International Journal of Fuzzy Logic and Intelligent Systems, vol. 3, no. 1, June 2003 pp. 127-132

Wrapper Generation for Collecting Comparative Shopping Information

Ju-Ri Shin, Bong-Ki Sohn, Keon Myung Lee[1)]

School of Electric and Computer Engineering, Chungbuk National University, and
Advanced Information Technology Research Center(AITrc), Korea
kmlee@cbucc.chungbuk.ac.kr

Abstract

This paper proposes a wrapper generation method for collecting comparative shopping information from various Internet shopping malls. The proposed method is a kind of supervised learning method to learn wrappers from sample web pages along with information locations designated by the administrators. It generates wrappers expressed in the form of generalized tags sequences and frame filling procedures for semi-structured web pages. The paper also presents how to use the learned wrappers and describes a prototype system which implemented the proposed ideas and methods.

Key words : wrapper, machine learning, comparative shopping service

1. Introduction

With the wide accessibility to the Internet, Internet shopping malls become one of most popular services. Although Internet shopping malls give customers many choices, customers should spend lots of time to surf various web sites to find the best place to buy. They should browse many shopping malls to compare the prices and other factors for the interesting items. To help avoid this tedious and time-consuming surfing, comparative shopping services have been developed and they are now in business.[8,9] Comparative shopping service systems collect product shopping information from Internet shopping malls, classify it into proper categories, and build their own comparative shopping information databases according to the categories. Then they receive users' queries about products and provide some comparative shopping information for the customers by retrieving their databases.

In comparative shopping services, the hardest part is to extract useful product shopping information from shopping mall web sites which have their own styles to present product information and which frequently change their page designs and layouts to attract customers' attention or to improve convenience in use. To build shopping information databases, most commercial comparative shopping service systems have some manually coded programs tailored to specific web sites or directly get the database information from Internet shopping malls by tying contracts. If such information is not provided by shopping malls in a coordinated format, the information gathering programs tailored to such web sites should be usually re-developed each time corresponding shopping malls change their design or ways to access for their information. It is really costly and time-consuming to do this according to the modification of shopping malls. Direct access to shopping malls' databases is the easiest way. However, many major shopping malls hesitate to allow their prices to be compared and tend not to open their databases outside. As a method to efficiently cope with these problems, learning-based wrapper techniques have been actively studied.[1,2,3,5,6]

Most shopping malls have been designed in semi-structured formats because they usually display multiple shopping items according to the user queries or browsing to their product databases. Thanks to this characteristic, most information related to the comparative shopping is expressed in table formats. From this observation, we have developed an instance-based learning technique to generate a wrapper which extracts comparative shopping information from Internet shopping mall pages. The developed method is a kind of supervised learning in which user needs to tell locations of relevant information to the wrapper learner. The wrapper learner analyzes the given example pages and produces the corresponding parse tree in left-child-right-sibling tree structure. It replaces all tags and texts with corresponding (sometimes, generalized) tags with consideration of the provided locations for the relevant information. In the proposed method, generated wrappers are expressed as a sequence of generalized tags. These sequences are used to extract comparative shopping information.

To see how effective the proposed wrapper learning method is, we have implemented the proposed wrapper learning method and a prototype of the information-gathering engine which collects from Korean shopping malls comparative shopping information using the learned wrappers. The paper presents how to implement them.

2. Wrapper Generation for Semi-Structured Shopping Mall Pages

Internet shopping malls usually sell lots of products and thus they maintain product information in databases. According to users' browsing, the shopping mall servers

dynamically generate pages containing related items. Due to this property of shopping malls, many pages of Internet shopping malls are formatted in tables since tables enable to flexibly design page layouts. Even though table cells may contain free texts in various forms, most shopping mall pages can be viewed as semi-structured documents. Therefore, we might apply more simplified wrapper learning method to this kind of semi-structured documents. We have developed a wrapper learning method applicable to Internet shopping mall pages which are semi-structured.

2.1 Wrapper Generation

In comparative shopping services, it is expected to provide the information about product name, price, manufacturer, URL of the shopping mall to sell the product, and URL of the page explaining the product. Most shopping mall pages provide these five pieces of information for each product, and this information unit is grouped together in tables. Figure 1 shows the typical layouts of product information units in the table-based web pages. In the figure, the hatched cell(s) corresponds to an information unit which describes a product.

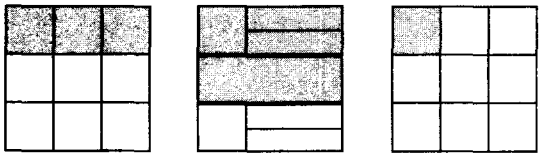

Fig. 1 Product Information Units

Figure 1-(a) represents a typical layout of web pages in which a row is used to present a product. Figure 1-(b) is the case in which multiple rows are used for a product. Figure 1-(c) uses a cell for a product, where tag information like
 and <p> and some punctuation marks (e.g., :, ~) as well as table layout information are required to discriminate the product information. One of key tasks in wrapper learning is to identify the product information units like Figure 1.

The proposed method for wrapper learning is a kind of supervised learning, in which a user labels which information to be extracted from the page given as a training example. In the method, the comparative shopping mall administrator chooses a Web page containing shopping items from which shopping information will be extracted, and informs the wrapper learner of the location of a table with shopping items and the locations of relevant fields (e.g., product name, price, etc.) for a shopping item to be extracted by picking these things through the graphical user interface. The learner analyzes the example HTML page, generates its corresponding parse tree, and then associates the designated locations with their corresponding nodes of the parse tree. For the selected information unit, it ignores irrelevant tags and items like images, and replaces the tags and texts with generalized tags and then gets a generalized pattern. To the unselected information units in the selected table, it applies the same procedure to get generalized patterns. By comparing the pattern of the selected information unit with the patterns of the other units, it determines a pattern to be used by the information-gathering engine to collect the comparative shopping information. The pattern generated in this way plays the role of a wrapper for the shopping mall from which the example page was chosen. The patterns are expressed as a sequence of tags. A shopping mall may employ several different page layouts to display products in a convenient and attractive way. Therefore we may need multiple wrappers for a shopping mall. The following shows the procedure used to learn a wrapper for an example page.

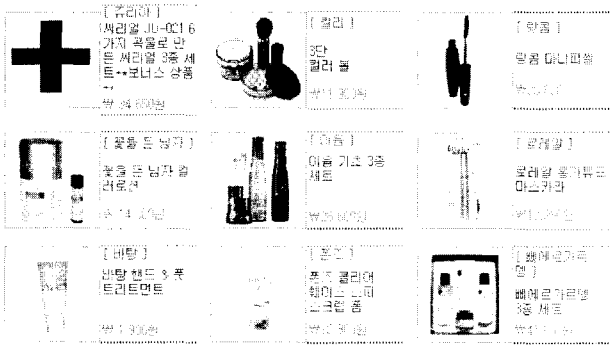

Fig. 2. An Example of Product Information at a Shopping Mall

procedure *Learning_Wrapper*
begin
- Locate through the graphical user interface the table containing shopping information in the example page and items for product name, price, and URL for a product.
- Parse the page and build a parse tree in left-child-right-sibling tree structure.
- Associate the selected items with their corresponding nodes in the parse tree.
- Replace the tags and texts with generalized tags for the nodes of the parse tree.
- Remove the irrelevant tags from the generated pattern.
- Find the deepest node DN that contains the designated items for product name, price, and URL in the parse tree.
- Check whether other siblings of DN contain the pattern of DN.
- If its adjacent siblings do not contain similar patterns to the pattern of DN, go up to its parent until the adjacent siblings contain similar patterns.
- Construct a pattern PP for the product unit in the current node and identify the node(s) covered by the pattern PP.
- Identify the nodes SP that have the same topological position with the pattern PP.
- Construct patterns for the nodes SP in the same way as in PP.
- Construct the wrapper pattern with the tags commonly appearing in both the pattern of DN and the other patterns of SP.

Validate the pattern by applying the procedure *Extract_Info_with_Wrapper* with the wrapper pattern to the current example page.
Report the constructed pattern as a wrapper for the corresponding page.
end.

Figure 2 shows a part of shopping mall page that contains product information. In the proposed method, the user is supposed to provide the locations of product names, sale prices, manufacturer, URL, and so on, for example pages in some way. Figure 3 shows the HTML code for the first product unit of the page of Figure 2. From the HTML code of Figure 3 and the label information, the proposed learner *Learning_Wrapper* generated the wrapper pattern shown in Figure 4.

```
<tr> <td width="90" rowspan="5">
<a href="prod_info.asp?code=C11hk03001"> <img
src="p_img/color.jpg" width="90" height="90" border="0"
class="imgbr"> </a></td><td height="1"
background="images/dotdot.gif"> </td></tr><tr> <td
width="95" class="txt"> <font color="#666666">[Kali]</font>
</td> </tr> <tr> <td width="95" class="a"> <a
href="prod_info.asp?code=C11hk03001" class="e3">Kalie<br>
3layer<br> colorball </a> </td></tr><tr> <td width="95"
class="txt"> <font color="#f3570a"> 11,600 won</font> </td>
</tr><tr> <td height="1" background="dotdot.gif"> </td> </tr>
```

Fig. 3 HTML code for a product of Figure 2

```
<TR> <TD> <A> <URL> <A> </TD> <TD> </TD></TR>
<TR> <TD> <FONT><MANUFACTURER> </FONT> </TD>
</TR> <TR> <TD> <PRODUCT> <A></A> </TD> </TD>
<TR> <TD> <FONT > <PRICE> </FONT> </TD> </TR>
```

Fig. 4 Learned Wrapper Pattern

2.2 Application of the Wrappers

The learned wrapper pattern is used to extract comparative shopping information from shopping mall pages downloaded by the Web crawling engine. As in the wrapper learning procedure, we first parse a downloaded page and construct a parse tree in left-child-right-sibling structure. Then we analyze the structure of the parse tree and identify the repeating patterns similar to the wrapper pattern. From the parts corresponding to the repeating patterns, we extract the information about product name, price, manufacturer, and related URLs. The next procedure shows how to extract comparative shopping information using the wrapper pattern from shopping mall pages.

procedure *Extract_Info_with_Wrapper*
begin
Parse the page downloaded from a shopping mall and construct its parse tree in left-child-right-sibling structure.
Build a generalized pattern that uses only the generalized tags appearing in the wrapper pattern for each node of the parse tree, and associate it with its corresponding node.
Follow down the tree until it reaches a node that contains the wrapper pattern but of which any child does not have the pattern.
At the level of the found node of the parse tree, extract from the page the information corresponding to product name, price, and related URLs.
If there remain other unprocessed nodes with the wrapper pattern in the parse tree, repeatedly apply the above steps to those nodes.
end

The information extraction is based on the tags' pattern in the proposed method. There is a possibility to extract some items even though they have nothing to do with shopping information due to their matching to the wrapper pattern by accident. In order to maintain quality information, additional processing is needed before putting extracted items into databases of comparative shopping service systems. One of possible ways to filter out this kind of irrelevant data is as follows: Assume that we have reasonable size of databases with comparative shopping information. To see if the extracted items from the current page are about shopping information, check how much percent of product names and manufacturers' name from the extracted items appear in

the databases. If the number is above the specified threshold, we consider them as shopping information. We still need to pay extra attention to inserting them into databases because products should be classified into proper categories and product names might be expressed in somewhat different way even though they indicate the same product.

3. Implementation of a Comparative Shopping Information Gathering System

To see the applicability of the proposed wrapper learning and application methods, we have implemented a prototype system to collect comparative shopping information from Korean Internet shopping malls. This section describes the architecture of the prototype system and its major components, and then presents how to determine the categories of extracted information.

3.1 System Architecture

The prototype system consists of four main components: wrapper generation subsystem, information gathering subsystem, category classification subsystem, data refinement subsystem. Figure 5 shows the schematic architecture of theprototype comparative shopping information service system.

Wrapper Learning Subsystem

The wrapper generation subsystem plays the role of learning a wrapper for a specific web page. It implemented

the proposed wrapper learning algorithm. This subsystem provides a graphical user interface which allows the system administrator to easily locate positions of the tables to contain shopping information and fields for product name, URLs, price, and manufacturer. The wrapper generation subsystem learns wrappers for the example pages and puts them into Wrapper Library which is the repository to keep wrappers learned by the wrapper generation subsystem.

Information Gathering Subsystem

The information gathering subsystem takes charge of collecting shopping information from shopping malls using wrappers. Comparative shopping information should be kept updating because shopping malls keep changing the product items and their prices. In order for the comparative shopping service system to maintain most recent information, it has to concurrently run many processes to collect comparative shopping information from shopping malls. In the implementation of the system, multi-agent paradigm was

introduced. In this paradigm, we create a wrapper for each shopping mall from which shopping information is collected, and let it activated according to specified schedule. For the purpose of integrating of information, so-called Information Integration agent was built to which all wrapper agents send collected shopping information. The Information Integration agent plays the role to store the collected information into Raw Product Database which stores shopping information collected by wrappers.

Category Classification Subsystem

In order to effectively provide comparative shopping information, the product information gathered from shopping malls should be classified into a proper category and maintained according to its category. In the prototype system, our system characterizes the category of a product into three levels: broad category, intermediate category, and detail category. For example, a specific printer from a company belongs to 'printer' (detail category), 'peripheral' (intermediate category), and 'computer stuff' (broad category). We developed a category classification method that uses an instance-based learning method to assign categories to products of a page. It assumes that some volume of product information database with assigned categories has already

been established. In order to assign a category to a page, it gets product information from the given page using wrappers and retrieves the categories for the products extracted from the established product information database. Based on the retrieved

category information, it finds the most frequent broad category, intermediate category, and detail category for the page. In addition, it evaluates the membership degrees of the page to specific categories with consideration of most frequent categories, the number of products without category information, the number of products with most frequent category and the number of products with categories different form most frequent category. If the evaluated membership degree is greater than the pre-specified threshold, it assigns the corresponding category to the page. Otherwise, it assigns only the broad category and/or the intermediate category, if available, to the page. Them. it gives the chance for system administrator to manually fill the unassigned category information later. Although this category classification method requires some manual works in the initial stage, it can efficiently classify products once some volume of product information is accumulated.

Data Refinement Subsystem

Internet shopping malls may represent the same shopping item in somewhat different ways for their convenience in decoration and manipulation. Therefore, the information gathered from different shopping malls should be refined to group together all the information about the same product. The data refinement subsystem takes charge of transforming the shopping information extracted by wrapper agents into refined formats suitable to comparative shopping information services. It creates standard names and manages them in databases. The databases are referred to update the raw information gathered by wrapper agents. They have some function of ontology for the terms used to describe product information. The subsystem also filters out illegitimately extracted data and logs them to be used in the validity test of wrappers. It constructs wrong category lists for the data with wrong category, and delivers them to the category classification subsystem where the information is used to adjust the category assignment to web pages. The data refinement subsystem uses some heuristic procedures to filter improper data out and performs the task to unify names for products and makers with reference to standard naming databases. However, these refinement tasks require frequent human intervention, so it is necessary to provide some convenient graphical user interface in the subsystem.

User Interface and Query Processor

The user interface module is the web interface to enable users to issue queries and get their results. The query processor module supports simple product queries and comparative product queries using the product information database.

3.2 System Implementation

A prototype system has been implemented on Windows 2000 platform with MFC library and ActiveX controls. Figure 6 shows a screenshot of the wrapper generation subsystem. In the figure, the user browses shopping malls with the web browser component located on the right upper window and labels the locations of information to

be extracted on the window. Then, the wrapper generation subsystem generates a wrapper pattern for the web page with the label information. The right lower window displays the information extracted by the learned wrapper as so to enable user to check its validation.

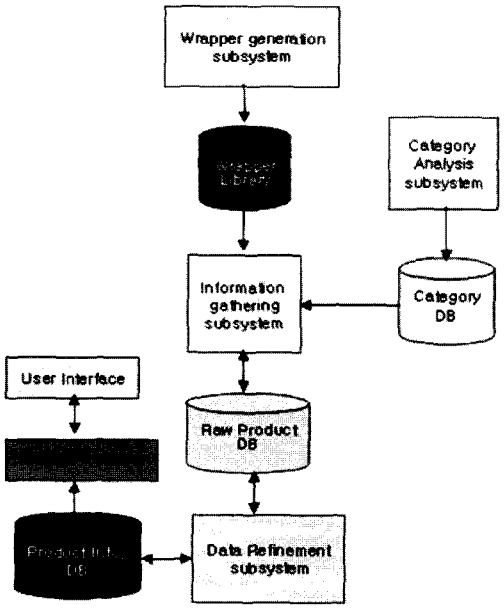

Fig. 5. The Architecture of the prototype System for Comparative Shopping Information Gathering and Services

4. Conclusions

This paper introduced a wrapper learning method for semi-structured shopping mall pages. The proposed method belongs to the family of supervised learning in which the system administrator picks the locations of table fields and labels them with their corresponding names. To generate a wrapper pattern, it parses the HTML pages to construct parse trees and generalizes nodes with generalized tags.

To see the applicability of the proposed learning method, we have implemented a prototype system to use the proposed wrapper learning and application methods. We have tested the system against several Korean shopping malls. From the experiments we have observed the proposed methods work very well when the shopping information is expressed in tables and each attribute of a product is expressed in different entries of tables. If a product information unit is expressed by a single entry of a table, its performance was poor because
 tags have the important role there but those tags usually are used carelessly. To this kind of pages which represent all product information in an entry of a table, it seems Cliff's frame-based extraction method[5] to be suitable to apply. It remains as one of our future studies to combine our wrapper learning method with the frame-based information extraction method which can be effectively used for semi-structure documents.

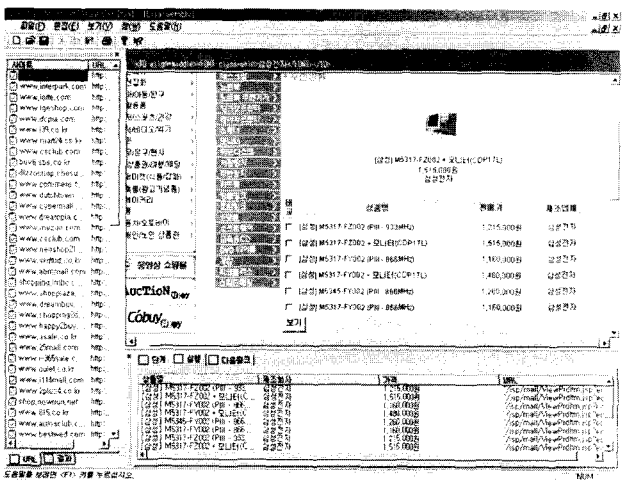

Fig. 6. A screenshot of the wrapper generation subsystem

References

[1] I. Muslea, S. Minton, C. Knoblock, A hierarchical approach to wrapper induction, *Proc. of the 3rd Annual Conf. on Autonomous Agents*, pp.190-197, Seattle, 1999.

[2] N. Kushmerick, D. S. Weld, and R. Doorenbos, Wrapper Induction for Information Extraction, *Proc. of International Joint Conference on Artificial Intelligence* (IJCAI), Nagoya, Japan, 1997.

[3] L. Xiaoying, AutoWrapper: Automatic wrapper generation for multiple online, *Proc. of Asia Pacific* Web Conference, Hong Kong, 1999.

[4] T. Mitchell, Machine Learning, (ch.2), McGraw-Hill Co., 1997.

[5] M. E. Cliff, R. Mooney, Relational learning of pattern-match rules for information extraction, *Proc. of the 6th National Conf. on Artificial Intelligence* (AAAI-99), pp.328-334, Orlando, 1999.

[6] N. Kushmerick, Regression testing for wrapper maintenance, *Proc. of the 6th National Conf. on Artificial Intelligence* (AAAI-99), Orlando, 1999.

[7] R.Doorenbos, O. Etzioni, D.weld. A scalable comparison-shopping agent for the World-Wide-Web. *Proc. of Automonous Agents*, 1997.

[8] BargainFinder, http://bf.cstar.ac.com/bf/.

[9] Jango, http://www.jango.com/.

[10] J. Cowie and W. Lehnert, Information Extraction, *Communication of the ACM*, Vol.39, No. 1, pp.80-101, 1996.

[11] G. Zacharia, A. Moukas, R. Guttman, P. Maes, An agent system for comparative shopping at the point of sale, *MIT Meida Lab.*, URL://ecommerce.media.mit.edu/, 1997.

[12] Nicholas Kushmerick, *Wrapper Construction for Information Extraction.* PhD thesis, Univ. of Washington, 1997.

[13] R. R. Korfhage, *Information Storage and Retrieval*, John Wiley & Sons, 1997.

Ju-Ri Shin
She received the B.S and M.S degrees in Department of Computer Science from Chungbuk National University, Cheongju Korea in 2000 and 2002. Her research interests are Agent System, Electronic-Commerce, Bioinformatics.

Bong-Ki Sohn
He received the B.S. degree in Department of Computer Sicence from Seowon University, Cheongju, Korea, in 1998. He received the M.S. degree in Department of Computer Science from Chungbuk National University, Cheongju, Korea, in 2000. He is currently working towards the Doctor degree in Department of Computer Science, Chungbuk National University, Cheongju, Korea. His research interests are in agent systems, machine learning and workflow system.

Keon Myung Lee
He received B.S., M.S., and Ph.D. in Dept. of Computer Science from Korea Advanced Institute of Science and Technology (KAIST) in 1990, 1992, and 1995, respectively. He joined Chungbuk National University, Korea in 1996 and is currently an associate professor at School of Electric and Computer Engineering. His research interests include data mining, agent systems, soft computing, bioinformatics, and computer security.