Incremental Induction of Rules for Document Image Understanding

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

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

Abstract. This paper aims at presenting the application of first-order logic machine learning techniques to two document domains in order to learn rules for recognizing the semantic role of their logical components. Specifically, the multistrategy incremental learning system INTHELEX has been applied to multi-format scientific papers and documents concerning European films from the 20's and 30's. The challenge comes from the different levels of formatting standards in these domains: from (more or less) standardized layouts, in scientific papers, to documents with almost no standard, in historical cultural heritage material. Experimental results in both domains and a comparison with the Progol system assess the advantages that the exploitation of INTHELEX can yield.

1 Introduction

One of the key issues regarding the use of information systems is the acquisition of new information, which often resides in paper documents. Automatically acquiring new information could be useful to populate knowledge bases to be exploited in support of human activities in many tasks. This holds, in particular, when dealing with important historic and cultural sources, which are fragile and distributed in various archives, and must be digitized in order to preserve them and make them available to a wider public. If such document bases are to be exploited by human experts, which happens for instance in the cultural heritage domain, the need of having learned rules expressed in such a way that they can be easily understood and checked is particularly strong.

In this context, automatic induction of rules that are able to recognize the semantic role of the significant layout components of documents in order to provide them to the domain experts would be very helpful in order to make the extraction of important information easier. In particular, the complexity of some document layout structures suggests the use of symbolic (first-order logic) descriptions and techniques. This was the motivation that led us to try applying the learning system INTHELEX in this environment. Incremental learning is necessary when incomplete information is available at the time of initial theory generation, which is very frequent in real-world situations like those described above, where a continuous flow of new documents requires a constant check

A. Cappelli and F. Turini (Eds.): AI*IA 2003, LNAI 2829, pp. 176–188, 2003.

(and possible revisions) of the learned theories. Hence, the need for incremental algorithms to complete and support the batch ones, that perform learning in one step and thus require the whole set of observations to be available from the beginning. The corpus of documents on which we worked concerns two domains: multi-format scientific papers and rare historic film censorship documents from the 20's and 30's. All material is analyzed, indexed, annotated and interlinked by domain experts, to which it could be useful to provide suitable knowledge management tools to support their work.

The following section presents the incremental learning system along with its reasoning strategies; then, Section 3 reports the experimental results obtained on the scientific papers and cultural heritage collections and the comparison of INTHELEX with Progol [8]. Lastly, Section 4 draws some conclusions.

2 Incremental Learning with INTHELEX

Automatic revision of logic theories is a complex and computationally expensive task. Considerations on the incremental learning systems available in the literature [4] led to the design and implementation of INTHELEX (INcremental THEory Learner from EXamples), whose most characterizing features are its incremental nature, the reduced need of a deep background knowledge, the exploitation of negative information and 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 [4]: 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 [10] 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, in order 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]).

INTHELEX incorporates two inductive refinement operators, one for generalizing hypotheses that reject positive examples (upward), and the other for specializing hypotheses that explain negative examples (downward). It exploits a (possibly empty) previous theory, a graph describing the dependence relationships among concepts (if any), and a historical memory of all the past examples that led to the current theory. Whenever a new example is taken into account, it is stored in such a repository and the current theory is checked against it.

The generalization process is started if a positive example is not explained. Starting from the current theory, the misclassified example and the past negative examples, it ends with a revised theory. Since there is no single clause responsible for not explaining the given example, the system must firstly choose a clause to be generalized among those making up the involved concept description. Then, the system tries to compute one of the lgg_{OI}'s of this clause and the example. The upward refinement operator extends the concept of *least general generalization* (*lgg*) introduced by Plotkin [9] to cope with the ordering induced by θ_{OI}-subsumption. Note that the *lgg* is no longer unique under θ_{OI}-subsumption, since the space of Datalog clauses is not a lattice when ordered by θ_{OI}-subsumption [10], while it is when ordered by θ-subsumption. In our framework, the lgg_{OI} is initialized as the literal generalizing the two heads; then, literals in Plotkin's *lgg* are progressively added to the partial generalization as long as they share at least one variable with it and fulfill the OI requirement. When a generalization is found, it is tested with respect to all the past negative examples. If none of them is explained, then the generalized clause is replaced in the theory by the generalization. Otherwise, an intelligent backtracking on the literals in Plotkin's *lgg* is performed, which yields another generalization (if any). If no clause in an incomplete definition can be generalized so that the resulting theory is complete and consistent, the system checks if the example itself, with the constants properly turned into variables, is consistent with the past negative examples. Such a clause is added to the theory, or else the example itself is added as an exception to the theory.

As to specialization, in our framework it consists of adding proper literals to an inconsistent clause, in order to avoid it explaining a negative example. Revisions performed by our operator are always minimal [11]: a specialization obtained by turning a variable into a constant is not provided since all clauses in the theory contain only variables as arguments. Starting from the current theory, the misclassified example and the past positive examples, the specialization algorithm yields a revised theory by adding proper literals to the inconsistent clauses. Since the space in which the literals to be added should be searched for is potentially infinite, the operator has to focus the search into the portion of the space that contains the solution of the diagnosed error. The search is firstly performed in the space of positive literals, that contains information coming from all the past positive examples, but not yet exploited by it. If the search in this space fails, the algorithm autonomously performs a *representation change*, extending the search to the space of negative literals, built by considering the negative example that caused the error [10]. First of all, the process detects all the clauses occurring in the SLD-derivation of the example. Then, the system tries to specialize one at the lowest possible level (which corresponds to a clause defining a concept whose level is the lowest in the dependency graph), in order to refine more "basic" concepts first. Since the downward refinements we are looking for must satisfy the property of maximal generality, this suggests to search for one (or more) positive literal(s) which can discriminate all the past positive examples from the current negative one. Specifically, if there exists one

(or a combination of) literal(s) that, when added to the body of the clause to be specialized, is able to discriminate the negative example that caused the inconsistency, then the downward refinement operator should be able to find it. If the derivation of an example in a theory is not unique, a single specialization step could be insufficient to restore the consistency of the theory. For such a reason, when a specialization of a clause is reached, the example is tested again to check if another SLD-derivation, different from the previous one, exists for it. If so, the process is iterated until no more derivations of the example are found. Note that only one step of specialization is needed whenever the SLD-derivation of the example involves just one clause. If no clause in the SLD-derivation of the example can be specialized by adding literals coming from the positive examples, an attempt is made to add a negative literal to the first clause of the SLD-derivation (the one related to the concept the example is an instance of). If none of the clauses obtained makes the theory complete and consistent again, then the system adds the negative example to the theory as an exception.

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 [7]. Namely, deduction is exploited to fill observations with information that is not explicitly stated, but is implicit in their description. 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, the system implements a saturation operator that exploits deduction to recognize such concepts and explicitly add them to the examples description. 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. Note that all the specific information used by saturation is left in the example description and it is preserved in the learning process until other evidence reveals it is not significant for the concept definition.

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 [5], an *abductive logic theory* is made up by a normal logic

program [6], 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 the system starts from a goal and a set of initial assumptions and results in a set of consistent hypotheses (abduced literals) by intertwining *abductive* and *consistency derivations*. Intuitively, an abductive derivation is the standard Logic Programming derivation suitably extended in order to consider abducibles [4].

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 [12], 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*, 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 them, and to neglect the number of occurrences of some kind of object. The abstraction theory must be provided to the system that automatically applies it to the learning problem at hand before processing the examples.

3 Experimental Results

INTHELEX has been exploited to learn rules for the automatic identification of logical components in two documents domains which are characterized by different (if any at all) formatting style in their layout structure. The two datasets consist of scientific papers belonging to three different series (*ICML*, *ISMIS*, *IEEET*) and of a large collection of documents concerning film censorship, used in the European Project COLLATE, coming from three historical European film archives (*FAA*, *DIF*, *NFA*). In the former, each series is characterized by its own layout standard, but fulfilled more or less strictly in the three cases. In the latter, a challenge comes from the low layout quality and different formatting standard of the material, which causes a considerable amount of noise in its description. In this environment, for example, abstraction can be exploited for ignoring superfluous specks on the layout, while abduction could be useful to hypothesize the presence of expected but unseen layout components.

The first-order descriptions of such documents, needed to run INTHELEX, were automatically generated by the system WISDOM++ [3] (note that the scientific papers dataset is the same as in [3], but the set of learned layout compo-

```
width_very_small(X) :- width(X,Y), Y >= 0, Y =< 15.
width_small(X) :- width(X,Y), Y > 15, Y =< 30.
width_medium_small(X) :- width(X,Y), Y > 30, Y =< 80.
width_medium(X) :- width(X,Y), Y > 80, Y =< 130.
width_medium_large(X) :- width(X,Y), Y > 130, Y =< 250.
width_large(X) :- width(X,Y), Y > 250, Y =< 400.
width_very_large(X) :- width(X,Y), Y > 400, Y =< 640.
```

Fig. 1. Abstraction rules for width of block

nents for each of the classes is different). Starting from scanned images, it is able to identify the layout blocks that make up a paper document and to describe them in terms of their size (height and width, in pixels), position (horizontal and vertical, in pixels from the top-left corner), type (text, line, picture and mixed) and relative position (horizontal/vertical alignment, adjacency). Since the inductive procedure embedded in INTHELEX is not able to handle numeric values (such as the number of pixels in the document descriptions provided by WISDOM++), a change of representation in the description language was necessary, such that final observations were made up of symbolic attributes only. The abstraction operator was used for breaking numeric values into intervals represented by symbolic constants, by providing the system with an abstraction theory containing rules that encode such a language shift. Figure 1 shows the rules of the abstraction theory that are in charge of discretizing the width of a layout block in a document. All the experiments described in the following were performed according a 10-fold cross validation methodology. For each class of documents in these datasets, the layout blocks that are semantically significant for indexing/retrieval purposes were identified and annotated by expert users, and subsequently used as examples to learn rules for automatically recognize them when new documents become available. Note that different document classes have different labels, as reported in the following (in square brackets the corresponding number of positive and negative instances are reported).

Let us start from the scientific papers dataset. The semantic labels of interest recognized by the domain experts for class *ICML* were: *abstract* [28+,340-], *author* [36+, 332-], *page_number* [27+, 341-] and *title* [29+, 339-]. As regards class *ISMIS* the following labels characterizing the objects belonging to it were provided: *abstract* [32+, 250-], *affiliation* [30+, 252-], *author* [30+, 252-] and *title* [30+, 252-]. Finally, for class *IEEET* we had: *abstract* [36+, 576-], *affiliation* [31+, 581-], *author* [34+, 578-], *index_term* [32+, 580-], *title* [40+, 572-], *running_head* [40+, 572-] and *page_number* [33+, 579-]. Each positive example for a label class to be learned was considered as negative for the others. Furthermore, any document block not labelled by the expert as significant was considered negative for all the components to be learned. Table 1, discussed at the end of this section, reports the experimental results averaged on the 10 folds as regards number of clauses defining the concept (Cl), number of performed generalizations (Lgg), *Runtime* (in seconds) and Predictive Accuracy.

Table 1. Statistics for understanding in scientific papers domain

	Clauses	Lgg's	Runtime	Accuracy %		
ICML				INTHELEX	PROGOL	t VALUE
abstract	2.00	7.70	66.61	98.64	99.45	1.96
author	1.10	5.50	47.77	99.73	98.09	-2.73
page_number	2.70	6.40	233.65	98.65	97.26	-1.48
title	3.30	12.90	515.14	98.08	97.79	-0.38
ISMIS						
abstract	4.50	12.30	92.31	96.44	95.02	-0.87
affiliation	5.30	13.30	515.61	94.65	91.81	-1.30
author	4.70	12.80	857.26	95.02	91.13	-1.99
title	3.80	11.40	329.41	95.75	96.84	0.78
IEEET						
abstract	5.90	18.20	594.63	96.89	97.87	1.62
affiliation	3.60	10.40	331.38	98.53	98.85	0.68
author	2.20	10.50	195.28	99.18	98.36	-1.85
index_term	4.20	13.90	892.92	97.88	98.53	1.18
page_number	3.80	13.90	647.71	97.38	99.18	3.16
running_head	5.10	15.90	766.62	95.91	97.87	2.71
title	3.50	12.70	474.93	97.39	95.42	-2.35

Figure 2 shows the definitions learned by INTHELEX for the target concepts of the *ICML* set of documents in one of the 10 folds. As shown in Figure 3, it is possible to exactly recognize and map on a sample document the layout blocks referred to in the rules. For instance, the rule describing the label *abstract* says that a block A is an abstract of a document B belonging to *ICML* collection if its width is medium-large and it is placed on the left (w.r.t. the horizontal position) middle (w.r.t. the vertical position) part of the document. This definition also refers to other three objects, C, D and E, one of which, C, has smaller size than the block A and is placed above A. The domain expert recognized block C as that containing the title (word) "Abstract" in a paper. Note that starting from document descriptions whose length ranges between 100 and 180 literals, the learned rules, shown in Figure 2, contain at most 15 literals.

In the COLLATE domain, the document 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. 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.

```
logic_type_abstract(A) :-
   part_of(B,A),width_medium_large(A),pos_left(A),pos_middle(A),
   part_of(B,C), part_of(B,D),part_of(B,E),on_top(C,A),
   width_medium_small(C),height_very_very_small(C),type_of_text(C).
logic_type_author(A) :-
   part_of(B,A),type_of_text(A),pos_upper(A),
   part_of(B,C),height_very_very_small(C),type_of_text(C),pos_upper(C),
   part_of(B,D),on_top(D,A),on_top(E,D),pos_center(E),pos_upper(E),
   width_very_large(D),height_smallest(D),pos_center(D),pos_upper(D),
logic_type_page_number(A) :-
   part_of(B,A),pos_upper(A),part_of(B,C),
   width_very_large(C),height_smallest(C),pos_center(C),pos_upper(C),
   on_top(A,E),width_very_large(E),pos_center(E),pos_upper(E),
   on_top(D,C),type_of_text(D),pos_center(D),pos_upper(D).
logic_type_title(A) :-
   part_of(B,A),type_of_text(A),pos_center(A),pos_upper(A),
   part_of(B,C),height_very_very_small(C),type_of_text(C),
   on_top(C,D),on_top(A,E),part_of(B,E),
   width_very_large(E),height_smallest(E),pos_center(E),pos_upper(E).
```

Fig. 2. Learned definition for semantic components of ICML

As for the previous dataset, each document in this domain has different semantic components according the class it belongs to. Thus, in the documents coming from the *DIF* archive, which are censorship decisions, the domain experts recognized the following characterizing labels: *cens_signature* [35+], *cert_signature* [35+], *object_title* [36+], *cens_authority* [36+], *chairman* [36+], *assessors* [36+], *session_data* [36+], *representative* [49+]. Definitions for each class were learned, starting from the empty theory and considering as negative examples for each label those that were positive for the other components. Table 2 reports the experimental results, averaged on the 10 folds, of the interpretation process in this environment.

Table 2. Statistics for understanding of DIF documents

	Clauses	Lgg's	Runtime	Accuracy %		
				INTHELEX	PROGOL	t VALUE
cens_signature	2.2	11.6	1459.883	98.32	99.33	1.41
cert_signature	2.2	7.6	176.592	98.31	99.32	1.96
object_title	5	15.2	3960.829	94.66	94.67	0.01
cens_authority	2.9	12.1	2519.45	97.64	98.02	0.30
chairman	4.6	13.8	9332.845	93.10	91.97	-0.92
assessors	4.6	15	12170.93	94.48	98.52	**4.71**
session_data	2.5	8.6	1037.96	97.68	98.98	1.76
representative	5.6	20.7	13761.958	92.98	96.98	2.88

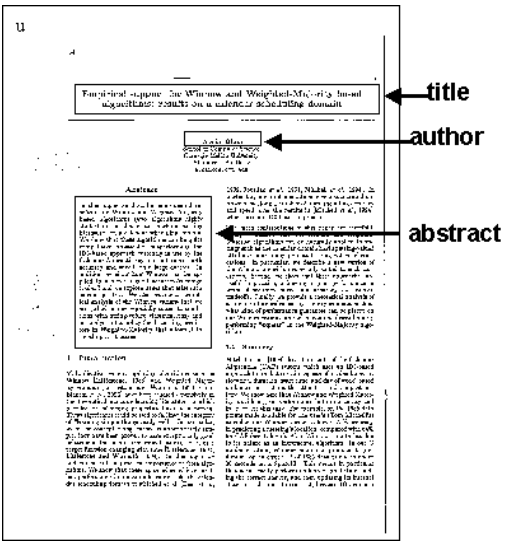

Fig. 3. Sample ICML document

As regards the class of documents coming from the *FAA* archive, the provided labels were: *registration_au* [28+], *date_place* [26+], *department* [17+], *applicant* [11+], *reg_number* [28+] , *film_genre* [20+], *film_length* [19+], *film_producer* [18+], *film_title* [20+]. Averaged results are reported in Table 3. Figure 4 shows the definition learned by INTHELEX for the concept *film_title* of the registration cards from *FAA* in one of the folds. Such a rule describes the identified blocks by means of their size and relative position w.r.t. the whole document and the other blocks. Figure 5 graphically highlights the components of the learned description in a sample of document. Also in this case, as for the scientific papers,

Table 3. Statistics for understanding of FAA documents

	Clauses	Lgg's	Runtime	Accuracy %		
				INTHELEX	PROGOL	t VALUE
registration_au	5.6	12.5	3739.366	91.43	89.35	-1.28
date_place	6.9	13.5	7239.625	86.69	89.80	1.94
department	1.9	6.6	118.625	98.95	100	1.00
applicant	2	4.5	93.993	97.89	98.91	1.49
reg_number	5.1	14.4	4578.208	91.95	97.30	**4.82**
film_genre	4	8.4	2344.899	93.02	94.18	0.38
film_length	5.5	9.9	3855.391	90.87	-	-
film_producer	4.9	10.4	4717.17	94.03	-	-
film_title	5.4	11.1	4863.084	89.82	-	-

```
logic_type_film_title(A) :-
   page_first(B),part_of(B,A),part_of(B,C),part_of(B,D),
   type_of_text(A),pos_upper(A),type_of_text(D),pos_upper(D),
   height_very_small(C),type_of_text(C),pos_center(C),pos_upper(C),
   width_very_large(E),height_smallest(E),
   type_of_hor_line(E),pos_center(E),pos_upper(E),
   height_very_very_small(F),pos_left(F),pos_upper(F),
   on_top(E,D),on_top(E,F),alignment_left_col(F,A).
```

Fig. 4. Learned definition for a semantic component of registration cards in FAA

starting from document descriptions whose length ranges between 40 and 379 literals, the learned rule, shown in Figure 4, contains 23 literals.

Finally, documents belonging to *NFA* archive were characterized by the following labels, almost all different from the previous ones: *dispatch_office* [33+], *applic_notes* [18+], *no_censor_card* [21+], *film_producer* [20+], *no_prec_doc* [20+], *applicant* [22+], *film_genre* [17+], *registration_au* [25+], *cens_process* [30+], *delivery_date* [16+] and *cens_card* [26+]. Table 4 shows the averaged results.

The overall outcomes reveal that INTHELEX was actually able to learn significant definitions for the layout blocks of interest in both the more standard-formatted documents and the low-quality ones. Indeed, the predictive accuracy is always very high in both cases, reaching even 99.73% in the case of scientific documents (in which never falls below 94.65%) and 98.95% in the COLLATE dataset (in which only 2 cases out of 28 fall below 90%, with percentages 86.69% and 89.82%). As shown, the learned rules have a high degree of understandability for human experts, which was one of the requirements for the experiment. As expected, the interpretation problem, that is concerned with the semantics of the layout blocks inside documents, turned out to be easier in the standard-formatted documents (scientific papers) than in the historical material. This is suggested by the tendential increase in runtime from Table 1 to Tables 2, 3 and 4. Such an increase is in some cases particularly evident, but it should be considered

Fig. 5. Components in the learned definition of a FAA registration card B

Table 4. Statistics for understanding of NFA documents

	Clauses	Lgg's	Runtime	Accuracy %		
				INTHELEX	PROGOL	t VALUE
dispatch_office	6.8	13.9	13149.31	94.28	94.86	0.55
applic_notes	2.5	5.7	231.05	98.81	100	1.82
no_censor_card	5.3	11.2	8136.796	95.48	96.38	0.89
film_producer	4.9	9.6	5303.78	93.98	97.29	2.29
no_prec_doc	4.6	11	5561.14	93.97	94.59	1.02
applicant	6.7	11.5	15588.15	93.66	93.38	-0.26
film_genre	2.8	6.9	684.35	98.53	99.41	1.96
registration_au	4.1	12.5	5159.74	94.64	97.31	2.23
cens_process	4.8	10.8	4027.90	98.51	-	-
cens_card	5.6	11.8	3363.86	94.61	95.23	0.72
delivery_date	4	9.1	3827.34	95.51	98.82	**4.31**

that the high predictive accuracy should ensure that few theory revisions can be expected when processing further documents.

The performance of the system on these datasets were compared to that obtained by the Progol batch system. For pairwise comparison a 10-fold cross validation paired t-test was used [2] in order to evaluate the difference in effectiveness of the rules induced by the two systems according to the predictive accuracy metric. Requiring a significance level of $\alpha = 0.995$, the test revealed no statistically significant differences in predictive accuracy among them in the scientific papers domain, while there are in the COLLATE dataset in favor of Progol for the following layout components: *assessors* for documents belonging to DIF dataset (Table 2), *reg_number* for FAA (Table 3) and *delivery_date* for NFA (Table 4). Such a result was expected in this domain because of the noise affecting historical material, given the approaches followed by the two systems. Indeed, batch systems may take into account the whole knowledge available since the beginning of the learning process, so having a general vision of the knowledge available, while incremental systems must work having, at any moment, only a limited vision of the domain.

However, it is worth noting that Progol was not able to learn some layout components at all. This happened for *film_length*, *film_title* and *film_producer* from FAA (where Progol learned theories made up of only positive exceptions, thus lacking any predicive accuracy) and *cens_process* of NFA (where after two days the process was stopped because no theory had been induced yet, for none of the folds). Such a difficulty in learning some components for Progol is confirmed by the presence of many positive exceptions also in the other learned theories (in some cases an average of 16 exceptions on 27 examples of the target concept are present). The failure in learning a theory, even for a single concept, is unacceptable in real-world applications like the one under consideration. Thus, such a comparison turns out encouraging on the possibility of exploiting INTHELEX in difficult domains, not only when the observations are provided incrementally.

4 Conclusions and Future Work

This paper presented experimental results proving that the incremental learning system INTHELEX can be successfully exploited to learn rules for automatic interpretation of scientific and cultural heritage documents. 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. A few situations are in favor of Progol, but its complete failure in learning some concepts is unacceptable in real-world applications. Thus, future work will concern deeper exploitation of the INTHELEX features aimed at improving its performance.

Acknowledgements

This work was partially funded by the EU project IST-1999-20882 COLLATE.

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. UIUCDCS-F-85-945. 177

[2] Thomas G. Dietterich. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998. 186

[3] 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. 180

[4] 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. 177, 180

[5] 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. 179

[6] J. W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, Berlin, second edition, 1987. 180

[7] 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, U. S. A., 1994. 179

[8] S. Muggleton. Inverse entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995. 177

[9] G. D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970. 178

[10] 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, *Logic Program Synthesis and Transformation*, number 1463 in Lecture Notes in Computer Science, pages 300–321. Springer-Verlag, 1998. 177, 178

[11] S. Wrobel. *Concept Formation and Knowledge Revision*. Kluwer Academic Publishers, 1994. 178

[12] 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. 180