Automatic Management of Annotations on Cultural Heritage Material

G. Semeraro, F. Esposito, S. Ferilli, T.M.A. Basile,
N. Di Mauro, L. Iannone, I. Palmisano
Dipartimento di Informatica – Università di Bari
Via E. Orabona, 4 - 70125 Bari
{semeraro, esposito, ferilli, basile, nicodimauro, iannone}@di.uniba.it
ignazio_io@yahoo.it

Abstract

The COLLATE project is concerned with digitised historical material. One of the main features of COLLATE system architecture is the integration of software components that exploit state-of-the-art techniques coming from the area of Artificial Intelligence and Knowledge Representation (KR). This work describes the results achieved by applying machine learning methods for automatic classification and labelling of documents. Furthermore, we also discuss the advantages obtained by exploiting brand new research achievements in KR for the design of COLLATE data model.

Introduction

The IST-1999-20882 project COLLATE (Collaboratory for Annotation, Indexing and Retrieval of Digitized Historical Archive Material) aims at developing a WWW-based *collaboratory* (Kouzes R.T. et al., 1996) for archives, researchers and end-users working with digitised historical material (http://www.collate.de). The chosen sample domain is a large corpus of multi-format documents concerning rare historic films from the 20s and 30s (see Figure 1), provided by three major European film archives: DIF (Deutsches Filminstitut, Frankfurt am Main), FAA (Film Archive Austria, Vienna) and NFA (Národni Filmový Archiv, Prague). One of the main features of COLLATE system architecture is the integration of software components

Figure 1 : Sample Collate Documents

that exploit state-of-the-art techniques coming from the area of Artificial Intelligence and closely related research, such as Knowledge Representation (KR). This work describes the results achieved by applying machine learning methods for automatic classification and labelling of documents and the advantages obtained by exploiting brand new research achievements in KR for the design of COLLATE data model. The need of automatically labelling such a huge amount of documents suggested the use of machine learning techniques to learn rules for such tasks from a small number of selected and annotated sample documents, as well as the development of a software component devoted to the management of annotated documents.

The challenge comes from the low layout quality (stamps that overlap to components) and standard (documents are typewritten sheets, i.e., all equally spaced lines in Gothic type) of such a material, which introduces a considerable amount of noise in its description. In particular, the complexity of the domain and the need that the rules are understandable by film experts, led to the choice of symbolic first-order logic learning. Furthermore, the possibility that the document collection will in the future include new documents calls for incremental learning models. Such considerations led to exploit INTHELEX (Esposito F. et al., 2000) as a learning component, because many of its features meet these requirements, as confirmed by experimental results.

The Learning Component

INTHELEX (INcremental THEory Learner from EXamples) carries out the induction of *hierarchical* first-order logic theories from examples. It learns simultaneously *multiple concepts*, possibly related to each other; it guarantees validity of the theories on all the processed examples; it is able to refine a previously generated version of the theory, but also to start learning from scratch.

The learning cycle performed by INTHELEX may be summarized as follows. A set of examples of the concepts to be learned, possibly selected by an expert, is provided by the environment. This set can be subdivided into training, tuning, and test examples, according to the way in which examples are exploited during the learning process. Specifically, training examples, previously classified by the expert, are exploited to obtain a theory that is able to explain them. Subsequently, the validity of the theory against new available examples, after storing them in the example base, is checked by taking the set of inductive hypotheses and a tuning/test example as input and producing a decision that is compared to the correct one. In the case of incorrectness on a tuning example, the cause of the wrong decision can be located and the proper kind of correction chosen, firing the theory revision process. In this way, tuning examples are exploited incrementally to modify incorrect hypotheses according to a data-driven strategy. Test examples are exploited just to check the predictive capabilities of the theory, intended as the behavior of the theory on new observations, without causing a refinement of the theory in the case of incorrectness on them. Another peculiarity of INTHELEX is the integration of multi-strategy operators that may help solve the theory revision problem (Ferilli, 2000). The purpose of induction is to infer regularities and laws that may be valid for the whole population. INTHELEX incorporates two inductive refinement operators, one for generalizing hypotheses that reject positive examples, and the other for specializing hypotheses that explain negative examples. Deduction is exploited to fill observations with information that is not explicitly stated, but is implicit in their description. Indeed, since the system is able to handle a hierarchy of concepts, some combinations of predicates might identify higher level concepts that are worth adding to the descriptions in order to raise their semantic level. Abduction aims at completing possibly partial information in the examples, adding more details. Its role in INTHELEX is helping to manage situations where not only the set of all observations is partially known, but each observation could also be incomplete. Abduction can be exploited both during theory generation and during theory checking to hypothesize facts that are not explicitly present in the observations. Lastly, abstraction removes superfluous details from the description of both the examples and the theory. The exploitation of abstraction in INTHELEX concerns the shift from the language in which the theory is described to a higher level one. The abstraction operators are applied automatically to the learning problem before processing the examples.

Experimental Results

The experimental dataset consisted of 102 documents from the three classes of interest above reported, plus 17 reject documents obtained from newspapers articles . The first-order logic descriptions of documents, needed to run INTHELEX, were automatically generated by the system WISDOM++ (Esposito F. et al., 2000). Each document was considered as a positive example for the class it belongs to, and as negative for the other classes; reject documents were considered as negative examples for all classes. Since each kind of document is composed by layout blocks having different roles it, the class the document belongs to must be learned before starting to learn definitions for the semantic labels in it. Hence, a first experiment, aimed at learning definitions for each class, starting from the empty theory, was carried out. The predictive accuracy of the resulting theories was tested according to a 10-fold cross validation methodology. Table 1 reports the experimental results, averaged on the 10 folds, of the classification task, as regards: number of clauses defining the concept (Cl.), Accuracy on the test set (expressed in percentage, Acc.) and Runtime (in seconds).

Table 1 Statistics for Classification

	Cl.	Acc.	Runtime
DIF	1.00	99.17	17.13
FAA	3.50	94.17	334.05
NFA	2.25	95.74	89.88

After this preliminary phase of classification, a further experiment was performed aimed at learning rules to identify the blocks of which the documents are made up. As regards the class of documents from FAA archive, the domain experts provided the following labels characterizing the objects belonging to it: registration_au, date_place, department, applicant, reg_number, film_genre, film_length, film_producer, film_title. The labels specified for class of documents belonging to DIF were: cens_signature, cert_signature, object_title, cens_authority, chairman, assessors, session_data, representative. Table 2 shows the results of a 10-fold cross-validation run on these dataset.

Table 2 Statistics for Understanding FAA and DIF

FAA	Cl.	Acc.	Runtime
registration_au	5.6	91.43	3739
date_place	6.9	86.69	7239
department	1.9	98.95	118
applicant	2	97.89	93
reg_number	5.1	91.95	4578
film_genre	4	93.02	2344
film_length	5.5	90.87	3855
film_producer	4.9	94.05	4717
film_title	5.4	89.85	4863

DIF	Cl.	Acc.	Runtime
cens_signature	2.2	98.32	1459
Cert_signature	2.2	98.31	176
object_title	5	94.66	3960
cens_authority	2.9	97.64	2519
chairman	4.6	93.10	9332
assessors	4.6	94.48	12170
session_data	2.5	97.68	1037
representative	5.6	92.98	13761

Finally, the documents of the class NFA were characterized by the following labels, almost all different from the others: dispatch_off , applic_notes, n_cens_card, film_produc, no_prec_doc, applicant, film_genre, registrat_au, cens_proc, cens_card, delivery_date. Again, a 10-fold cross-validation was applied, whose averaged results are reported in Table 3.

As expected, the classification problem turned out to be easier than the interpretation one (that is concerned with the semantics of the layout blocks inside documents).

Table 3 Statistics for Understanding of NFA

NFA	Cl.	Acc.	Time	NFA	Cl.	Acc.	Time
dispatch_off	6.8	94.28	13149	applicant	6.7	93.66	3739
applic_notes	2.5	98.81	231	film_genre	2.8	98.53	684
n_cens_card	5.3	95.47	8136	registrat_au	4.1	94.64	5159
film_produc	4.9	93.98	5303	cens_proc	4.8	98.51	4027
no_prec_doc	4.6	93.97	5561	cens_card	5.6	94.62	3363
delivery_date	4	95.52	3827				

This is suggested by the increase in number of clauses and runtime from Table 1 to Tables 2 and 3. Such an increase is particularly evident for the runtime, even if it should be considered that the high predictive accuracy should ensure that very few documents will cause theory revision.

The XML Engine

XML Content Manager (XMLCM) is the software component devoted to the management of information flow within COLLATE. It is a set of software entities that manipulate information at different levels of abstraction. As its name suggests, it is based on the eXtensible Mark-up Language (XML) technology by W3 Consortium[1]. Figure 2 provides an overall sketch of the whole XMLCM architecture. It can be seen as a set of layers relying on each other, where each layer has a different abstraction level, designed to tackle one single issue in the whole system. Persistence Layer is devoted to guarantee an effective physical storage of XML resources. The design of this layer is based on the *Strategy* pattern (Gamma E. et al., 1995), allowing for the implementation of different physical persistences, in order to exploit the better solution for a specific problem, thus enabling the component to be reused in a wide variety of systems. Three implementations have been developed: binary compression on file system (relying on an API called PDOM[2]), RDBMS storage of XML (based on Oracle 9i[3] RDMBS and its XML capabilities), and native XML Databases (based on XML dedicated Database server such as Software AG's Tamino[4]). The last one has been developed within the project COVAX (Licchelli

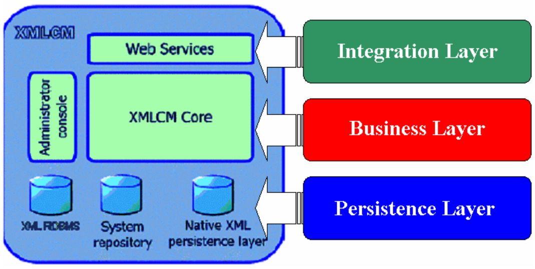

Figure 2: XMLCM Overall architecture.

[1] http://www.w3.org/xml
[2] http://www.infonyte.com/en/prod_pdom.html
[3] http://otn.oracle.com/products/oracle9i/content.html
[4] http://www.softwareag.com/tamino

O. et al., 2003). The Business Layer grants all the typical operations on an XML resource. They range from the simple creation, update, deletion and retrieval of a document (or a bunch of them), to the manipulation of basic XML documents subunits called elements. Again for each element (or bunch of elements) creation, updating, deletion and retrieval are possible. This layer embeds a language for querying single o multiple documents fully compliant with XQL specification (Xavier E., 2001). Besides this, the component provides support for document versioning. More than a version can be stored for each document: the system tracks version histories providing support for comparisons and merging of two or more versions of the same XML resource. Also, in order to navigate through versions and carry out ordinary XQL-query among documents, the system can be instructed with a proprietary XML based language that allows to restrict the scope of XQL query on a set of versions adjusting some parameters (e.g.: authors, date, etc.). The Integration Layer is responsible for integration of XMLCM with external systems. It has been designed aiming to interoperability and, for this reason, we used the Web Services (Curbera et al, 2002) paradigm. This makes able any application to exploit all XMLCM services using a very simple communication protocol such SOAP[5]. Web Services seem to be the most promising response to distribution and decentralization needs of Internet based application. Developing such a layer we guaranteed the possibility of *moving* single subsystems of a wider architecture (say COLLATE whole system) across the internet. Moreover this paradigm guarantees loosely coupling between system components resulting in high maintainability of the whole architecture.

RDF Management

XML alone cannot guarantee the whole support needed by COLLATE. In fact, the need of enriching documents belonging to the film heritage through the addition of annotations by film scientists, and consequently the need of performing search on documents as well as on their annotations, require COLLATE system to be able to manage what has been called "scientific discourse" (Frommholz I. et al., 2003). As an example, final users had to be able to navigate among heterogeneous resources following strongly typed links among them. The most common resources were the annotations that film scientist made upon documents. All these annotations were

[5] www.w3.org/TR/SOAP

considered scientific discourses on a given resource in the archive. In such a view the need for a well established technology for addition of metadata to COLLATE document arouse. This brought us to adopt more powerful KR languages - Resource Description Framework (RDF)[6] and its evolution DAML+OIL[7] (Horrocks I., 2002), (Decker et al, 2000) – and to develop a specific XMLCM architectural layer dealing with RDF. This component, in addition to usual operations on RDF Models and Statements, has some extra features not available in other tools, such as the support to RDF resource sharing among multiple users (both human and software agents). Furthermore there is support for multi query language capabilities. The need for having more than a query language for RDF (and its derivatives such as DAML+OIL) was the complete lack of standards in querying RDF resources. We currently allow for querying RDF resources using many query languages such as RDQL , RQL (Karvounarakis G. et al., 2002) and SquishQL. Another feature of that layer is the compliance with DAML specification that, together with OIL, gives the possibility of accomplishing basic reasoning processes, such as classification of instances, with the theoretical support of Description Logics (DL) research. DAML+OIL is a specific DL language, thus XMLCM can be easily integrated with state-of-the-art DL reasoners, such as FaCT (Horrocks I., 1998), process which is under development.

References

Curbera F., Duftler M., Khalaf R., Nagy W., Mukhi N., Weerawarana S. 2002
Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI,
IEEE Internet Computing: 86-93

Decker S., Melnik S., Van Harmelen F., Fensel D., Klein M., Broekstra J., Erdmann M. and Horrocks I. 2000
The Semantic Web: The roles of XML and RDF
IEEE Internet Computing Vol. 15 No.3:63-74

Esposito F., Malerba D. and Lisi F.A. 2000
Machine learning for intelligent processing of printed documents.
Journal of Intelligent Information Systems, 14(2/3): 175-198.

Esposito F., Semeraro G., Fanizzi N. and Ferilli S. 2000
Multistrategy Theory Revision: Induction and Abduction in INTHELEX.
Machine Learning, 38(1/2): 133-156
Kluwer Academic Publ., Boston.

[6] http://www.w3.org/RDF

[7] DARPA Agent Manipulation Language + Ontology Inference Layer

Ferilli S. 2000

A Framework for Incremental Synthesis of Logic Theories: An Application to Document Processing.

Ph.D. thesis, Dipartimento di Informatica, Università di Bari.

Frommholz I., Brocks H., Thiel U., Neuhold E., Iannone L., Semeraro G., Berardi M. and Ceci M. 2003

Document-centered Collaboration for Scholars in the Humanities - The COLLATE System. pp: 434-445

Proceed. of 7th European Conference on Digital Libraries, edited by T. Koch and I.T. Solvberg, Lectures Notes in Computer Science, Springer: Berlin.

Gamma E., Helm R., Johnson R., Vlissides J. 1995

Design Patterns.

Addison-Wesley Pub Co; 1st edition.

Horrocks I. 1998

Using an expressive description logic: FaCT or fiction? pp. 636-647

In *Principles of Knowledge Representation and Reasoning: Proc. of the Sixth Int. Conference*, edited by Cohn, A. G., Schubert, L. and Shapiro S. C.

Morgan Kaufmann Publishers.

Horrocks I. 2002

DAML+OIL: a Reason-able Web Ontology Language.

In *Advances in Database Technology - EDBT*, pp. 2-13, edited by Jensen, C. S. et al.

Lectures Notes in Computer Science 2287, Springer: Berlin.

Karvounarakis G., Alexaki S., Christophides V., Plexousakis D., Scholl M. 2002

RQL: A Declarative Query Language for RDF.

The 11th International WWW Conference (WWW'02).

Kouzes R.T., Myers J.D. and Wulf W.A. 1996

Collaboratories: Doing science on the internet.

IEEE Computer, 29(8): 40-46.

Licchelli O., Esposito F., Semeraro G. and Bordoni L. 2003

Personalization to Improve Searching in a Digital Library. pp. 46-55

In *Proc. of New Technologies for Information Systems*, Proceed. of the 3rd Int. Workshop on New Developments in Digital Libraries, in conj. with ICEIS, edited by P. Isaías, F. Sedes, J.C. Augusto and U. Ultes-Nitsche.

Xavier E. 2001

Using Extensible Query Language (XQL) for Database Applications.

In Proc. of 8th Annual IEEE ECBS '01, Washington DC pp.2-9

http://www.computer.org/proceedings/ecbs/1086/1086toc.htm