Automatically Extracting Domain Ontology Based on Semantic Characteristics of Web Tables *

Song-il CHA 1,2, Z. M. MA 1,*, Xiao-long JIAO 1

1 College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
2 Department of Computer Science, The University of Science, Pyongyang 999093, DPR Korea

Abstract

Since tables themselves are organized structurally and semantically, they are good resources from which we can easily extract ontology. This work propose algorithms to automatically generate total instances or properties belonging to ontology class from Web table, taking several instance or attribute belonging to class composed of proper noun as the seed. Through experiment, we show that our method can rapidly extract proper nouns which adopt to interpret the table.

Keywords: Domain Ontology; Web Table; Seed; Proper Noun Extraction; Instance; Property

1 Introduction

Recently, the Semantic Web comprises techniques that promise to dramatically improve the current WWW and its use. With the emergence of the Semantic Web and the growing number of heterogeneous data sources, the benefits of ontologies are becoming widely accepted. Web ontologies define terms used as data (metadata) for explaining things of a special domain. Nowadays, researchers are paying attention to automatic transformation of Internet resources in the areas into ontologies.

Previous studies for constructing domain ontologies from the Web table are centralized to interpret table structure: Jung et al. [1] suggested a method for extracting table-schemata based on table structure and heuristics. Using this method, a table is converted into a table-schema and a triple. Chen et al. [2] employ heuristic rules to filter out non-genuine tables from their test set and make assumptions about cell content similarity for table recognition and interpretation. Wang et al. [3] proposed a machine learning based approach to classify given table entity as either genuine or non-genuine. Pivk et al. [4] focused on understanding table-like structures only due to their structural dimension and transforming the most relevant table types into F-logic frames.

*Project supported by the National Natural Science Foundation of China (NO.60873010 and NO.61073139) and the Fundamental Research Funds for the Central Universities (N090504005, N100604017 and N090604012), and in part by the Program for New Century Excellent Talents in University (NCET-05-0288).

*Corresponding author.
Email address: mazongmin@ise.neu.edu.cn (Z. M. MA).

March 2012
Though Jung [5] proposed heuristics for detecting semantic characteristics based on the location of table cells, he did not mention which becomes semantic core element. Through the observation about semantic features of the table, we found that if there are proper nouns on the table, then they can become a semantic core element. In this paper, we propose an automatic extraction method of the instance composed of proper nouns. In order to rapidly and accurately obtain brief domain ontology consisting of proper nouns, we employ automatic learning method based on bootstrapping [6].

The paper is structured as follows: In the next Section 2 we present an automatic extraction method of the instance belonging to given class from Web tables; In Section 3, we describe a property extraction method based on proper noun extraction; In Section 4, we evaluate our method according to the experiment; Finally, in Section 5, we provide conclusions to our work.

2 Automatically Extracting Instance From the Table

In order to automatically generate semantically correct ontology, certainly, must be based on some clues. This section describes method generating automatically other instances in the table, taking some proper nouns such as already familiar product name as a clue.

2.1 Proper noun extraction model based on bootstrapping

Bootstrapping is shown as follows: firstly, generate a pattern from the document in accordance with a small amount of seed terms, then using this pattern again extract other words from the document, and lastly, using the extracted terms create another word. We can extract a large amount of terms from a small amount of seed terms by a repeat of this process. To begin with we define the fundamental notions.

**Definition 1** A proper noun $p$ is a noun that is the name of a specific individual, place, or object. For example, personal name, country name, denomination name, organization name, and so on. A proper noun set $P$ is a set of proper nouns, i.e. $p \in P$.

**Definition 2** A seed term $s$ is a proper noun specified artificially before proceeding with automatic learning. A seed term set $S$ is a set of the seed terms, i.e. $s \in S$.

From the above definition, we can know that $s \in P$ and $S \subset P$.

A domain table set $T$ is a set of genuine tables which is chosen in the Web tables of given domain, collected using search engine such as Google or Yahoo. In this paper, in order to obtain genuine tables of given domain, we use the algorithm proposed in the previous study [1]. Withal, search keys for obtaining of the domain table set are a domain name (that is, a class name) and the seed term set selected by user in the beginning.

The proper noun extraction model based on bootstrapping is shown below. In this paper, the model known as PNB-Model (Proper Noun Bootstrapping-Model). Fig.1 shows PNB-Model. The input of the Model is the domain table set $T$ and the initial seed term set $S$, the output is the proper noun set. A quality of the seed term set greatly affects the accuracy of the extracted proper nouns. Therefore, users must select more obvious and important terms for the initial seed term set. In addition, through experimental study, we confirmed that can increase the accuracy of the proper noun extraction in the case of that $Ns$ (the number of the seed terms) is three or
more. The number of the seed list used in this model is always \( N_s \). A threshold value \( N \) of the seed list is the number of an old seed term among the seed terms entered in the each loop of bootstrapping. In the beginning, \( N = N_s \). Institute the threshold value \( N \) of the seed list as greater one than two. This means that it must contain at least two initial seed terms in the seed terms used at the every loop. That is, can effectively extract terms when \( N \geq 2 \). But if not, can’t guarantee accuracy of the extracted terms. The seed list is possible combinations of initial seed terms composed of greater one than two.

![Fig. 1: PNB-Model](image)

At the first loop period of bootstrapping, this model extracts terms taking \( N_s \) initial seed terms. But, at the second loop period, it extracts possible candidate terms taking \( N_s - 1 \) initial seed terms and a new seed candidate. Likewise, at the third loop period, it extracts possible candidate terms taking \( N_s - 2 \) initial seed terms and two new seed candidates. The loop period finishes when \( N < 2 \).

In the pattern production process, we institute the TABLE tags that wrap each of the seed term from the table. In the table selection and extraction process of the candidate term, model selects tables that contain the seed term pattern from the domain table set and extract all rest cells which appears with the seed terms in same row or same column of the table. In the selection and evaluation process, model firstly, only adds the obtained candidate terms which do not overlap to the proper noun set. Secondly, selects the seed candidates for next loop among the proper noun set and add to a seed candidate set. Finally, evaluates the threshold value \( N \) of the seed list, and finishes when \( N < 2 \), if not repeats the loop period.

### 2.2 Instance extraction algorithm

In this section, we propose the detailed algorithm for extracting the instances based on PNB-Model. Our goal is to rapidly and accurately generate the domain ontology. Therefore, we used tree seed terms, i.e. \( N_s = 3 \). Algorithm 1 shows the instance extraction algorithm.

\begin{algorithm}
\begin{algorithmic}
\State \textbf{Step 1} input \( S, T \).
\State \textbf{Step 2} choose \( s_1, s_2, s_3 \in S \), and \( X \leftarrow \{ s_1, s_2, s_3 \} \); \( X_0 \leftarrow \{ s_1, s_2, s_3 \} \); \( P \leftarrow s_1 \cup s_2 \cup s_3 \); \( C_0 \leftarrow \emptyset \).
\State \textbf{Step 3} create pattern which wraps \( S \) from \( T \).
\State \textbf{Step 4} using pattern that wraps \( X_0 \), construct candidate term set \( C \) from the table in \( T \).
\end{algorithmic}
\end{algorithm}
Step 5 from $C$ remove terms that don’t correspond to evaluation condition.
Step 6 from $C$ remove all $c$ that $c \in \text{Cbigwedge} \in P$, it’s result also remains to $C$, and $P \leftarrow P \cup C$.
Step 7 if isn’t $|X| \geq 2$, then output $P$, and end algorithm.
Step 8 if processed all elements in $C_0$, then from $S$ select new element pair and replace $X$ by it; If $C_0 \neq \phi$, then $C_0 \leftarrow C$.
Step 9 select new seed candidate set $X_0$ from $X$ and $C_0$, and go to step 4.

In this algorithm, $X$ is an element pair which consisted by arbitrary combination of elements in the seed term set $S$, for example $\{s_1, s_2, s_3\}, \{s_1, s_2\}, \{s_1, s_3\}, \{s_2, s_3\}, \{s_1\}, \{s_2\}, \{s_3\}$, and so on. $C_0$ is a candidate term set extracted when the seed term is $\{s_1, s_2, s_3\}$. $C$ is a candidate term set which is consisted at the rest every loop period. $X_0$ is a seed term set that is used as input of every loop period. In the beginning, the algorithm enters the bootstrapping process, taking $\{s_1, s_2, s_3\}$ which consist of three terms. Then add the new extracted terms to the candidate term set $C$. In the second loop, takes two old (before used) terms and a newly extracted term as a seed term. Because we can’t consider that the new extracted terms are certainly right instances. This will prevent that the accuracy is lower, and takes another new elements as a seed. Fundamental steps of algorithm are shown as follows:

### 2.2.1 Pattern production

In order to produce pattern, we found TABLE tags which wraps each of the seeds from arbitrary table of the domain table set. We can institute the $\langle td \rangle$ term $\langle /td \rangle$ used for defining table cells as a pattern. Every table’s creator is using TABLE tag of different description form, but this algorithm employs only $\langle td \rangle$ tag which denotes table cell for the purpose of pattern production. Withal, for brief descriptive purpose, we consider only the case of non-use of $\langle td \rangle$ tag attribute such as ALIGN, VLIGN, EIDTH, HIGHT, BACKGROUND, BTCOLOR, and so on.

### 2.2.2 Extracting the candidate term

In order to construct the candidate term set, we assume as follows: If certain term appears with the proper noun in same row or same column of a table, then this term also is a proper noun. Therefore, we take the row or column of table as object of candidate term extraction. Before extracting the candidate term, we must determine table reading orientation, i.e. row wise or column wise [2]. For example, if the first row of the table consists of the attribute cells, and the others are value cells, then this table is the column wise. If the table is row wise, then take each cell of the row in which exists proper noun as candidate term. And if the table is column wise, then take each cell of the column in which exists proper noun as candidate term.

### 2.2.3 Evaluation

In order to extract more accurate candidate term, we must resolve table HEAD. Withal, we must determine candidate term extraction range in the row (or column) in which exists proper noun, that is, we must resolve value region corresponding to the table HEAD. In this step we evaluate whether each term $c$ of the candidate term set $C$ is a term belonging to a given class, and choose
only fit terms. Withal, we evaluate whether newly extracted proper noun exists in the proper noun set \( P \), and only add proper noun which does not overlap to \( P \).

## 3 Automatically Extracting Property from the Table

### 3.1 Observation on the table property

It is difficult to extract inclusively, if property also as well as instance does not depend on any clue. Therefore, in this section we propose an automatic extraction method of property belonging to the class using proper noun extraction approach based on bootstrapping.

In order to extract the property from the Web table, firstly, based on the proper noun extraction method mentioned above, extract the instance belonging to the class, and take a set of these instances as \( P \). Then, determine three initial seed property belonging to the class, and denote them as \( a_1, a_2, a_3 \). Denote a set of the properties as \( A \), i.e. \( a_1, a_2, a_3 \in A \). For example, in the first row of the table on Fig.2, properties such as Type, Standard, Resolution, and so on, are property belonging to camera model class, and elements of the first column, are instances of camera model class. As shown in the figure, when the table is row wise, generally, the properties are constructed by the cells which lie at the first row of the table. When extracting the property using the proper noun extraction method, the seed term contains three properties and one instance which is selected from instances collected already. In each product there are special and common attributes. For example, not only digital camera but also computer has “resolution” attribute, that is, this is the common attribute. When we take only three properties as the seed property, it has possibility which can extract even properties belonging to other class. Thus, in order to extract rightly the property belonging to given class, it additionally contains one instance of the seed term set \( S \).

### 3.2 Property extraction algorithm based on the proper noun extraction

Property extraction algorithm based on the proper noun extraction method similar to the instance extraction algorithm. Property discussed in this algorithm is Owl:DatatypeProperty [7, 8]. Algorithm 2 shows property extraction algorithm.

**Algorithm 2: Property Extraction Algorithm**

**Step 1** input \( A, P, T, \) and \( R \leftarrow s_1 \cup s_2 \cup s_3 \).
Step 2 create pattern which contains $A$ and $P$ from $T$.

Step 3 $X \leftarrow \{s_1, s_2, s_3\}$, and take arbitrary $p \in P$; $X_0 \leftarrow \{p, s_1, s_2, s_3\}$; $C_0 \leftarrow \phi$.

Step 4 using pattern that $X_0$ is contained, construct candidate property set $C$ from table in $T$.

Step 5 from $C$ remove properties that isn’t DatatypeProperty.

Step 6 from $C$ remove all $c$ that $c \in C \land c \in R$, it’s result also remains to $C$, and $R \leftarrow R \cup C$.

Step 7 if isn’t $|X| \geq 2$, then go to step 10.

Step 8 if processed all elements in $C_0$, then ${\text{from } A \text{ select new element pair and replace } X \text{ by it; If } C_0 = \phi, \text{ then } C_0 \leftarrow C}$.

Step 9 select new seed property set $X_0$ from $P$, $X$, $C_0$, go to step 4.

Step 10 if processed all elements in $P$, then output $R$, and end algorithm, else go to step 3.

In this algorithm, $X$ is an element pair which is consisted by arbitrary combination of elements in the seed property set $A$. $C$ is a candidate property set which is consisted at the every loop period. $R$ is a property set. $X_0$ is a seed term set that is used as input of every loop period. In the beginning, the algorithm enters the bootstrapping process, taking $s_1, s_2, s_3$ which consist of three properties and an instance of $P$. And the second loop, takes two old (before used) properties, a newly extracted property, and an instance of $P$ as a seed term. If all elements of property set were processed, then replace the instance by a new element and repeat process. In the subsection below, stepwise instantiate major steps of algorithm.

4 Evaluation

4.1 Evaluation of the instance extraction method

In our experiment, in order to rightly construct the seed term set, we selected twenty publishing companies based on revenue rank of the world’s largest publishing companies in 2008 from Web pages. Next, we collected arbitrary two hundred Web tables among around 60,700 Web pages of the “publishing” domain using search key “publishing company”, “McGraw-Hill Education”, “Wiley”, and “Reed Elsevier” of Google search engine. Then, according to the instance extraction algorithm, progress the proper noun extraction. Previous researches endeavored to interpret the table by using structural characteristics, hence, it is difficult to experimentally compare with our method. Withal, it is troublesome to rightly measure recall and precision of the experimental results obtained by our method. Therefore, in this experiment, we evaluated that the proposed instance extraction algorithm how to effectively increase the instances about every time operation. For the above purpose, we compared with twenty instances extracted above, i.e. watch obtaining process of correct twenty instances. Fig.3 shows the comparison of the experimental results.

We can interpret the recall and precision as follows: Firstly, discuss the recall. In the case of the simple input, the system registers singly the correct instance depending on the input sequence. Therefore, the recall is proportional to input time, in the case of seventeenth input only arrive at 100%. In the case of our proper noun extraction, is extracted a large amount of candidate instances at first time, at second time the recall arrives at 64%. However, at next time the candidate term is overlapped, and rising speed is slow, at sixth time arrive at 100%. Through
this, we confirmed that the proper noun extraction method can increase effectively the instance from a small amount of user input. Next, discuss the precision. In the case of the simple input, the precision is 100%. But, in the case of our proper noun extraction, because irrelevant term was extracted, the precision decreases and at sixth time also arrive at 59%. To contain only twenty publishing companies in the correct option set was a cause of low precision.

### 4.2 Evaluation of the property extraction method

Our property extraction method employs the instances which extracted by the instance extraction algorithm. For an experiment, we collected products which have comparatively many attributes such as mobile phone, digital camera, LCD TV, LCD monitor, CRT monitor from Web pages. In order to illustrate experiment process for property extraction, take mobile phone as an example. Firstly, we select three attributes belonging to the mobile phone as the seed property, i.e. \{dimensions, weight, display size\}. Next, extract the instances belonging to mobile phone using the instance extraction algorithm. Then, according to the property extraction algorithm, extract the properties taking \{mobile phone instance1, dimensions, weight, display size\} as the seed term.

Likewise, we can extract properties belonging to other class. Table 1 shows the result of total experiment.

<table>
<thead>
<tr>
<th>Product class</th>
<th>No. of web tables</th>
<th>Recall(%)</th>
<th>Precision (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile phone</td>
<td>117</td>
<td>79.4</td>
<td>66.9</td>
</tr>
<tr>
<td>Digital camera</td>
<td>143</td>
<td>78.0</td>
<td>52.8</td>
</tr>
<tr>
<td>LCD TV</td>
<td>139</td>
<td>86.8</td>
<td>69.0</td>
</tr>
<tr>
<td>LCD monitor</td>
<td>97</td>
<td>82.3</td>
<td>61.2</td>
</tr>
<tr>
<td>CRT monitor</td>
<td>138</td>
<td>88.7</td>
<td>64.5</td>
</tr>
</tbody>
</table>

As shown in the table 1, the recall arrives at 80~90%. Through this, we confirmed that it can obtain almost the properties belonging to a given class. Withal, the precision totally arrives at 60~70%. It is very ideal to extract the properties from Web pages which are described only actual attributes of the product. But, in this experiment, did not choose from Web pages applied like that, therefore, the precision is decreased.
5 Conclusions

In this paper, we proposed an automatic extraction method of instances and properties from the Web tables. Our approach was able to extract fast a great amount of instances and properties using bootstrapping from a small amount of seed terms entered by human. We consider that this approach offer a good basis not for construction of arbitrary domain ontology fit with our intent, but for right semantic interpretation of table structure. In our future work, a more precised method for extracting ObjectProperty and other relationships will be developed based on extracted proper nouns.

Acknowledgement

The author thanks Samuel Erhabor and Wei Yan for participating in the perfection of the paper.

References