Data Extraction from Web Pages Based on Structural-Semantic Entropy

Xiaoqing Zheng
School of Computer Science
Fudan University
Shanghai, 201203, China
zhengxq@fudan.edu.cn

Yiling Gu
School of Computer Science
Fudan University
Shanghai, 201203, China
justice360@gmail.com

Yinsheng Li
School of Computer Science
Fudan University
Shanghai, 201203, China
liys@fudan.edu.cn

ABSTRACT
Most of today's web content is designed for human consumption, which makes it difficult for software tools to access them readily. Even web content that is automatically generated from back-end databases is usually presented without the original structural information. In this paper, we present an automated information extraction algorithm that can extract the relevant attribute-value pairs from product descriptions across different sites. A notion, called structural-semantic entropy, is used to locate the data of interest on web pages, which measures the density of occurrence of relevant information on the DOM tree representation of web pages. Our approach is less labor-intensive and insensitive to changes in web-page format. Experimental results on a large number of real-life web page collections are encouraging and confirm the feasibility of the approach, which has been successfully applied to detect false drug advertisements on the web due to its capacity in associating the attributes of records with their respective values.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – retrieval models; H.4.2 [Information Systems Applications]: Types of Systems – decision support (e.g., MIS).

General Terms

Keywords
Web information extraction, Structural-Semantic Entropy, False advertisement detection

1. INTRODUCTION
A huge amount of data is available on the World Wide Web and it continues to grow rapidly. It is obviously useful for a variety of purposes to automate the translation of web pages into structured data and make software tools gain database system functionality over these structured data. For instance, it enables easier comparison between products from different online stores and can support a variety of advanced applications such as product recommender, online false advertisement detecting, and demand forecasting systems. The main obstacle to providing better support to those applications is that, at present, the most of web content is not machine-accessible. What is worse, even web data that is automatically generated from back-end databases will lose the original structural information and makes it difficult for software tools to properly access or manipulate them as done in traditional databases.

Many previous systems for data extraction from web pages have been developed. A traditional approach is to write specialized programs, called "wrappers" or "extractors", to extract the contents of the web pages based on the knowledge of their formats. These wrappers were developed manually in early time. In other words, programmers have to observe the extraction rules in person and write wrappers for each web site. These processes require onerous manual coding and debugging. Since even small changes at the web site may prevent the wrappers from working properly, and the template or layout of web pages is often subject to change, maintaining those wrappers would be expensive and inefficient.

Due to the difficulty in writing and maintaining wrappers, some wrapper induction tools have been proposed to better address the issue of generating wrappers for web data extraction. Such tools take a set of HTML pages labeled with examples of the data to be extracted, and generate a wrapper by inferring a grammar for the HTML code – usually a regular grammar – and then use this grammar to parse the pages and extract the data of interest. The main limitation of existing wrapper induction techniques is the need for manually prepared training examples. A considerable amount of human effort is required to label a set of web pages for each site, and this task is, therefore, tedious and labor-intensive. This approach has another limitation that the target web sites must be known in advance, which is not possible in all cases. For instance, consider the case of "focused crawling" applications [1], which need to automatically crawl the web to search for topic-specific information.

Unsupervised IE systems do not rely on user-specified examples, and have no user interactions during the wrapper generation process. Some of them can only be applicable to the web pages that contain two or more data records, and learning wrappers can be solved by discovering repetitive patterns; others work by comparing the HTML structure of two (or more) given sample pages belonging to a same "page class", generating as a result a wrapper based on their similarities and differences. Although those proposals differ in the learning method used for wrapper generation, they all make some assumptions either about the repetitive patterns that already occur in given web pages, or about multiple sample pages belonging to the same template are provided in advance. Human effort is still required to pick those
web pages before the wrapper generation process. In addition, since they fail to associate the attributes of the schema with their respective values, due to lack of domain knowledge, the columns must be named manually after the data has been extracted.

This paper presents an information extraction algorithm that can locate and extract the data of interest from web pages across different sites. Our approach clearly departs from the ones in the previous literature in the following respects:

- Our algorithm are designed for the record-level extraction tasks that discover record boundaries, divide them into separate attributes, and associate these attributes with their respective values automatically. The algorithm does not rely on training examples, and does not require any interaction with the users during the extraction process.

- It works without the requirements that the web pages need to share the similar template or multiple records need to occur in a single web page. The algorithm can treat a single web page containing only one record. If a set of keywords used to describe the data of interest is collected, the extraction is fully automated, and it is easy to move from one application domain to another.

- A wrapper generated works for pages from many distinct web sources belonging to the same application domain. It also has the capacity of continuing to work properly when the format features of the source pages change, thus it is completely insensitive to changes in web-page format. The algorithm can be applied for two typical web information retrieval problems: web information extraction and topic distillation.

The paper is organized as follows. First, we give an overview of the research goals, and try to convey an intuition of the key ideas in Section 2. Section 3 presents a notion of structural-semantic entropy, which could be used to recognize the data of interest in web pages. Section 4 reports results of a number of experiments on real-life web sites and show the effectiveness of the proposed approach. The Web information extraction system for detecting false drug advertisement is introduced in Section 5. Section 6 presents a brief overview of related work. The conclusion and future work are summarized in Section 7.

2. OVERVIEW

The targets of this research are the data-intensive web pages that present products with their attributes and values, and are usually generated by scripts – i.e., programs – from back-end databases. HTML sites, however, are in some sense modern legacy systems, since such a large body of data cannot be easily accessed and manipulated [6]. The reason is that web data are formatted in diverse ways from site to site for human browsing. There are at least three irregularities in the presentation, which make the extraction tasks difficult.

- **HTML tags used to lay out the data regions of the web pages containing product descriptions.** The various layout formats can range from regular table, list to unstructured free texts.

- **Attributes used to describe products.** Although different sites may share some common attributes of the same product, they also convey some different product attributes. In addition, the attributes may occur in different orders from site to site, and some of them are optional, even in the same site.

- **Textual names used to interpret values of attributes.** The same attribute may have several different names depending on the sites used. They are synonymous words or phrases having the same or similar meaning.

A good wrapper should be capable of tackling the difficulties caused by the above three presentation irregularities, and extracting the data of interest across different sites. Before introducing our approach, it would be better to make clear the meaning of the data-intensive web pages in this paper. Sites usually contain those pages that mainly serve the purpose of offering access links to other pages. The anchors of these links may contain the contents that are repeated in the destination pages. We call those pages "link offer". For example, consider a drug store site; a possible link offer is the one that presents a list of the drugs that usually belong to one category. This page offers
links that lead to pages usually presenting a drug with more
detailed information, i.e., the attribute-value pairs of the drug.
From these observations, we do not need extract data from link
offers. They serve as listing of a given category, and do not carry
any useful information. The data they contain are redundant with
those data provided by other richer pages. The richer pages
containing more detailed information are called data-intensive
pages from which we intend to extract the data. Figure 2 shows a
typical link offer while Figure 1 shows a data-intensive web page
where the data-rich region is highlighted with dashed box.

Many sites, especially online stores, are most likely to provide as
much help information and convenience as possible to assist their
user with browsing. Consequently, the data-intensive pages
usually present their data along with some "decoration", such as
navigation bars and advertisement regions. As shown in Figure 1,
the web page shows information about a nasal spray that can be
used to generally alleviate cold or allergy symptoms. This page
contains several blocks of information. The data-rich region high-
lighted with dashed box provides the attribute-value pairs of the
nasal spray. In the leftmost block, there is a navigation bar that
contains links in order to navigate between the pages of the web
site. In the region below the dashed box, it provides a list of drugs
with the similar effects, which is like a small plug-in "link offer".
Although the "decoration" can enhance consumer appeal,
usability, and visual attractiveness, it brings great challenge to
locate the data-rich region, in order to differentiate the data of
interest from its "decoration".

2.1 Problem Formulation and Solution
In this paper we focus on the problem of detecting and extracting
the attribute-value pairs of products from the data-rich regions in
the data-intensive web pages. The problem studied can be
formulated as follows: "given a HTML page, identify if it is a
data-intensive page or not; if it is data-intensive, then locate the
data-rich region of the page and extract the attribute-value pairs
from the region." Notice that if we can locate a data-rich region in
a page, the page must be data-intensive. In other words, if it is a
data-intensive page, a data-rich region should be detected in the
page. Therefore, the most important thing is how to locate the
data-rich region in a given page.

As it can be seen in Figure 1, the web page contains the details of
a nasal spray with eight attributes: title, availability, product code,
manufacturer, price, description, ingredients and warnings. These
attributes can be considered as metadata used to describe the data.
Each value is associated with a meaningful name or metadata
label (except title) in order to help the user correctly interpret the
values of attributes. It is a common practice that the data
published into HTML pages are accompanied by metadata labels,
especially for e-Commerce web sites. For example, consider the
price information on the web: without the help of metadata labels
such as "suggested price", "member price", "shipping", and "tax",
these data would be completely misunderstood.

It shows that the textual labels used to interpret values of an
attribute do not occur with a wide range of words, which are
actually synonyms to each other. It can also be observed that
several different attributes usually occur together in a data-rich
region, i.e., they occupy a contiguous region in a data-intensive
page. Once such a set of synonyms for a domain has been
collected, it can be used to locate the metadata labels of interest in
the web pages. Some of these synonyms (not all possible
synonyms) can be collected easily, and this task does not require a
high level of expertise.

However, it still needs a method to automatically recognize the
data-rich regions of web pages in order to differentiate the data of
interest from the "decoration" and uninteresting parts. We use a
notion of structural-semantic entropy to measure the density of
occurrence of metadata labels on the DOM tree representation of
the HTML pages (see Section 3). If the density is greater than a
given threshold, it is most likely a data-rich region. Otherwise, it
is probably not. This threshold can be learned by experiments,
and do not need to be adjusted when we move to other application
domains.

3. WEB DATA EXTRACTION
The most challenging part of wrappers is that they must be able to locate
the data of interest among other uninteresting pieces of web
pages, such as advertisement regions, navigation bars, and inline
code. Our algorithm is based on the observation that the attribute-
value pairs of a record usually occur near to each other in the well
designed web pages. In the DOM tree representations of those
web pages, each record is composed of a set of sibling subtrees
(consecutive or interrupted by some noisy nodes), each of which
is an attribute-value pair of the record. Locating the data-rich
regions is equivalent to finding the lowest common parent nodes
of the sibling subtrees forming the records, and these nodes are
called data-rich nodes.

3.1 Structural-Semantic Entropy
A notion of structural-semantic entropy is used to identify and locate
the data-rich nodes. We define the structural-semantic entropy of a node in a DOM
tree in terms of the semantic roles of its descendant leaf nodes. The leaf nodes are all annotated
with their corresponding semantic roles, i.e., attributes of a product.
The annotation process can be accomplished by identifying
metadata labels in the web pages, and a leaf node that does not
belong to any of the semantic roles of interest is assigned to be
unidentified. For each attribute, a set of keywords used to indicate
this attribute is collected previously. Notice that it is only required
to collect some of these keywords, but not all of them, to run the
algorithm, and this task can be done easily without a high level of
expertise. The more keywords we collected, the better the results
will be.

Definition 1. The structural-semantic entropy $H(N)$ of a node $N$
in the DOM tree representation of a web page can be defined as

$$H(N) = -\sum_{i=1}^{n} p_i \log(p_i),$$

where $p_i$ is the proportion of descendant leaf nodes belonging to
semantic role $i$ of the node $N$, and conventionally the base of the
logarithm is 2.

Notice that it is not sufficient to only use the number of semantic
roles or the proportion of them to identify the data-rich regions,
since wrappers must be able to identify the data of interest among
other uninteresting parts of web pages, such as advertisement
regions and navigation bars, which may also include many nodes
annotated with the semantic roles. We use entropy to make sure
that not only there are enough numbers of semantic roles, but also
enough different types of such roles. Entropy is a measure of
disorder, or uncertainty in the system. More entropy means more
possible variation, and hence greater capacity for storing and
transmitting information. Conversely, by measuring a random
variable with higher entropy you are able to learn more. In our
case, the higher structural-semantic entropy a node has, the more likely the tree rooted at the node contains the data-rich region.

Algorithm 1. DE-SSE (Data Extraction from web pages based on Structural-Semantic Entropy)

```
Input: P: a web page
       R: a set of semantic roles for a given domain, each
           of which has a set of keywords \( K_r \) used to
           annotate the leaf nodes with the semantic role \( r \)
       \( H_d \): a threshold used to identify the data-rich nodes
       \( H_l \): a threshold used to identify the list nodes
Output: V: a set of attribute-value pairs of records

Begin
1:   cleans up the bad HTML tags and syntactical errors in P, and
2:   turns the body of P into a DOM tree, T.
3:   for each leaf node \( i \) in T do
4:      if the content of \( i \) matches any keyword in \( K_r \) then
5:         annotate \( i \) with the semantic role \( r \)
6:      if the content of \( i \) does not match any keyword
7:         annotate \( i \) with the unidentified role
8:      if \( i \) is annotated with \( d (d > 1) \) semantic roles
9:         separate \( i \) into \( d \) nodes, and annotate \( d \) nodes with their
          corresponding semantic roles
10:    traverse T in a breadth-first way, and sort all non-leaf nodes of T in the reverse order of the traversal sequence
11:   for each non-leaf node \( j \) in T do
12:      calculate the structural-semantic entropy \( e_j \) for \( j \)
13:     if \( e_j > H_d \) and \( j \) has a greater structural-semantic
14:        entropy than all its descendant nodes then
15:        \( j \) is a data-rich node, and makes all its descendant
16:        nodes non data-rich node
17:     if \( H_l < e_j < H_d \) and \( j \) is the common parent node of
18:        the sibling nodes that have the same nonzero values
19:        of structural-semantic entropy then
20:        \( j \) is a list node
21:     if any structural-semantic entropy of the sibling
22:        nodes is less than \( H_l \) then
23:        \( j \) is a link offer node
24:   end
25:   end
26:   insert the attribute-value pairs of a record into V
27: Return V
\{ DE-SSE \}
```

The two thresholds of \( H_d \) and \( H_l \) can be learned from experiments, and our experiments showed the best performance was achieved when set \( H_l \) to 2 (see Section 4). Representation tags and HTML attributes are deleted to reduce the running time of the algorithm since the experiments indicated that block-level tags, such as table, and list, contain a significant amount of useful information. For each attribute of a product, a regular expression is constructed to match the keywords used to interpret the values of the attribute (those keywords are synonyms for human), so the leaf nodes can be annotated with their semantic roles quickly. For example, the metadata shown in Figure 4 is used by Algorithm 1 to identify the nodes with the semantic role "Price" and check the possible

3.2 Algorithm Description

Our algorithm relies on the DOM tree representation of a web page, and traverses it in a bottom-up fashion in order to find the data-rich nodes and list nodes from the web page in terms of the structural-semantic entropy. The DE-SSE algorithm for locating data-rich regions and extracting the attribute-value pairs from the identified regions is described in Algorithm 1 that works on each web page automatically.

```
Figure 3. Sample DOM tree and the structural-semantic entropies of the nodes in the DOM tree

```
values for the role. The synonyms listed were employed to learn a regular expression for recognizing the "Price" nodes. These synonyms can be collected gradually. The more these synonyms are collected, the higher the precision and recall of the algorithm will be. For the attribute "Price", another regular expression (\d+(?:\d{2})?\$/ was constructed to check its possible values. The value matching regular expressions might be left undefined for the attributes such as "instructions" and "description". It is impossible or very difficult to define regular expressions for them because they usually are longer text. Those metadata can be reused when moving from one application domain to another.

**Figure 4. Metadata for the attribute "Price"**

<table>
<thead>
<tr>
<th>ATTRIBUTE: PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description: The amount as of monetary value, asked for or given in exchange for something else.</td>
</tr>
<tr>
<td>Synonyms: Price, list price, asking price, discount price, bid, cost, fee, fare, pay, rate, charge, toll, expense</td>
</tr>
<tr>
<td>Regular expression \d+(?:\d{2})?$/ for testing values:</td>
</tr>
</tbody>
</table>

The effort of the step 10 is to calculate the structural-semantic entropies of the nodes in a DOM tree in a bottom-up fashion because deciding the type of a node is relies on the states of its child nodes. When a node and its parent have the same entropy we make the child node a data-rich node unless the ancestor of the node has greater structural-semantic entropy. Actually, each data-rich node contains a record and the algorithm can identify record boundary automatically.

The content of the next text-node of the node that annotated with a semantic role is usually extracted as the value of the semantic role (or attributes). For some types of attributes, such as price and time, regular expressions are constructed to check if the strings are valid values for those attributes in order to increase precision. If the extracted string is not valid for the attribute, the content of the next-next node will be extracted until meets the node annotated with another semantic role. There are many situations in which the title of a record is not explicitly associated with a string to describe what follows is the title. However, the titles usually occur in the same relative position with respect to the data-rich nodes, and they can be extracted from the first leaf child of the data-rich nodes or the previous leaf of that node.

Our algorithm does not rely on the requirements that the web pages need to share the similar template or the multiple records need to occur in a single web page. The algorithm can extract the data of interest from one web page containing only one record. Once a set of keywords used to describe the data of interest is collected, such task that can be done easily by non-experts, the extraction process is fully automated. Only the set of keywords is changed as we move from one application to another. Wrappers generated by using our approach are inherently adaptable and resilient. They work for web pages from many distinct sources belonging to the same application domain, and continue to work well when the format of the source pages changes.

4. EXPERIMENTS

This section describes the data we used in our experiments and reports results of the experiments. For the experiments reported in this paper, we developed a web crawler that creates a copy of all the visited pages and runs a number of experiments on them. A HTML parser, named NekoHTML 1, is used to clean up the bad HTML tags, fix up syntactical errors, and turn web page into DOM trees. The DE-SSE algorithm has been used to conduct experiments on several sites. All experiments were conducted on an IBM XSeries X235 equipped with an Intel Xeon processor working at 2.40GHz, with 2GB RAM, running Linux and Sun Java development Kit 1.5.

We conducted two sets of experiments. The goal of the first one is to examine the effect of varying the threshold $H_d$ that used to identify the data-rich nodes has on the overall performance of the algorithm. The second is to see how well our algorithm could extract the attribute-value pairs from product descriptions across distinct sites. We created a data set containing randomly chosen 9035 pages (10 percent of the web pages collected) from ten websites in three different application domains (drugs, health food and cosmetics). The experimental results were obtained by comparing the data extracted by the DE-SSE algorithm to manually annotated data by six human volunteers who are non-project members. A set of keywords used in the experiments was collected by a non-expert within three hours through browsing some websites, and picking up the attributes and their local names (synonyms) from several different web sites.

4.1 Results

We measured the results of the data extraction techniques at two main stages of the algorithm: identifying the data-rich nodes of web pages, and extracting the attribute-value pairs of records. The reason for performing the evaluation at these two stages is that it allows us to separately evaluate the effectiveness of the techniques used at each stage, and the result of the first stage is critical for the success of the algorithm.

We use the standard metrics recall, precision and F-measure to evaluate the algorithm. In the first stage, recall is the ratio of the number of correct data-rich nodes identified by the algorithm to the total number of records that should be extracted, and precision is the ratio of the number of data-rich nodes correctly identified to the total number of all data-rich nodes identified by the algorithm. F-measure is the harmonic mean of precision $p$ and recall $r$, and is defined as $2 \cdot p \cdot r / (p + r)$.

We report in Table 1 a list of results relative to ten websites. For each site we have randomly selected a number of web pages as samples, which contains 9035 pages and 4456 records. We explored the effect of varying the threshold $H_d$ (defined in Section 3.2) from 1.00 to 2.75. The experiment showed that the higher the threshold $H_d$ the higher the precision and the lower the recall we will have, which is consistent with our intuition. The best performance can be achieved when set $H_d$ to 2. Our experiments also showed that once this threshold is learned it can be used when move to other domains without any loss in precision or recall.

As it can be seen from Table 1, in some web pages the algorithm failed to identify the data-rich nodes. There are two main reasons for these behaviors: use undefined metadata labels to describe the data, and contain such regions that contain a "link offer" with only one item. In the former case, more keywords can be collected to locate these metadata labels. The latter case is more

---

complicated, and it will result in a record to be extracted redundantly. Notice that link offers contain access paths to other richer pages usually with more detail information about the same record. In this case, the algorithm was unable to differentiate the data-rich nodes from the link offer nodes, and an additional post-processing step is required to delete duplicate records from the data extracted.

The results of the second stage are reported in Table 2 where we set $H_d$ to 2. Table 2 contains the following elements: (#pairs): the total number of attribute-value pairs that should be extracted; (#correct): the number of correct pairs extracted by the algorithm; (#incorrect): the number of incorrect pairs extracted; (recall): the number of correct extracted pairs divided by the number of pairs that should have been extracted; (precision): the number of correct extracted pairs divided by the number of all extracted pairs; (F-measure): the same as that defined above. It is counted as a correct pairs if both the attribute name and the corresponding value are correct.

As shown in Table 2, the recall, precision, and F-measure reach to respectively 87.89%, 96.85%, and 92.15%. Besides the causes of errors discussed above, most of the failures come from two cases. One is that the value of an attribute is empty, and it fails to locate the metadata label of the next attribute, so the label may be extracted as the value of the previous attribute that should be empty. The other is that an attribute may be of a nested structure that contains more than two levels and has internal nodes. For example, the web pages have a nested structure with a list of people, and for each person, a list of attributes including name, gender, age and etc., and for each name a list of first and last names. The algorithm could be further improved to handle the nested structures.

It is difficult to make empirical comparisons with other existing works due to the differences in inputs required and outputs expected. RoadRunner, for example, is one of the web data extraction systems available for download. RoadRunner and our algorithm all targeted for record-level extraction tasks, but RoadRunner requires as input multiple pages conforming to the same template while the DE-SSE algorithm consumes only one page at a time and has some priori knowledge about the schema of data. Our algorithm is able to associate the attributes of records with their respective values, and does not require training to make it still work properly when the format of the source pages changes. We use the analysis framework proposed in [3] to compare some existing IE systems (or tools) to ours, as shown in Table 3. We believe that our algorithm is more suitable for such applications that the sources cannot be known in advance, e.g. focused crawling applications.

### 5. APPLICATION

Thousands of online pharmacies have emerged in China for the last decade, and most of them are unauthorized by State Food and Drug Administration (SFDA). The internet advertisements created by those undercover websites lead to illegal distribution of drugs that could be dangerous for Chinese patients. SFDA lacks the effective measure to limit the ability of "rogue" online pharmacies.

---

### Table 1. Average F-measure for various thresholds used to identify the data-rich nodes

<table>
<thead>
<tr>
<th>Sites</th>
<th>#records</th>
<th>#pages</th>
<th>1.00</th>
<th>1.25</th>
<th>1.50</th>
<th>1.75</th>
<th>2.00</th>
<th>2.25</th>
<th>2.50</th>
<th>2.75</th>
</tr>
</thead>
<tbody>
<tr>
<td>51yao.com.cn</td>
<td>807</td>
<td>1082</td>
<td>95.04%</td>
<td>95.04%</td>
<td>95.15%</td>
<td>95.09%</td>
<td>95.19%</td>
<td>95.19%</td>
<td>99.07%</td>
<td>99.00%</td>
</tr>
<tr>
<td>360kxr.com</td>
<td>296</td>
<td>507</td>
<td>92.04%</td>
<td>92.04%</td>
<td>92.04%</td>
<td>92.04%</td>
<td>97.66%</td>
<td>97.58%</td>
<td>97.58%</td>
<td>90.77%</td>
</tr>
<tr>
<td>818.com</td>
<td>173</td>
<td>578</td>
<td>82.13%</td>
<td>82.30%</td>
<td>89.90%</td>
<td>91.32%</td>
<td>94.52%</td>
<td>94.34%</td>
<td>80.37%</td>
<td>71.95%</td>
</tr>
<tr>
<td>818shyf.com</td>
<td>350</td>
<td>881</td>
<td>94.98%</td>
<td>94.98%</td>
<td>94.98%</td>
<td>94.98%</td>
<td>93.39%</td>
<td>93.78%</td>
<td>92.50%</td>
<td>40.18%</td>
</tr>
<tr>
<td>baijk.com</td>
<td>328</td>
<td>976</td>
<td>89.34%</td>
<td>89.34%</td>
<td>89.47%</td>
<td>89.47%</td>
<td>91.34%</td>
<td>91.34%</td>
<td>97.76%</td>
<td>98.79%</td>
</tr>
<tr>
<td>bxdyf.com</td>
<td>258</td>
<td>467</td>
<td>96.99%</td>
<td>96.99%</td>
<td>96.99%</td>
<td>96.99%</td>
<td>97.54%</td>
<td>97.54%</td>
<td>97.54%</td>
<td>88.56%</td>
</tr>
<tr>
<td>jianke.com</td>
<td>167</td>
<td>1944</td>
<td>53.70%</td>
<td>53.70%</td>
<td>53.70%</td>
<td>50.23%</td>
<td>50.22%</td>
<td>50.23%</td>
<td>78.96%</td>
<td>32.35%</td>
</tr>
<tr>
<td>jxdyf.com</td>
<td>264</td>
<td>356</td>
<td>99.81%</td>
<td>99.81%</td>
<td>99.81%</td>
<td>99.81%</td>
<td>99.81%</td>
<td>99.97%</td>
<td>27.70%</td>
<td>18.21%</td>
</tr>
<tr>
<td>xingshitang.com</td>
<td>1617</td>
<td>1709</td>
<td>99.94%</td>
<td>99.94%</td>
<td>99.94%</td>
<td>99.94%</td>
<td>99.94%</td>
<td>99.97%</td>
<td>27.70%</td>
<td>18.21%</td>
</tr>
<tr>
<td>yfang.net</td>
<td>196</td>
<td>535</td>
<td>90.32%</td>
<td>90.32%</td>
<td>90.32%</td>
<td>90.53%</td>
<td>89.90%</td>
<td>79.89%</td>
<td>62.80%</td>
<td>31.76%</td>
</tr>
<tr>
<td><strong>Overall</strong></td>
<td><strong>4456</strong></td>
<td><strong>9035</strong></td>
<td><strong>94.34%</strong></td>
<td><strong>94.35%</strong></td>
<td><strong>94.67%</strong></td>
<td><strong>94.59%</strong></td>
<td><strong>95.12%</strong></td>
<td><strong>94.72%</strong></td>
<td><strong>65.07%</strong></td>
<td><strong>52.61%</strong></td>
</tr>
</tbody>
</table>

### Table 2. Results obtained from the attribute-value pairs extraction

<table>
<thead>
<tr>
<th>Sites</th>
<th>#pairs</th>
<th>#correct</th>
<th>#incorrect</th>
<th>recall</th>
<th>precision</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>51yao.com.cn</td>
<td>2403</td>
<td>2254</td>
<td>12</td>
<td>93.80%</td>
<td>99.47%</td>
<td>96.55%</td>
</tr>
<tr>
<td>360kxr.com</td>
<td>876</td>
<td>876</td>
<td>2</td>
<td>100.00%</td>
<td>99.77%</td>
<td>99.89%</td>
</tr>
<tr>
<td>818.com</td>
<td>492</td>
<td>471</td>
<td>20</td>
<td>95.73%</td>
<td>95.93%</td>
<td>95.83%</td>
</tr>
<tr>
<td>818shyf.com</td>
<td>1017</td>
<td>999</td>
<td>6</td>
<td>98.23%</td>
<td>99.40%</td>
<td>98.81%</td>
</tr>
<tr>
<td>baijk.com</td>
<td>981</td>
<td>915</td>
<td>96</td>
<td>93.27%</td>
<td>90.50%</td>
<td>91.87%</td>
</tr>
<tr>
<td>bxdyf.com</td>
<td>774</td>
<td>685</td>
<td>89</td>
<td>88.50%</td>
<td>88.50%</td>
<td>88.50%</td>
</tr>
<tr>
<td>jianke.com</td>
<td>501</td>
<td>409</td>
<td>93</td>
<td>81.64%</td>
<td>81.47%</td>
<td>81.56%</td>
</tr>
<tr>
<td>jxdyf.com</td>
<td>792</td>
<td>587</td>
<td>25</td>
<td>74.12%</td>
<td>95.92%</td>
<td>83.62%</td>
</tr>
<tr>
<td>xingshitang.com</td>
<td>4851</td>
<td>3898</td>
<td>23</td>
<td>80.35%</td>
<td>99.44%</td>
<td>88.88%</td>
</tr>
<tr>
<td>yfang.net</td>
<td>561</td>
<td>549</td>
<td>14</td>
<td>97.86%</td>
<td>97.51%</td>
<td>97.69%</td>
</tr>
<tr>
<td><strong>Overall</strong></td>
<td><strong>13248</strong></td>
<td><strong>11644</strong></td>
<td><strong>380</strong></td>
<td><strong>87.89%</strong></td>
<td><strong>96.85%</strong></td>
<td><strong>92.15%</strong></td>
</tr>
</tbody>
</table>
from reaching the consumers due to difficulties in detecting and tracking the illicit practices, which include:

- Selling counterfeit drugs;
- Selling drugs without SFDA approval;
- Selling drugs illegally imported into the country;
- Offering prescription drugs to be sold without a prescription;
- Selling controlled drugs;
- Advertising drugs with incomplete information that may mislead the customers.

A system shown in Figure 5 was developed for SFDA to enhance its ability to limit promotion to the consumers by "rogue" online pharmacies and hold all contributing parties accountable for conduct that results in vast profits at the expense of the public health. A crawl was designed with the capability of making use of the search engines such as Google and Baidu to find the online pharmacies or websites advertising drugs that are not known by the system through regularly querying the search engines using a set of predefined keywords such as "medicine", "drug", "dosage", etc. The metadata for the attributes describing drugs are stored in the metadata database, and are inquired by the DE-SSE algorithm in the process of data extraction. The extracted attribute-value pairs of every drug advertisement will be compared with the records in the database maintained by SFDA to decide which types of illicit practices, if any, the advertisement possibly leads to. All the extracted data and results of comparison will be stored in a database and uploaded to a data warehouse for reporting and analysis. The example rules as follows are used to recognize

![Figure 5. A web information extraction system for detecting false drug advertisements](image)
possible illicit advertisements by combining the data extracted and the records stored in the SFDA database:

\[ R_1. \neg \exists S (E.\text{authenticationCode} = S.\text{authenticationCode}) \rightarrow E \in \text{Unapproved} \quad [0.90] \]

\[ R_2. (E.\text{proprietaryName} = S.\text{proprietaryName} \lor E.\text{genericName} = S.\text{genericName} ) \land E.\text{authenticationCode} = S.\text{authenticationCode} \land E.\text{manufacturer} \neq S.\text{manufacturer} \rightarrow E \in \text{Counterfeit} \quad [0.85] \]

where \( E \) denotes a record extracted from the web page, and \( S \) a record stored in the SFDA database. The strings that follow \( E \) or \( S \) are attribute names of the records. Each rule is given with the information about the level of confidence we may have in it. The confidence level for a rule is indicated by a number in the interval \([0, 1]\), and such numbers were given by the experts from SFDA. The rule \( R_1 \) is: if an authentication code listed on the web page cannot be found in the SFDA database, the drug referred by the code is probably not approved by SFDA with 90% confidence. Every drug was approved for production and sale within China will be given a unique code. The rule \( R_2 \) is: if a drug's proprietary name or generic name, and authentication code are matched with the data in the SFDA database, but its manufacturer is not, the drug is more likely to be produced by unqualified manufacturer, and thus is counterfeit. SFDA will issues different authentication codes to different manufacturers, even for the same drug.

The result of testing whether two values of the attribute are equal is not simply true or false, but the Levenshtein distance between two value strings divided by the maximum length of two strings. The Levenshtein distance was defined as the minimum number of edits needed to transform one string into the other, with the edit operations being insertion, deletion, or substitution of a single character. The probabilistic logic has been used to deal with the reasoning under uncertainty. In probabilistic logic, probabilities are assigned to propositions to indicate levels of confidence, and the goal is to calculate the degree of confidence one can have in a conclusion derived from these propositions [21], which can be formulated as a linear programming problem.

The system has been deployed at SFDA, and it can provide the following key services:

- Detecting the online advertisements of drugs and health food, and extracting the values of fifteen attributes: proprietary name, generic name, authentication code, suggested price, discount price, manufacturer, country or region, availability, package, ingredients, indications, contraindications, warnings, dosage, and description;
- Rich query access to all the data extracted from the web;
- Reporting the newfound "rogue" online pharmacies or the websites containing false drug advertisements with the types of illicit practices, the numbers of records, IP addresses, host names, and physical locations (if possible) each week;
- Reporting the trend of growth in the online pharmacy market.

6. RELATED WORK

Many approaches in the literature have addressed the problem of data extraction from web pages. Programs that perform this task are referred to wrappers, and developing wrappers manually has many well known shortcomings. The key problem in developing wrappers is how to automatically or semi-automatically generate them. Related work on wrapper generation can be divided into (a) Grammar-based; (b) Template-based; (c) Ontology-based; (d) NLP-based approach. Below we review some representatives for these four types. We notice that the tools and systems covered here must not be regarded as complete. Some surveys of web data extraction systems and tools can be found in [2] and [3].

Grammar-based approaches: One of the first initiatives was the development of languages specially designed to assist users in constructing wrappers. Minerva [4], a formalism for wrapper development, combines a declarative grammar-based approach with the flexibility of procedural programming. It provides an explicit exception-handling mechanism inside of the grammar parser. The grammar used by Minerva is defined in the EBNF style. Hong and Clark [5] describe a principled method for generating extraction wrappers using grammatical inference. They take the set of stochastic context-free grammars to capture general structures of web pages. Domain-specific knowledge in the form of declarative rules is required to complete the process of data extraction from the web pages. One major shortcoming of this technique is that programming wrappers require manual coding which generally entails extensive debugging. This task is obviously labor-intensive and time-consuming, and requires a high level of expertise. In addition, since the format of web pages is often subject to change, the wrappers by their nature tend to be brittle and difficult to maintain.

Template-based approaches: Such techniques make assumption that repetitive patterns occur in a web page, or multiple sample pages conform to a common template. In other words, the data records are formatted in a consistent manner that the occurrences of each attribute in several records are formatted in the same way, and they always occur in the same relative position with respect to their contexts. The extraction processes are based on algorithms that compare the tag paths and structures of the sample pages, and take the matches as the part of templates and the mismatches as the values of data records.

IEPAD [6] is one of the first systems that attempts to generate repetitive patterns from unlabeled web pages. The discovery of repeated patterns is realized through a data structure called PAT trees and string alignment techniques. IEPAD exploits the fact that if a web page contains multiple data records, they are often rendered regularly using the same template for good visualization. It is probable for IEPAD to induce incorrect patterns along with the correct ones, and human work in post-processing of the output is still required. In addition, IEPAD fails in the situation where pages containing single data record.

RoadRunner receives as input multiple pages belonging to the same template, and uses them to generate a schema that can be used to extract the data contained in the pages conforming to the template [7] [8]. It works by comparing the tag structure of the samples pages and inducing a union-free regular expression that handle structural mismatches found between the two structures. One limitation of the approach is that the web pages for training are required to be generated from the same template, so human effort is still required to prepare these web pages before the data extraction. Crescenzi et al. [8] partially solve this problem by developing an approach to identify the different pages classes in the target sites based on tag structures and URL similarity. Their approach, however, focuses on the web pages from the same site, and it is not applicable to the important applications that need to extract information from different websites to provide topic-specific information. In addition, since it fails to associate the attributes of the schema with their values due to lack of domain
knowledge, the column must be manually named after the data has been extracted.

Álvarez et al presented a method for detecting a list of structured records in a web page and extracting the data fields from the list [9]. The method begins with detecting the target list by finding the node with maximum repetitive path patterns from the root to its leaf nodes on the DOM tree representation of a web page. Then, string alignment techniques and edit-distance similarity algorithm are used to separate the list into record. One limitation of the approach arises in the pages where the attributes forming a data record are not contiguous in the page. For instance, the attributes belonging to the same product are presented in two regions and are separated by three button elements in the example page shown in Figure 1. In addition, the method requires a single page containing a list of structured data records as input, and thus will fail in the situation where pages containing single data record.

Ontology-based approaches: Such approaches first construct an ontology that describes a set of concepts, relationships between these concepts, and keywords within a domain of interest. Then the ontology helps to identify and extract data from unstructured documents, and transform it into structured form according to the scheme produced by parsing the ontology. One of representative ontology-based approach is the tool developed by the Brigham Young University Data Extraction Group [10]. Though their approach does not require training examples and is insensitive to changes in the format of web pages, a significant manual effort is still required to define the domain ontology by someone who is skilled with regular expressions, ontology theory and domain knowledge. For each application domain, a new ontology must be constructed. In addition, their method for recognizing ontology instances seems to use many ad hoc heuristics that are restricted in the domains and structures they can recognize.

NLP-based approaches: Natural language processing (NLP) techniques have been used by several tools to learn extraction rules for extracting data from free texts. These tools usually apply such techniques as filtering, part-of-speech tagging, lexical semantic tagging, and parsing to build relationship among phrase and sentence elements. The most representative NLP-based tool is the KnowItAll system that extracts facts from a collection of web pages starting with a seed set of factual patterns that are either manually specified or semi-automatically engineered [11]. The NLP-based tools are usually more suitable for web pages consisting of free texts, possibly in telegram style, such as job lists, and apartment rental advertisements. But, it can be seen that the product descriptions are not often full sentences, which makes them easier to be scanned quickly.

Several other works have also discussed the web data extraction techniques that do not belong to the above categories. Vadrevu et al presented an IE system that transforms a web page into a semi-structured hierarchical document using presentation regularities and domain knowledge by a statistical model [12]. Wang and Lam developed a framework that jointly extracts information and conducts mining from multiple web pages by an undirected graphical model, called conditional random field, which models the interdependence between neighboring text fragments within a single web page, as well as text fragments from different web pages [13]. Probst et al [14] also described an approach to extract attribute-value pairs from product descriptions as we did in this paper, but they formulated the data extraction as a classification problem, and use a Co-EM (expectation maximization) algorithm along with Naïve Bayes. Cohen et al [15] have addressed the problem of how to wrap tabular data in HTML documents which is complementary to our work.

7. CONCLUSION

We described an approach to automatically locate the data-rich regions, and extract the relevant attribute-value pairs of records from web pages across different sites. Our approach is based on the observation that the attributes and their values of the records usually occur near to each other in well designed web pages. In this paper, we focus on how to identify the data-rich regions, and show that locating the data-rich regions is equivalent to finding the lowest common parent nodes of the sibling subtrees forming the records in the DOM tree representation of a web page. After annotating the leaf nodes with their corresponding semantic roles with the help of a set of domain keywords, the structural-semantic entropy is calculated for each node in a DOM tree. The higher entropy a node has, the more likely the tree rooted at the node contains the data-rich region.

A little effort is required to generate wrappers by our approach, and the generated wrappers are insensitive to changes in web-page format. Experiments on a large number of real-life web page collections yield promising results, and the approach has been successfully applied to false drug advertisement detection due to its capacity in associating the attributes of records with their respective values. In such applications as false advertisement detections, the data extracted from web pages should be compared with the data stored in the local databases in attribute-level to decide which is false or not, and the wrappers also need to be able to handle the sources that cannot be known in advance.

Although the focus of this paper is on the problem of web data extraction, our work raises the critical performance issue of how to efficiently extract the data from web pages based on the notion of structural-semantic entropy. The current algorithm requires that the entropy should be calculated for every non-leaf node of a DOM tree. One of the possible optimizations is to find rules to terminate the calculation before the entropies of all nodes are calculated in a bottom-up way. A second direction is to accelerate data extraction for the pages conforming to the same template that we have observed several times during the process of crawling a web site. An evaluation method also should be developed to help in selecting the best value for a given attribute among multiple candidates extracted from different possible positions in order to increase precision.

8. ACKNOWLEDGMENTS

The work was supported by a grant from the National Natural Science Foundation of China (No. 60903078) and a grant from Shanghai Leading Academic Discipline Project (No. B114). The authors thank Mu Zhu, Yong Fang, Bo Yu, Jiao Li, Hao Chen, and Hui Bai for contributions to the experiments of this work.

9. REFERENCES


[20] Zhai, Y. H., and Lui, B. Structured data extraction from the web based on partial tree alignment. IEEE Transactions on Knowledge and Data Engineering, 18(12), 2006, 1614-1628.