovered by DocCat and delimited by ",".
- dc:description – summary of Web document, generated by SENEX.
- dc:relation – out-links to other Web pages contained in the Web document indicated by rdf:about.
- dc:content – content of interest information extracted from Web HTML document by the Web wrapper.
- keywords – some keywords of the news, extracted by KEX.
- vector—vector representation of the Web document indicated by rdf:about.

5.3 Application Overview

The application proceeds as follows. First, wrappers are generated by SGWRAP for Yahoo!Finance, Boursorama, FirstInvest and LesEchos. The Yahoo wrapper automatically extracts financial news from the Yahoo!Finance site every day; the category hierarchy relationship is also extracted. News classified by categories are stored in Oracle text tables to make up the domain knowledge base.

The three other wrappers work around the corresponding Web sites to extract the XML images of the news. We then apply to them the Indexing, SENEX, DocCat, and KEX text mining algorithm. The outputs are semantically enriched XML documents that are stored in an XML repository. The XML repository can then be queried using rich XQueries to locate relevant data for applications.

5.4 Benefits for Web Applications

SEWISE provides direct support for topic-centric Web information search. Traditional Web search heavily relies on matching terms in user's text query with the terms present in documents. Such search is topic-independent. Users enter several keywords that describe what they search for, then a list of documents is returned ordered by relevance, as with GOOGLE. Often the list is very long and irrelevant documents are retrieved.

In fact, when the users enter keywords for searching for information, they have some relevant topics in mind and their needs are topic-dependent. For example, "I want the news about France finance". This query is related to "finance" topic rather than "sport" topic. According to keyword-based search, one may enter "France", "news", "finance". In this case, term matching is executed against all news and some news about France finance can not be found if they do not contain the "finance" word. By organizing Web news into different topics and compiling them into XML databases, SEWISE can support topic-specific structured query. For example, the news corresponding to the above example can be obtained by using an XQuery where clause such as « contains(content, "France") and topics="Finance" ». The search is executed only on the news having finance as topic.

6. Conclusion

As the Web is exponentially expanding, it is very hard to find interesting information directly from Web sources using keyword-based search engines. One approach explored in certain research prototypes or commercial systems is to extract domain-specific information from relevant Web sites, semantically integrate them according to a common domain-specific ontology, and provide structured or semi-structured access. This is the approach experimented in SEWISE. The system integrates wrapper and text mining technology. An application can be perceived as a domain ontology-based semantic Web information search engine.

We have experimented in the financial domain, where a common ontology structure has been defined for Web finance news. Web finance news are extracted by the wrappers from HTML Web pages. Then statistic text mining techniques are used to semantically enrich extracted finance news with XML descriptors. The images are enriched by titles, categories, keywords, summaries, etc. Finally, we compile Web finance news from different Web sites into an XML database. A topic-centric information search application can be build on top of SEWISE.

SEWISE couples a wrapper generator with text mining techniques to generate the ontology-based XML images of Web pages from sources of similar domains. The resulting XML images can be stored in an XML database and queried in XQuery as proposed above. Another interesting approach would be to query directly Web page XML images using XQuery. A mediator component [Gardarin+02a] could then be used to perform on the fly data integration and query processing. This would lead to a powerful search engine for the semantic Web.

References

[Amann+99] B. Amann and I. Fundulaki: Integrating Ontologies and Thesauri to Build RDF Schemas, ECDL-99, Lecture Notes in Computer Science, Paris, September 1999; pp.234-253.

[Chawathe+94] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom: The TSIMMIS Project: Integration of Heterogeneous Information Sources. In Proc. of IPSJ Conference, 1994; pp.7-18.

[DOM] W3C: Document Object Model, http://www.w3.org/DOM/

[Gardarin+02a] G. Gardarin, A. Mensch and A. Tomasic: An Introduction to the e-XML Data Integration Suite, in the proceeding of EDBT,2002; pp.297-306,

[Gardain+02b] G.Gardarin, H. Kou, A.A.d'HEYGERE and K. Zeitouni: Document Categorization in an XML Database, Technique report at PRiSM laboratory, no. *Rapport # 2002/15*, 2002.

[Goldstein+99] J. Goldstein, M. Kantrowitz, V. O. Mittal, and J. Carbonell: Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In Proceedings of SIGIR-99.

[Gruber93] T.R. Gruber: A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition,5(2), 1993; pp.199-220.

[Jeffery98] K. Jeffery: Metadata: An Overview and some Issues. ERCIM NEWS, (35), October, 1998. http://citeseer.nj.nec.com/jeffery98metadata.html

[Kou+02] H. Kou and G. Gardarin: Study of Category Score Algorithms for k-NN Classifier, in the proceeding of 25th SIGIR,2002; pp.393-394.

[Mittal+98] Mittal, V.O., kantraowitz,M., Goldstein, J., and Carbonell, J.: Selecting Text Spans for Dcument Summaries: Heuristics and Metrics. In Proceedings of AAAI-99, Orlando, FL, July 1999; pp.467-473.

[Meng+02] X. Meng, H. Lu, M. Gu, H. Wang: SG-WRAP: A Schema-Guided Wrapper Generator, in the proceeding of the 18th ICDE, 2002; pp. 331-332.

[Porter80] Porter: An algorithm for suffix stripping, Program, Vol. 14, no. 3,1980; pp. 130-137.

[Radev+98] Radev, D., and McKeown, K.: Generating natural language summaries from multiple online sources. Coputational linguistics 24, 3, September 1998; pp. 469-501.

[RDF] http://www.w3.org/RDF/

[Salton83] G. Salton and McGill: Introduction to Modern Information retrieval, 1983, McGraw-Hill Book Company.

[Strazalkowski+98] Strazalkowski, T., Wang, J., and Wise, B.: A robust practical text summarization system. In AAAI Intelligent Text Summarization Workshop,Stanford, CA, Mar, 1998; pp.26-30.

[TIDY] http://www.w3.org/People/ Raggett/tidy.

[Wache+01] H. Wache, T. Vogele, U. Visser, H. Stuchenschmidt, G. Schuster, H. Neumann and S. Hubner: Ontology-Based Integration of Information- A survey of Existing Approaches, In Stuckenschmidt, H., ed., IJCAI-01 Workshop: Ontologies and Information Sharing, 2001; pp.108-117.

[Yang+97] Y. Yang and Jan O. Pederson: A Comparative Study on Feature Selection in Text Categorization, In the 14th Int. Conf. On Machine Learning,1997; pp.412-420.

[Yang+99] Y. Yang. and Xin Liu: A re-examination of text categorization methods, In the Proceedings of 22th SIGIR,1999; pp.42-49.