Information Extraction to feed Digital Library Databases

Alejandro Bia        Rafael Muñoz
abia@dlsi.ua.es      rafael@dlsi.ua.es
Bib.Virtual Miguel de Cervantes  Dpto. Lenguajes y Sistemas Informáticos
Tel: 34-96-5903400 #9567  Tel: 34-96-5903653  Fax: 34-96-5909326
Universidad de Alicante, apartado de correos 99, E-03080, España

20 de febrero de 2002

Resumen

Most often, Digital Libraries have the need to extract information from poorly marked-up documents to fill databases or create new hypertext documents with a highly structured markup. In this work, we approach the problem of extracting bibliographic information from literary reports in HTML format to fill a Digital Library database of Galician publications used for Internet searches. An information extraction approach that takes advantage of both HTML markup and Natural Language Processing (NLP) techniques was successfully used for this purpose.

KEYWORDS: Information extraction, digital libraries

1. Introduction

As Sperberg & Burnard [9] states: “A descriptive markup system uses markup codes which simply provide names to categorize parts of a document.” The advantages of structural or descriptive markup, which defines the structural components of a document, compared to procedural markup, that merely defines display or printing formatting features (fonts, sizes, emphasized characters) are clear [2]. Descriptive markup languages, like XML, add some semantic value to the text that can be useful for further processing like extracting information to fill databases or generate diverse formats of documents, summaries or listings. But usually, information sources come with little or no descriptive markup, like scanner-OCR output ¹, or electronic documents in formats like RTF, PDF, PS or HTML.

¹The best OCR programs nowadays can reproduce the original procedural formatting, but not more.
2. The problem

In this work, we approach the problem of extracting bibliographic information from literary reports in HTML format, written in Galician language\(^2\) to fill a database of bibliographic references for Digital Library queries. These reports consist of free text, that contains descriptions of books that appear as clusters of sentences that describe each book. These clusters include bibliographic details such as author-name, title, place of publication, publisher, date or year, number of pages, ISBN, an abstract of the book, etc. Some of these entities are always present, others are optional, like graphic artist, cover design, collection or issue number. Other free text paragraphs that appear apart from the book information clusters need to be ignored. These paragraphs may contain headers, introductions, literary comments, and so on. The formatting is consistent (in the sense of titles, and font sizes conventions being maintained throughout the whole text) but there are differences in format and data field delimiters among these book descriptions that make the use of a conventional parser impractical for IE.

On one hand, HTML, being compliant of the SGML standard, is not as rich as XML in descriptive-structural markup capacity. It defines some basic structural elements (such as title, body, headers, paragraphs, etc), but its strength lies in procedural markup aspects. So the texts to parse are well structured, but poorly marked for our purpose (without lexical and syntactic tags). On the other hand, from the multiple applications that information extraction (IE) has, perhaps the most interesting is the introduction of data extracted from non-structured texts into a database (DB), where the goal is to obtain templates filled with the information that must be stored in the DB \([4, 8]\).

Our problem is halfway between completely non-structured and highly-structured documents. In our case we have poorly structured HTML texts with a consistent formatting. This formatting can be used to preselect the relevant sections of text, that will then undergo an IE process with the help of a few NLP resources like dictionaries. This will identify the specific pieces of information to be extracted and stored in a DB.

3. Brief survey of Information Extraction

IE is not a new idea, we can find its roots in the mid-60s. But it is in the 80s when IE begins to grow quickly. This is due to the DARPA\(^3\) intervention, which fomented the competition between different research groups to develop IE systems. These meetings gave place to the Message Understanding Conference (MUC). The conferences’ objective was to establish a quantita-

---

\(^2\)Language used only in some north-west provinces of Spain

\(^3\)DARPA is the agency of defense of the United States
tive regime of evaluation for IE systems. These conferences established the text assembly on which all the systems were evaluated.

Four tasks were defined as fundamental in MUC-6. Following the definitions made by Cunningham [5], these four tasks are:

- **Named Entity Recognition (NE).** The objective of this task is to identify and to classify all person names, organizations, places, dates, and amounts of money mentioned in the text. These names are called entities.

- **Co-reference Resolution (CO).** The objective of this task is identifying identity relations between entities recognized in the texts. This task establishes relations between entities recognized by the previous task as well as relations between an entity and an anaphoric expression (pronoun, definite description, adjective, etc.).

- **Template Element production (TE).** The objective of this task is to add descriptive information to the NE results.

- **Scenary Template extraction (ST).** The objective of this task is to represent the output of the IE system tying together TE elements and relation descriptions.

In MUC-7 a new task was defined named **Template Relation (TR).** The objective of this task is to capture the following three relations in the text: *employed of, located in* and *product of*.

Most systems are based on the structure proposed by Grishman [6], shown in figure 1. They are observed to have modular structure. This architecture consists of a series of independent modules that perform the basic tasks of IE systems. Moreover, a module’s input is the output of the previous module.

LaSIE-II [7] and Proteus [10] are the most representative systems that follow Grishman’s architecture.

4. **The method**

This method combines HTML markup techniques and Natural Language Processing techniques. As can be seen in figure 2 our method has these two different stages.

4.1. **Application of HTML markup techniques.**

At this stage a text in HTML format is taken by a segmenter which extracts all the clusters of relevant information based on HTML markup (figure 3 shows a cluster). In this way, much of the irrelevant text is skipped. To do this, we must observe first the formatting features of the clusters we
Figura 1: Architecture of an Information Extraction System
are interested in and take note of the changes that may indicate where the information we are looking for is. This is what a human does when doing fast reading: changes in font size, highlighting, titles, justification, etc., helps us to identify the parts of the text where the information we are looking for may be (in this case bibliographic descriptions), and ignore the rest of the text.

This previous observation of the source material is converted to filtering instructions in the segmenter program. When the text has a consistent formatting, this helps us to extract potential clusters of interesting information with high precision (100% in our case, where the text shows a consistent formatting and adequate segmenting rules are used). This reduces the problem of IE to processing only this clusters and not the whole source text. Besides, we already know there is a one to one correspondence between bibliographic descriptions and clusters as the study of the formatting suggested. As a result, this segmentation process separates noise from signal.

In next stage, for each one of the extracted clusters a template with the relevant information will be created, using NLP techniques.

4.2. Application of NLP techniques.

4.2.1. Named entity recognition.

In this second stage we use a set of heuristics deducted from a study of a fragment of the texts to be processed (analogous to a training corpus) to extract the relevant information (entities) from the clusters. Among others, we must extract information about author, title, publisher, etc. All the en-
Figura 3: Cluster

Entities share a set of characteristics. These characteristics make up the set of heuristics rules of which we distinguish two types. On one hand, we have a set of heuristic rules about information of where each type of entity appears (author, title, etc.). We will refer to these rules as general patterns. On the other hand, we have a set of specific heuristic rules that recognize each entity type (author, title, publisher, etc.).

These specific heuristic rules are based on different features for entity detection: 1) names consist of words that begin with capital letters 2) some entities include special words or abbreviations that act like triggers, (e.g. words as “Ilust.” (graphic artist), eds. (editors) or ISBN). The intrinsic difficulty of this stage is to identify the beginning and end of each entity. One of the problems we find is the appearance of prepositions, articles and conjunctions, within the entity that we want to recognize (e.g. “del” in the name “Fernández del Riego” or “A in the location “A Coruña”).

The following rules are some of the heuristic rules used (every component between “≪” and “≫” is an optional component):

- General patterns
  - Author, Title, City: Editors, Year, Pp. “(“Identification ”)”.
  - Author, Title. Comments, City: Editors, year, Pp. “(“Identification ”)”.
  - Author, Title, City: Editors, Collection, Year, Pp. “(“Identification ”)”.
  - Author, Title. Comments, City: Editors, Collection, Year, Pp. “(“Identification ”)”.

- Fine-grained patterns
4.2.2. Template manipulation

1. Template filling.
A specific label is added to each recognized entity in order to mark each attribute of the template to be filled up (figure 4). Having recognized all entities of the cluster, we obtain a tagged cluster. Later, a process fills the corresponding template slot by relating each label with a specific slot. In this way, we obtain a template for each cluster as can be seen in figure 5 which shows a filled template.

2. DB storage.
All these templates, with all mandatory slots filled (although some of the optional attributes may be empty), are stored in the relational database. We obtained a database with the bibliographic references of all the literary works that appear in the clusters.
Figura 5: Filled template

Figura 6: Screen shot of application
Cuadro 1: Results obtained

<table>
<thead>
<tr>
<th>File name</th>
<th>Clusters</th>
<th>NE Recognition</th>
<th>Filled Template</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gal005.html</td>
<td>25</td>
<td>75</td>
<td>20</td>
<td>80.0 %</td>
</tr>
<tr>
<td>Gal006.html</td>
<td>23</td>
<td>84</td>
<td>20</td>
<td>86.9 %</td>
</tr>
<tr>
<td>Gal007.html</td>
<td>31</td>
<td>106</td>
<td>26</td>
<td>83.9 %</td>
</tr>
<tr>
<td>Total</td>
<td>79</td>
<td>265</td>
<td>66</td>
<td>83.5 %</td>
</tr>
</tbody>
</table>

5. Evaluation

This approach has been evaluated on different fragments of HTML text (different from those used to deduct the heuristics). On one hand, it achieves a precision of 83.5% in full template filling. Moreover, this approach achieves a precision of 92% in the task of author, title and location recognition.

Table 1 shows the results obtained after processing 3 HTML files. These files are made up of 79 different clusters. Every cluster corresponds to a template. Our method recognizes 66 templates successfully. In the NE task (named entity) our method achieves a precision of 92% (243 of 265 entities).

The figure 6 shows an example processed by our application. We can see a cluster without HTML marks extracted from the HTML file into a text box at the right side (stage 1). At the left side, we can see all the template slots (filled or not).

6. Conclusions and Future works

We have developed an IE system as a tool to fill an information bibliographic database from HTML texts. This approach proved to be good in extracting the relevant information. We used both segmentation based on procedural marks and Natural Language features to detect and extract the information. The source text was well structured but poorly marked-up. The Natural Language techniques used, based on heuristic rules, helps to detect entities in order to create or update databases. The structural knowledge acquired through this IE process can also be used to generate new documents with a richer descriptive markup [3, 2]. The only NLP resource needed was a dictionary of names and places to achieve a precision of 83.5%.

Referencias


\(^4\) Precision is the ratio between the number of correctly solved templates and the number of processed clusters.


