Incremental Induction of Classification Rules for Cultural Heritage Documents

Teresa M.A. Basile, Stefano Ferilli, Nicola Di Mauro, and Floriana Esposito

Dipartimento di Informatica - Università degli Studi di Bari
via E. Orabona, 4 – 70125 Bari - Italia
{basile, ferilli, nicodimauro, esposito}@di.uniba.it

Abstract. This work presents the application of a first-order logic incremental learning system, INTHELEX, to learn rules for the automatic identification of a wide range of significant document classes and their related components. Specifically, the material includes multi-format cultural heritage documents concerning European films from the 20's and 30's provided by the EU project COLLATE. Incrementality plays a key role when the set of documents is continuously augmented. To ensure that there is no performance loss with respect to classical one-step systems, a comparison with Progol was carried out. Experimental results prove that the proposed approach is a viable solution, for both its performance and its effectiveness in the document processing domain.

1 Introduction

Many important historic and cultural sources, which constitute a major part of our cultural heritage, are fragile and distributed in various archives, which still lack effective and efficient technological support for cooperative and collaborative knowledge working. The IST-1999-20882 project COLLATE (Collaboratory for Annotation, Indexing and Retrieval of Digitized Historical Archive Material) aims at developing a WWW-based *collaboratory* [6] for archives, researchers and end-users working with digitized historic/cultural material (URL: `http://www.collate.de`). The chosen sample domain includes a large corpus of multi-format documents concerning rare historic film censorship from the 20's and 30's provided by three major European film archives, specifically, Deutsches Film Institut (DIF), Film Archive Austria (FAA) and Národni Filmový Archiv (NFA). In-depth analysis and comparison of such documents can give evidence about different film versions and cuts, and allow to restore lost or damaged films, or to identify actors and film fragments of unknown origin. The COLLATE system aims at providing suitable task-based interfaces and knowledge management tools to support film experts' individual work and collaboration in analyzing, indexing, annotating and interlinking such documents. Continuously integrating valuable knowledge about the cultural, political and social contexts into its digital data and metadata repositories, it will provide improved content-based functionality to better retrieve and interpret such a material.

In this environment, the automatic induction of rules that are able to recognize the document classes and their significant components in order to provide them to the film

R. Orchard et al. (Eds.): IEA/AIE 2004, LNAI 3029, pp. 915-923, 2004.

experts would be very helpful. In particular, the complexity of the available documents layout structure suggests the use of symbolic (first-order logic) descriptions and techniques. Good results in this field would be a strong motivation to extend the application of the presented techniques to other kinds of documents of interest in the field of office automation. In this perspective, one possible application would be the classification of new incoming documents, e.g. invoices/letters/advertisements, and store them by type in a database. This led us to try applying the INTHELEX learning system to this domain.

The following section presents INTHELEX, an incremental learning system, along with its reasoning strategies; then, Section 3 reports the experimental results obtained on COLLATE documents and the comparison with the batch system Progol [10], that differently from INTHELEX performs learning in one step, i.e. it requires all the information needed for carrying out its task to be available when the learning process starts. Lastly, Section 4 draws some conclusions.

2 Incremental Learning with INTHELEX

Automatic revision of logic theories is a complex and computationally expensive task. Incremental learning is necessary when incomplete information is available at the time of initial theory generation, which is very frequent in real-world situations. Hence, the need for incremental models to complete and support the classical batch ones, that perform learning in one step and thus require the whole set of observations to be available since the beginning. Such a consideration, among others on the incremental learning systems available in the literature led to the design and implementation of INTHELEX[1] (INcremental THEory Learner from EXamples) [3], whose most characterizing features are in its incremental nature, in the reduced need of a deep background knowledge, in the exploitation of negative information and in the peculiar bias on the generalization model, which reduces the search space and does not limit the expressive power of the adopted representation language.

2.1 The Inductive Core

INTHELEX is a learning system for the induction of *hierarchical* logic theories from examples: it is *fully incremental* (in addition to the possibility of refining a previously generated version of the theory, learning can also start from an empty theory); it is based on the *Object Identity* assumption (terms, even variables, denoted by different names within a formula must refer to different objects) and learns theories expressed as sets of Datalog^{OI} clauses [11] from positive and negative examples; it can learn simultaneously *multiple concepts*, possibly related to each other (recursion is not allowed); it retains all the processed examples, so to guarantee validity of the learned theories on all of them; it is a *closed loop* learning system (i.e., a system in which feedback on performance is used to activate the theory revision phase [1]).

[1] It is currently available in binary format for i586 DOS-based platforms (`http://lacam.di.uniba.it:8000/systems/inthelex/`)

The learning cycle performed by INTHELEX can be described 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 three subsets, namely 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 stored in the base of processed examples, then they are exploited to obtain a theory that is able to explain them. Such an initial theory can also be provided by the expert, or even be empty. Subsequently, the validity of the theory against new available examples, also stored 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. Specifically, INTHELEX incorporates two inductive refinement operators to revise the theory, one for generalizing hypotheses that reject positive examples, and the other for specializing hypotheses that explain negative examples. 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.

Whenever a new example is taken into account, it is stored in the historical memory of all past examined examples and the current theory is checked against it. If it is positive and not covered, generalization must be performed. One of the clauses defining the concept the example refers to is chosen by the system for generalization. The set of generalizations of this clause and the example is computed, by taking into account a number of parameters that restrict the search space according to the degree of generalization to be obtained and the computational budget allowed. If one of such generalizations is consistent with all the past negative examples, then it replaces the chosen clause in the theory, or else a new clause is chosen to compute generalization. If no clause can be generalized in a consistent way, the system checks if the example itself, with the constants properly turned into variables, is consistent with the past negative examples. If so, such a clause is added to the theory, or else the example itself is added as an exception.

If the example is negative and covered, specialization is needed. Among the theory clauses occurring in the derivation of the example, INTHELEX tries to specialize one at the lowest possible level in the dependency graph by adding to it one (or more) positive literal(s), which characterize all the past positive examples and can discriminate them from the current negative one. Again, parameters that bound the search for the set of literals to be added are considered. In case of failure on all of the clauses in the derivation, the system tries to add the negation of a literal, that is able to discriminate the negative example from all the past positive ones, to the clause related to the concept the example is an instance of. If this fails too, the negative example is added to the theory as an exception. New incoming observations are always checked against the exceptions before applying the rules that define the concept they refer to.

2.2 Multistrategy Learning

Another peculiarity in INTHELEX is the integration of multistrategy operators that may help in the solution of the theory revision problem by pre-processing the incoming information, according to the theoretical framework for integrating different learning strategies known as Inferential Learning Theory [9]. Namely, deduction is exploited to fill observations with information that is not explicitly stated, but is implicit in their description, and hence refers to the possibility of better representing the examples and, consequently, the inferred theories. Conversely, abduction aims at completing possibly partial information in the examples (adding more details), whereas abstraction removes superfluous details from the description of both the examples and the theory. Thus, even if with opposite perspectives, both aim at reducing the computational effort required to learn a correct theory with respect to the incoming examples.

INTHELEX requires the observations to be expressed only in terms of the set of predicates that make up the description language for the given learning problem. To ensure uniformity of the example descriptions, such predicates have no definition. Nevertheless, since the system is able to handle a hierarchy of concepts, combinations of these predicates might identify higher level concepts that is worth adding to the descriptions in order to raise their semantic level. For this reason, INTHELEX implements a saturation operator that exploits deduction to recognize such concepts and explicitly add them to the examples description. The system can be provided with a Background Knowledge, supposed to be correct and hence not modifiable, containing (complete or partial) definitions in the same format as the theory rules. This way, any time a new example is considered, a preliminary saturation phase can be performed, that adds the higher level concepts whose presence can be deduced from such rules by subsumption and/or resolution. In particular, the generalization model of implication under Object Identity is exploited [5]. Differently from abstraction (see next), all the specific information used by saturation is left in the example description. Hence, it is preserved in the learning process until other evidence reveals it is not significant for the concept definition, which is a more cautious behaviour.

Abduction was defined by Peirce as hypothesizing some facts that, together with a given theory, could explain a given observation. According to the framework proposed in [7], an *abductive logic theory* is made up by a normal logic program [8], a set of *abducibles* and a set of *integrity constraints*. Abducibles are the predicates about which assumptions (*abductions*) can be made: They carry all the incompleteness of the domain (if it were possible to complete these predicates then the theory would be correctly described). Integrity constraints (each corresponding to a combination of literals that is not allowed to occur) provide indirect information about them. The proof procedure implemented in INTHELEX corresponds, intuitively, to the standard Logic Programming derivation suitably extended in order to consider abducibles.

Abstraction is a pervasive activity in human perception and reasoning. When we are interested in the role it plays in Machine Learning, inductive inference must be taken into account as well. The exploitation of abstraction concerns the shift from the language in which the theory is described to a higher level one. According to the

framework proposed in [13], concept representation deals with entities belonging to three different levels. Concrete objects reside in the *world*, but any observer's access to it is mediated by his *perception* of it. To be available over time, these stimuli must be memorized in an organized *structure*, i.e. an *extensional* representation of the perceived world. Finally, to reason about the perceived world and communicate with other agents, a *language* is needed, that describes it *intensionally*. Abstraction takes place at the world-perception level by means of a set of operators, and then propagates to higher levels, where it is possible to identify operators corresponding to the previous ones. An abstraction theory expresses such operators, that allow the system to replace a number of components by a compound object, to decrease the granularity of a set of values, to ignore whole objects or just part of their features, and to neglect the number of occurrences of some kind of object. In INTHELEX the abstraction theory must be given, and the system automatically applies it to the learning problem at hand before processing the examples.

3 Experimental Results on the COLLATE Dataset

Supported by previous successful application to the paper document processing domain [12], the symbolic learning system INTHELEX has been applied to learn rules for the automatic identification of a wide range of significant COLLATE classes and their related components, to be used for indexing/retrieval purposes and to be submitted to the users for annotation. A challenge comes from the low layout quality and standard of the material, which causes a considerable amount of noise in its description (see Figure 1). The layout quality is often affected by manual annotations, stamps that overlap to sensible components, ink specks, etc. As to the layout standard, many documents are typewritten sheets, that consist of all equally spaced lines in Gothic type. Such a situation requires the system to be flexible in the absence of particular layout components due to the typist's style, and to be able to ignore layout details that are meaningless or superfluous to the identification of the interesting ones.

The dataset consisted of 29 documents for the class `faa_registration_card` (a certification that the film has been approved for exhibition in the present version by the censoring authority), 36 ones for the class `dif_censorship_decision` (decision whether a film could or could not, and in which version, be distributed and shown throughout a Country), 24 ones for the class `nfa_cen_dec_model_a` and 13 for the class `nfa_cen_dec_model_b` (both these classes represent a list of intertitles needed to check whether a film shown in the cinema was the same as the one examined by the censorship office). Other 17 reject documents were obtained from newspaper articles. Note that the symbolic method adopted allows the trainer to specifically select prototypical examples to be included in the learning set. This explains why theories with good predictiveness can be obtained even from fewer observations.

The first-order descriptions of such documents, needed to run the learning system, were automatically generated by the system WISDOM++ [4]. Starting from scanned images, such a system is able to identify the layout blocks that make up a paper document along with their type and relative position.

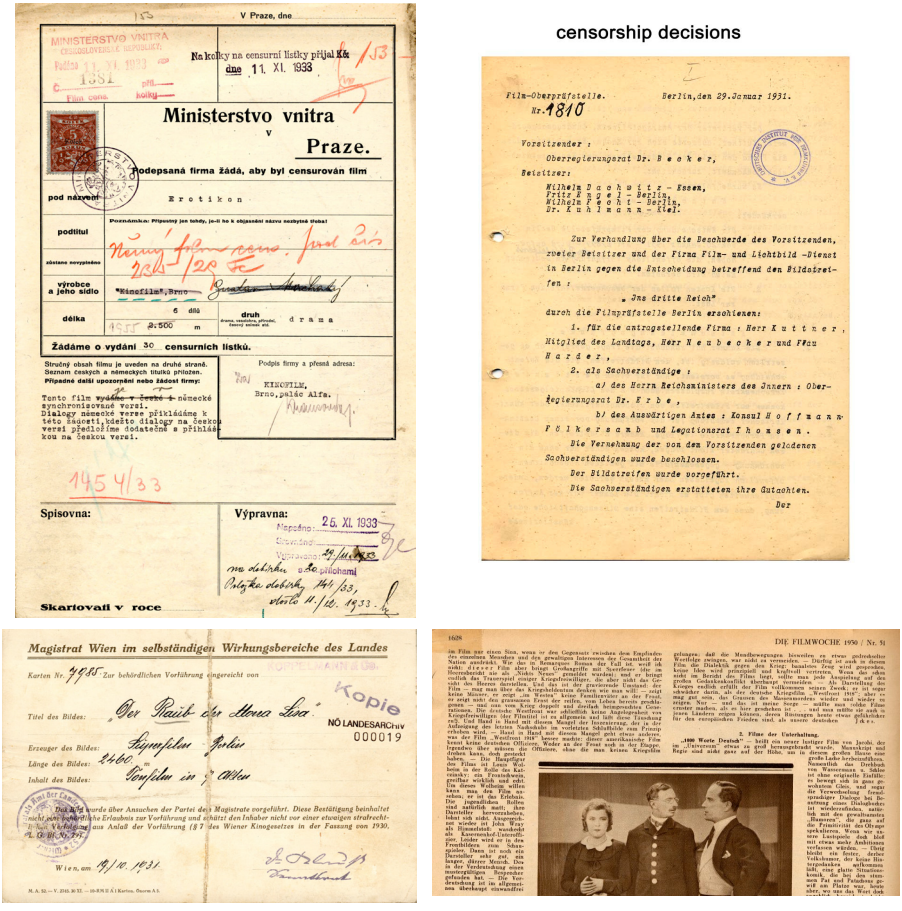

Fig. 1. Sample COLLATE documents

Each document was then described in terms of its composing layout blocks, along with their size (height and width), position (horizontal and vertical), type (text, line, picture and mixed) and relative position (horizontal/vertical alignment, adjacency).
The description length of the documents ranges between 40 and 379 literals (144 on average) for class `faa_registration_card`, between 54 and 263 (215 on average) for class `dif_censorship_decision`; between 105 and 585 (269 on average) for class `nfa_cen_dec_model_a` and between 191 and 384 (260 on average) for class `nfa_cen_dec_model_b`.

Each document was considered as a positive example for the class it belongs, and as a negative example for the other classes to be learned; reject documents were considered as negative examples for all classes. Definitions for each class were learned, starting from the empty theory and with all the negative examples at the beginning (in order to simulate a batch approach), and their predictive accuracy was tested according to a 10–fold cross validation methodology, ensuring that each fold contained the same proportion of positive and negative examples.

As regards the rules learned by INTHELEX, Figure 2 shows a definition for the classification of documents belonging to `dif_censorship_decision` class. It is interesting to note that it is straightforwardly understandable by humans. Specifically, the English translation of the concept expressed by this rule is "*a document belongs to this class if it has long length and short width, it contains three components in the upper-left part, all of type text and having very short height, two of which are medium large and one of these two is on top of the third* ". This feature was greatly appreciated by experts in charge of working with the processed documents, and was one of the goals we aimed to.

```
class_dif_cen_decision(A) :-
   image_lenght_long(A),image_width_short(A),
   part_of(A,B),
   width_medium_large(B),height_very_very_small(B),
   type_of_text(B),pos_left(B),pos_upper(B),
   part_of(A,C),
   height_very_very_small(C),type_of_text(C),
   pos_left(C),pos_upper(C),
   on_top(C,D),
   width_medium_large(D),height_very_very_small(D),
   type_of_text(D),pos_left(D),pos_upper(D).
```

Fig. 2. Examples of Learned Definitions

Two remarks are worth for this class: first, the features in the above description are common to all the learned definitions in the 10 folds, which explains why the performance of the system on this class of documents is the best of all; second, starting with descriptions whose average length was 215, the average number of literals in the learned rules is just 22.

Experiments were run not only with INTHELEX, but also with the state-of-the-art system Progol. The aim was checking that there is no loss in performance using the incremental technique instead of the batch one. This would allow to safely exploit the incremental approach in domains characterized by a continuous flow of new documents. Table 1 reports the statistics regarding the performance of the two exploited approaches, averaged on the 10 folds, of the classification process in this environment as regards number of clauses that define the concept `Clauses`, Accuracy on the test set (expressed in percentage) and Runtime (in seconds).

The difference in computational time between the two systems is noteworthy, confirming that the incremental approach should be more efficient than the batch one (since it has to just revise theories stepwise, instead of learning them from scratch). On the contrary, predictive accuracy seems very similar, which suggested to perform a statistical test for assessing its significance.

Thus, to better assess the goodness of INTHELEX, the performance of the system on these datasets in the above 10-fold cross validation was compared, according to a paired *t*-test [2], to that obtained by the Progol batch system. The aim was to evaluate the difference in effectiveness of the rules induced by the two systems according to the predictive accuracy metric. Table 2 reports the results of such a comparison, requiring a

significance level of $\alpha = 0.995$. In hypothesis testing, the significance level of a test is the probability of incorrectly rejecting the null hypothesis. In our experiment the null hypothesis is: "*the two systems are equally performing*". Here, we want some guarantee that the two systems are comparable, in order to apply the incremental one in real word domains, such as office automation. In other words, we want be sure with a high probability that there are no differences among the two systems. Thus, we have chosen an high significance, that assures the systems equality with a very small margin to make a mistake on evaluating their performance. The test revealed no statistically significant differences in predictive accuracy among the systems in the classification task in the cultural heritage material environment.

Table 1. Statistics for Document Classification

	Clauses		Runtime (sec.)		Accuracy (%)	
	Progol	INTHELEX	Progol	INTHELEX	Progol	INTHELEX
DIF	1.0	1.0	687.70	17.13	99.17	99.17
FAA	3.6	3.5	3191.98	334.05	95.83	94.17
NFA_A	4.6	2.8	1558.90	87.71	95.73	93.92
NFA_B	1.0	1.7	359.67	92.05	97.63	97.56

Table 2. INTHELEX – Progol Comparison

	Accuracy (%)		
	Progol	INTHELEX	t-value
DIF	99.17	99.17	0
FAA	95.83	94.17	0.80
Mod_A	95.73	93.92	0.57
Mod_B	97.63	97.56	0.03

The experimental results show that INTHELEX is able to learn theories with performance comparable to the batch systems ones. This seems interesting, since the incremental setting implies having, at any moment, only a limited vision of the domain. Conversely, batch systems such as Progol can consider the whole knowledge available since the beginning of the learning process, so having a general vision of the knowledge available. Thus, such a feature should in principle result in a better predictive accuracy of theories learned by the latter with respect to those provided by the former.

4 Conclusion and Future Work

This paper presented experimental results proving the benefits that the addition of the incremental learning system INTHELEX in the architecture of the EU project COLLATE can bring, in order to learn rules for automatic classification and interpretation of cultural heritage documents. The domain is particularly challenging because of the low layout quality and standard of the material, which can represent a good testbed in the perspective of applying the same techniques to the field of office automation, where the continuous flow of new documents makes incrementality a necessary feature. INTHELEX works on symbolic (first-order logic) representations, that proved very powerful in describing a complex environment such as the COLLATE

one. This was confirmed by a comparison with the batch learning system Progol on the predictive accuracy metric, revealing that there are generally no statistically significant differences among the two systems.

The presented experiments were carried out by exploiting only pure induction. Nevertheless, the multistrategy features provided by INTHELEX could probably further improve the performance. Thus, future work will concern performing new experiments aimed at comparing multistrategy learning with respect to the baseline results presented in this paper.

Acknowledgement. This work was partially funded by the EU project IST-1999-20882 COLLATE "Collaboratory for Annotation, Indexing and Retrieval of Digitized Historical Archive Material".

References

1. J. M. Becker. Inductive learning of decision rules with exceptions: Methodology and experimentation. B.s. diss., Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1985.
2. Thomas G. Dietterich. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895--1923, 1998.
3. F. Esposito, G. Semeraro, N. Fanizzi and S. Ferilli. Multistrategy Theory Revision: Induction and abduction in INTHELEX. *Machine Learning Journal*, 38(1/2):133--156, 2000.
4. F. Esposito, D. Malerba and F.A. Lisi. Machine learning for intelligent processing of printed documents. *Journal of Intelligent Information Systems*, 14(2/3):175--198, 2000.
5. F. Esposito, N. Fanizzi, S. Ferilli and G. Semeraro. Refining logic theories under oi-implication. In Z. W. Ras and S. Ohsuga, editors, *Foundations of Intelligent Systems*, No. 1932 in Lecture Notes in Artificial Intelligence, pages 109--118. Springer-Verlag, 2000.
6. R.T. Kouzes, J.D. Myers, and W.A. Wulf. Collaboratories: Doing science on the internet. *IEEE Computer*, 29(8), 1996.
7. E. Lamma, P. Mello, F. Riguzzi, F. Esposito, S. Ferilli and G. Semeraro. Cooperation of abduction and induction in logic programming. In A. C. Kakas and P. Flach, editors, *Abductive and Inductive Reasoning: Essays on their Relation and Integration*. Kluwer, 2000.
8. J. W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, Berlin, second edition, 1987.
9. R.S. Michalski. Inferential theory of learning. developing foundations for multistrategy learning. In R.S. Michalski and G. Tecuci, editors, *Machine Learning. A Multistrategy Approach*, volume IV, pages 3--61. Morgan Kaufmann, San Mateo, CA, 1994.
10. S. Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245--286, 1995.
11. G. Semeraro, F. Esposito, D. Malerba, N. Fanizzi and S. Ferilli. A logic framework for the incremental inductive synthesis of Datalog theories. In N.E. Fuchs, editor, *Proceedings of 7th International Workshop on Logic Program Synthesis and Transformation* LOPSTR97, volume 1463 of *LNCS*, pages 300--321. Springer, 1998.
12. 12.G. Semeraro, N. Fanizzi, S. Ferilli and F. Esposito. Document classification and interpretation through the inference of logic-based models. In P. Constantopoulos and I.T. Sølvberg, editors, *Research and Advanced Technology for Digital Libraries*, number 2163 in Lecture Notes in Computer Science, pages 59--70. Springer-Verlag, 2001.
13. J. D. Zucker. Semantic abstraction for concept representation and learning. In R. S. Michalski and L. Saitta, editors, *Proceedings of the 4th International Workshop on Multistrategy Learning*, Desenzano del Garda, Italy, 1998.