Mining Literary Texts by Using Domain Ontologies

Miriam Baglioni[1], Mirco Nanni[2], Emiliano Giovannetti[2]

[1] Department of Computer Science, University of Pisa,
Via Buonarroti 2, 56100, Pisa, Italy
baglioni@di.unipi.it
[2] Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo" , KDD Laboratory – CNR
Via G. Moruzzi 1, 56124, Pisa, Italy
{mirco.nanni, emiliano.giovannetti}@isti.cnr.it

Abstract. This paper describes a query system on texts and literary material with advanced information retrieval tools. As a test bed we chose the electronic version of Dante's *Inferno*, manually tagged using XML, enriched with a domain ontology describing the historical, social and cultural context represented as a separate XML document.

1 Introduction

One of the key conditions for transforming large quantities of texts into effective repositories of knowledge is to allow a user to "search by an idea". Text search tools cannot still match an expert looking for relevant information in a document collection.
Answers of an expert are intelligent as he has, at his disposal, a lot of data and he is able to compute all the available information using a sophisticated reasoning process. To match the answers of a human being, the system should to be able to reply to queries such as: *which are Dante's attitudes towards holders of feudal power?* and *is there a statistical correlation between the belonging to a feudal system and salvation, or between belonging to city system and damnation?* Answering to these queries requires the evaluation of different types of knowledge: text content knowledge and context knowledge. Some queries will be solved using only one of the available sources of information while others will require a comparison between them.

Only recently the use of intelligent techniques in humanistic fields has been receiving attentions from researchers thanks to the design of new tools for text representation and the application of advanced markup tools for conceptual manipulations. The markup of several aspects of a text (e.g. typographical conventions, semantic features, and so on) is the goal of the Text Encoding Initiative. By using SGML texts can be stored obtaining a rich meta-representation of their multilevel information [2]. Hypertext representation can be used for the realization of a theory of narrative evolution, typical of certain literary trends [3]. In "The World of Dante" Project a hypermedia environment for the study of *Inferno* of *Divina Commedia* has been implemented [4]. New techniques, aimed at processing the meaning of a text, require the treatment of the knowledge "not only of the text itself, but of the world" [5]. About data mining

applications to literary texts the majority of the efforts has been devoted to the discovery of frequent expressions (patterns) within texts of particular authors without taking into account implicit knowledge information [14].

In replying to queries like the ones reported above, there is the need of added information that describe both the text and the context. To do this we have manually tagged the text (the *Inferno* canticle of Dante's *Divina Commedia*) and defined an *a priori* domain ontology containing the contextual information.

The chosen test bed has proven to be extremely structured both from the textual and the knowledge point of view, and for this reason ideal to demonstrate the power of querying systems like the one presented in the paper.

2 System Architecture

The system architecture is composed of six components: a **user interface** to interact with the user; a **query manager** to evaluate the queries and to decide if to recover stored knowledge or to create new knowledge; a **XML query executor** to execute queries which do not need to apply data mining procedures; a **DM executor** to execute queries which need data mining procedures; a **repository** to store the results of queries and a **knowledge base** containing the meta-representation of a text and the domain ontology. The role of the interface is to create chunks of conceptual elements that will be used by the search mechanism. This mechanism will map conceptual clusters, produced by the analysis of a query, onto the two partitions of the knowledge base. In the search module, retrieval, data mining and discovery tools are triggered; they exploit background knowledge of the domain and knowledge procedurally extracted from the meta-representation.

The knowledge base has been structured taking into account the general partition into textual and contextual knowledge. The text **meta-representation** represents the XML mark-up of Dante's Divina Commedia. It contains semi-structured data representing the whole text of *Inferno* while the **ontology** represents the conceptual base of the domain to provide all the information not directly derivable from the text. The considered meta representations will be used to feed the knowledge extraction process, via the connected query processing process, to allow the interpretation of user questions and the construction of intelligent answers.

3 The Knowledge Discovery Process

Knowledge discovery is the process of extracting useful and novel information from large databases [6], [7], [8]. Imielinski and Mannila [10] pointed out that the KDD process can be seen as a query process and they defined the base paradigm for the design of a query language to obtain a KDD Management System (KDDMS). The basic requirement that a KDD query language must satisfy is the property of closure, together with the support for two classes of objects: KDDObjects and KDDQueries. KDDObjects are the models extracted from data and KDDQueries are predicates

returning a KDDObject or a database object. KDDObject can be both generated at run time and stored in a repository. In the last case the query language has to be able to retrieve the stored objects.

We chose an environment composed of two abstraction levels, using **KDDML** [11] (KDD Markup Language) at the lower level and **MQL** [12] (Mining Query Language) at the higher one. KDDML is a query language born to handle the KDD process with the aim of making the interoperability between DM algorithms and KDD tools in general easy to obtain. The basic idea is that both queries and models are represented by XML definitions, each model being checked for validity over its corresponding DTD[1]. We put MQL on top of KDDML because in the latter also queries are expressed as XML documents, lacking user-friendly input methods. MQL language definition is targeted to maintain all the functionalities owned by KDDML and, at the same time, to be easier to use. Furthermore, it has an algebraic foundation that allows us to prove basic properties of the extracted models. It is worth noting that both the languages have been designed to fit Imielinski and Mannila's requirements to eventually obtain a KDDMS.

We chose this system, MQL and KDDML together, as a supporting environment to the knowledge base query processing – XQuery – because the XQuery results are XML documents that can be built in a suitable way to match the definition of an element of the KDD environment.

4 The Information Retrieval Process

The information retrieval process operates through the use of XQuery and MQL; it can generally be divided into three distinct steps. The first one is known as *query formulation*. Queries to which the systems must provide an answer represent the informative need expressed by the user. Some of them require simple information that can be directly extracted from the text and the ontology, e.g., "*How many and which are the mythological characters that appear in the Inferno?*". Other kinds of queries require the application of data mining techniques to discover the information hidden in data, e.g., "*Classify the religious characters in accordance with Dante's attitude: Benevolent, not Benevolent*". Hence, the formulation of the initial query determines the choice of the most adequate technique to transform the informative problem into queries. The second step to be considered is known as *query modeling* and it concerns the automatization of the data pre-processing. In general, the application of a mining algorithm requires a particular format of the input data, and so the initial data set must be stored into the data repository in one of the allowed formats. This step has been realized through the formulation of queries in XQuery. The last step is known as *result visualization* and it concerns ways in which the obtained answers are presented to the user. With our system the obtained results are visualized with a browser as HTML pages, no matter if they are the result of a XQuery or they represent a model extracted by mean of a mining algorithm.

[1] We are working to include PMML DTDs as model specifications

5 Running Example

The following is a knowledge extraction problem whose goal is to build a classification tree to predict Dante's attitude toward a character on the basis of his/her characteristics (or *features*). The adopted strategy to model the problem requires to select the needed elements, to carry out a pre-processing phase on these elements to let them fit the required input format and to apply on them a mining algorithm to extract a classification tree.

```
<Table>
{ for $a in document("text.xml")        //Dialogues
  for $c in document("ontology.xml")    //Characters
  where $c/KnownAs = $a/CharactersDialogue/Name and
        $a/AttDante != ""
  return <Dialogue>
         {   $c/Type,
             $c/Epoch,
             …
             $a/AttDante
          </Dialogue>  }
</Table>
```

Fig. 1. XQuery to obtain Dante's attitudes

In the first step, known as *element selection*, the idea is to determine the data set to be classified and then to obtain a training set to predict the classification attribute. We can formulate the XQuery (Fig. 1) that selects from the informative sources (*ontology.xml* and *text.xml*) the elements that are considered useful for the classification.

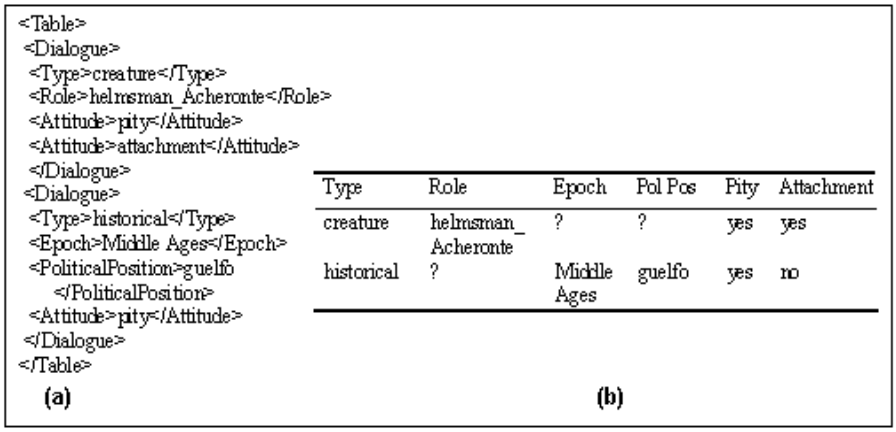

Fig. 2. (a) Data set in XML format. (b) Representation of the relative data set

The previous query returns an XML document in which each element is a sequence of characters' features for all characters present in both of the considered documents.

In the second step, known as *training set construction*, we want to model a query that, starting from a generic XML document (e.g. derived by the one produced by the query of Fig.1) produces a document that fits the DTD for type table. Since we do not know a priori which of Dante's attitude we are interested in, we decided to add at the end of each character's features all the different attitudes, and then to build a classifier specifying as the class, the chosen one. So we have a XQuery query that, starting from a document as the one depicted in Fig. 2(a), produces another XML document which represents the table shown in Fig. 2(b). In this table the number of rows depends on the number of dialogues while the columns are the elements that represent each dialogue with added the target attributes (which are considered to be valid only if they have boolean type) depending on which classification tree is built.

In the last step, *decision tree construction*, we can perform the classification task on the previous representation of data, in order to obtain a decision tree through an invocation of the C4.5 classification algorithm [12]. The corresponding MQL query is shown below:

```
begin query
        attitude_tree = CreateTree pity
                                from attitudes_table C4_5 true true o.5 1
end query
```

In general, the *CreateTree* operator used to build the classifier has to be provided with three groups of input parameters: the target attribute of the classification, a table object which represents the training set and the specifications of the classification algorithm to be used. In this case, the target is the attribute *pity*, the training set is the file stored as *attitudes_table.xml* in the data repository and the classification algorithm selected is *C4_5*. From the viewpoint of an expert on Dante or on Italian studies, the results obtained so far could be not correct and reliable. The fact is that the knowledge base has been completed with values which have not been provided by domain experts. Such operation was necessary due to the lack of data. In particular, the added values regard the *Attitude* element, which describes Dante's and the other characters' mood shown during a dialogue. In fact, the structure of the text requires that an element *AttDante* and an element *Attitude* are inserted at the end of the annotation of each dialogue; their content is deduced from the annotated dialogue. As a consequence, in order to produce correct values for the knowledge base, the dialogue has to be critically analyzed by a domain expert. Hence we plan to evaluate the system as soon as we obtain consistent tagging information about the text and the context that must be provided from experts of the field.

6 Conclusions

The application described in this paper shows that text processing can be profitably improved by the integration of advanced knowledge representation tools. Even literary texts, whose structure is inherently complex, due to the subtle elaboration of every linguistic level, can be interpreted from a semantic viewpoint, once an adequate

representation of their domain is specified. Current knowledge representation languages are able to account for the structure and the relationships of the world of a text and of its "external" context. Moreover, they are able to discover inherent conceptual chunks hidden in the representation. The system is then able to provide a user with useful answers with respect to all data at its disposal. Answers are typically computed through the computation of different types of knowledge, each type pertaining to a different partition in the conceptual organization of an author's world.

This work focused on the application of knowledge discovery techniques to mine the Commedia of Dante for the extraction of information not directly retrievable neither from the semi-structured text nor from the ontology. We have introduced an example through which the main steps of the knowledge discovery process have been described.

References

1. Cappelli, A., Catarsi, M. N., Michelassi, P., Moretti, L., Baglioni, M., Turini, F., Tavoni, M.: Knowledge mining and discovery for searching in literary texts. In: Proceedings of the Third International Conference on Language Resources and Evaluation. Las Palmas (2002)
2. Sperberg-MaQueen, C. M., Burnard, L. (eds.): Guidelines for Electronic Text Encoding and Interchange, Chicago and Oxford: Text Encoding Initiative (1994)
3. Sutherland, K.: A Guide Through the Labirint: Dickens's Little Dorrit as Hypertext. Literary and Linguistic Computing, Vol. 5, No. 4 (1990), 305-309
4. URL: http://etcweb.princeton.edu/dante/index.html
5. De Vuyst, J.: Knowledge Representation for Text Interpretation. Literary and Linguistic Computing, Vol. 5, No. 4 (1990), 296-302
6. Fayyad, U. M., Piatesky-Shapiro, G., Smith, P.: From Data Mining to Knowledge Discovery: An Overview. In: Fayyad, U. M., Uthurusami, R., Smith, P., Piatesky-Shapiro, G. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI/MITT Press (1996)
7. Piatetsky-Shapiro, G., Frawley, W. J.: Knowledge Discovery in Databases. AAAI/MIT Press, Cambridge, MA (1991)
8. Han, J., Mamber, M.: Data mining: concepts and techniques. Morgan Kaufmann (2000)
9. Frawley, W. J., Piatetsky-Shapiro, G., Matheus, C. Knowledge Discovery In Databases: An Overview. In: [7], 1-30 (1991)
10. Imielinski, T., Mannila, H.: A Database Perspective of Knowledge Discovery. In: Communication of the ACM, 39 (11): 58-64 (1996)
11. Alcamo, P., Domenichini, F., Turini, F.: An XML based environmen in support of the overall KDD precess. In: Proceedings of the Fourth International Conference on Flexible Query Answering Systems. Advances in Soft Computing, Springer-Verlag (2000) 413-424
12. Baglioni, M., Turini, F.: MQL: An Algebraic Query Language for Knowledge Discovery. In: Proceedings of 8th Congress of the Italian Association for Artificial Intelligence. Lecture Notes in Computer Science. Springer (2003) 225-236
13. Quinlan, J. R.: C4_5: Programs for Machine Learning. Morgan Kaufmann Publ. Inc. (1993)
14. Takeda, M., Matsumoto, T., Fukuda, T., Nanri, I.: Discovering characteristic expressions from literary works: A new text analysis method beyond n-gram and KWIC. Theor. Comput. Sci., (2001).