Incremental Multistrategy Learning for Document Processing

Floriana Esposito, Stefano Ferilli, Nicola Fanizzi, Teresa M. A. Basile and Nicola Di Mauro
Dipartimento di Informatica, Università di Bari
Via E. Orabona, 4 – I-70125 Bari, Italia
{esposito, ferilli, fanizzi, basile, nicodimauro}@di.uniba.it

This work presents the application of a multistrategy approach to some document processing tasks. The application is implemented in an enhanced version of the incremental learning system INTHELEX. This learning module has been embedded as a learning component in the system architecture of the EU project COLLATE, which deals with the annotation of cultural heritage documents. Indeed, the complex shape of the material handled in the project has suggested that the addition of multistrategy capabilities is needed to improve effectiveness and efficiency of the learning process. Results proving the benefits of these strategies in specific classification tasks are reported in the experimentation presented in this work.

Numerous valuable historic and cultural sources – a major part of our cultural heritage – are scattered in various national archives. Thus, full knowledge and usage of this material are severely impeded by access problems, due to sources that are difficult-to-use or electronically unavailable and the lack of appropriate content-based search and retrieval aids that help users to find what they really need. Moreover, many informal and non-institutional contacts between cultural archives constitute specific professional communities which today, however, still lack effective and efficient technological support for cooperative and collaborative knowledge working. An answer to these problems might lie in the creation of digital libraries, enhanced by the concept of an annotation collaboratory, that are able to bundle documents, interpretation knowledge, work processes and an expert network in a very flexible working environment. This could be a very useful service for archiving in cultural institutions.

Arts and Humanities are areas that are mainly based on the interpretation of cultural objects such as texts, paintings, works of arts, or historical/ethnological remains and monuments. Such objects are often unique, very valuable, fragile, irreplaceable and locally preserved in scientific collections at museums, in archives, or in urban and historic areas. Archives, museums and other cultural institutions do not simply conserve these objects, they also manage large documentations on them in the form of photo collections, expertises, records, scientific studies and analyses. Both the objects themselves as well as the supplementary documentation are often accessible only through direct contact with the users. Duplicates such as text documents (e.g., critical editions), or image documents (facsimiles, photographs) on paper are extremely expensive in terms of man-power, know-how and printing costs, and often cannot be justified for a small scientific audience. Electronic formats for object documentation in digital libraries might alleviate this access problem.

Object and document collections in the Arts and Humanities always represent work in progress. The inventory at cultural institutions is growing steadily due to donations, acquisitions and to their own daily scientific and conservation services. These additions must be incorporated into the existing collections, but there are often problems of space, scientific know-how and lack of personnel. Professionals and experts classify, analyze, assess and expose or edit these objects and documents. Highly qualified external specialists are frequently difficult to locate, if they are not part of an academic network. Internal experts are often overburdened with routine work due to small cultural budgets and can only invest time sporadically in integrating new inventories. Many scientific members of cultural institutions have temporary contracts and leave after a few years, taking with them a great part of the accumulated know-how.

The intrinsic nature of the document processing procedures supporting the progressive work on historic material, as outlined in this introduction, poses several constraints that require solutions specifically tailored to the tasks mentioned above. Over the years Intelligent Systems have become valuable working instruments for researchers involved in humanistic areas. The new challenge is

now to provide these people with tools that are able to facilitate the fruition and investigation of the cultural heritage, so that even non-experts or communities of researchers may use up-to-date tools for both their personal work and for collaborative purposes. Technologically, the World Wide Web can serve both as a standard communication platform for such communities and as a gateway for document-centered digital library applications. Yet, while the Web may solve the problem of the diffusion and access of this material in its digital form, new automated tools are needed to allow more intelligent processing and more personalized utilization of this knowledge. According to the situation previously described, besides being effective and efficient, such automatic tools must be able to cope with situations in which the continual growth of the available material and knowledge is a fundamental and unavoidable issue. Hence, the need for a system component that is able to build incrementally upon previously acquired knowledge through diverse reasoning mechanisms. Specifically, the availability of systems that can automatically identify and separate document classes and meaningful parts within them would eliminate the need for experts to accomplish low-level tasks, thus allowing them to concentrate on more intellectual interpretation-intensive tasks. For such systems to be successful in a real operating environment, however, their behavior and results must be comprehensible to human experts, which can be achieved only when symbolic representations are used. The choice of these symbolic mechanisms, which closely resemble the human way of reasoning, also allows for a more direct understandability and control of the knowledge synthesized at every step of the process.
In the following we will present the problems raised by a particular cultural heritage application domain provided by the EU project COLLATE. Then, we will describe the incremental learning system, INTHELEX, along with experiments designed to test its multistrategy capabilities in such a domain. Lastly, after some comments on the experimental results, we will provide our conclusions.

A CULTURAL HERITAGE APPLICATION DOMAIN: THE COLLATE PROJECT

As already pointed out, many important historic and cultural sources, which constitute a major part of our cultural heritage, are fragile and distributed in various archives, which causes severe problems for full access, knowledge and usage. Moreover, many informal and non-institutional contacts between archives constitute specific professional communities, which today still lack effective and efficient technological support for cooperative and collaborative knowledge working. COLLATE is an EU project[1] that aims at developing a WWW-based *collaboratory* (Kouzes et al., 1996) for archives, researchers and end-users working with digitized historic/cultural material.

The documents concern European films of the early 20th century. Some typical documents are, for instance, the censorship cards, original scenarios or newspaper review articles, which refer to objects (the films themselves) that are typically not accessible in digitized formats; e.g., films might have been censored and the original, uncensored versions might be lost. The censorship cards referring to a single film may be different, depending on the different censorship processes to which the film has been submitted during the years, as well as on the different countries. The need is to acquire and manage all the documents referring to a unique subject in order to reconstruct an entity in the knowledge base available on the Web. Document image analysis tools are essential to support data entry from printed documents, which are of different nature, often on partially damaged supports, with different standard and ancient typing characters. A straightforward application of OCR technology produces poor results because of the variability of the layout structure of printed documents.

A more advanced solution would be to develop intelligent document processing tools that automatically transform a large variety of printed multi-page documents into a web-accessible form

[1] IST-1999-20882 project COLLATE - Collaboratory for Annotation, Indexing and Retrieval of Digitized Historical Archive Material (URL: http://www.collate.de).

such as XML. This transformation requires a solution to several image processing problems, such as the separation of textual and graphical components in a document image (*document analysis*), the recognition of the kind of document (*document classification*), the identification of semantically relevant components of the page layout (*document understanding*), the transformation of portions of the document image into sequences of characters (OCR), and the transformation of the page into HTML/XML format. A large amount of knowledge is required to effectively solve these problems.

In this paper we will focus on the task of document classification by using a multistrategy Machine Learning approach.

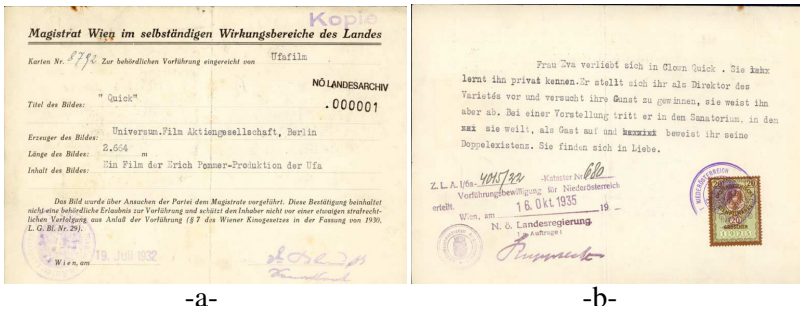
The documents are provided by three major European national film archives (Deutsches Filminstitut – DIF –, Filmarchiv Austria – FAA – and Národní Filmový Archiv – NFA), and include a large corpus of rare historic film censorship documents dating from the twenties and thirties and also newspaper articles, photos, stills, posters and film fragments. An in-depth analysis and comparison of such documents can provide evidence on different film versions and cuts, and allow the restoration of lost/damaged films or identify actors and film fragments of unknown origin.

The archives have many characteristics in common: all of them have some of the largest film collections of their Countries; their collections are similar; they aim at serving as centers for information, documentation, preservation and mediation of films and related material for a large and varied community of national and international users; all have a special interest in developing indexing schemes for material in the collections and improving access to it by means of advanced technology (e.g., digitisation); all collaborate with relevant film institutions.

Based on these premises, all the material is analyzed, indexed, annotated and interlinked by film experts, for whom the COLLATE system will provide suitable task-based interfaces and knowledge management tools to support both individual work and collaboration. By continuously integrating the hereby-derived user knowledge into its digital data and metadata repositories, the system can offer an improved content-based retrieval functionality. Thus, enabling users to create and share valuable knowledge on the cultural, political and social contexts in turn allows other end-users to retrieve and interpret the historic material better.

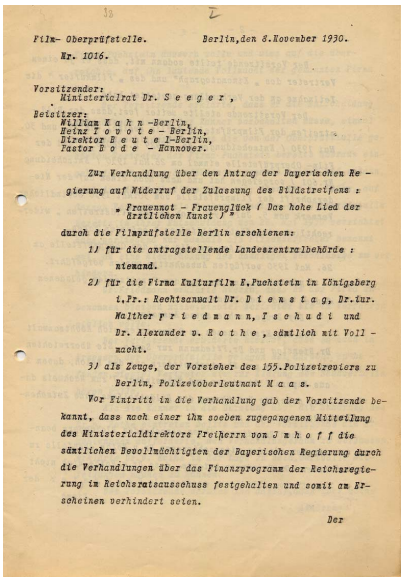
Supported by previous successful experience in the application of symbolic learning techniques to the classification and understanding of paper documents (Esposito et al., 1994; Esposito et al., 1998; Ferilli, 2000; Semeraro et al., 2001), our task is applying a multistrategy Machine Learning system, INTHELEX, to these documents. The objective is learning to automatically identify and label document classes and significant components, to be used for indexing/retrieval purposes and to be submitted to the COLLATE users for annotation. Indeed, by combining results from the manual and automatic indexing procedures, elaborate content-based retrieval mechanisms can be applied (Brocks et al., 2001).

We decided to focus our attention on three classes of documents. Specifically, the first class concerns registration cards provided by FAA (see Figure 1:a-b), corresponding to certifications that a film has been approved by the censoring authority for exhibition in the present version. The distribution company paid for the "registration cards" and enclosed them with the prints, whose owner or projectionist had to present them when the police checked the cinemas from time to time. The second class consists of censorship decisions from DIF (see Figure 1:c-d), representing decisions on whether a film could or could not be distributed and shown throughout the country, and in which version. The "censorship decision" is often a protocol of the examination meeting and is issued by the censorship office or headquarters. Finally, the last class we considered collects censorship cards from DIF (see Figure 1:e-f) and (fragments of) pages containing articles and images that concern the same films as the censorship documents (see Figure 1:g-h). This class is characterized by a high variability as regards the document size, layout and content, due to the different sources (journals, newspapers, etc.) from which the corresponding documents are drawn.

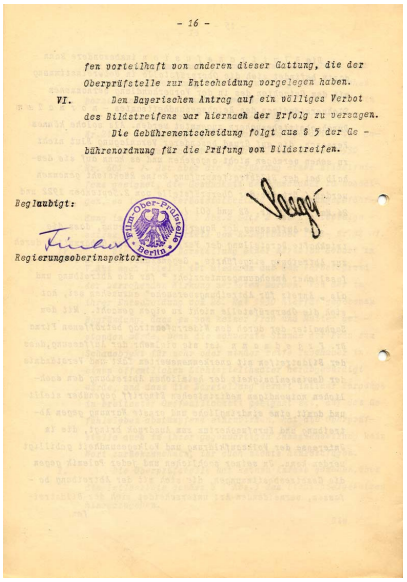


-a-

-b-

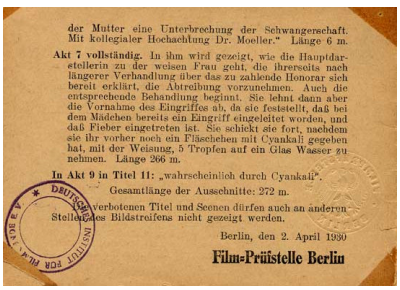


-c-



-d-

-e-



-f-

-g-

Beschwerden über den Remarque-Film

Die Interessengemeinschaft Groß-Berliner Filmerstaufführungstheater schreibt uns:

„Der Film ‚Im Westen nichts Neues', der seinerzeit bereits des öfteren die Oeffentlichkeit beschäftigt hat und der jetzt durch die Zensur für geschlossene Veranstaltungen freigegeben ist, begegnet seiner Vorführung vor weiten Kreisen des Publikums wieder erheblichen Schwierigkeiten. Dieses Mal aber liegen diese Schwierigkeiten bei der Firma, die diesen Film herausbringt. Sie verlangt nämlich von den Lichtspieltheaterbesitzern Leihmieten, die ca. das Doppelte der sonst üblichen Verleihpreise ausmachen. Außerdem verlangt sie aber vom Theaterbesitzer, daß er den Film zu erhöhten Eintrittspreisen bei der herrschenden wirtschaftlichen Not vorführt. Es ist im höchsten Maße bedauerlich, daß dadurch den Lichtspieltheaterbesitzern die Möglichkeit genommen wird, diesen Film breiteren Schichten der Bevölkerung zu normalen Preisen zu zeigen."

-h-

Figure 1 Sample COLLATE documents: first and last page of an FAA censorship card (a-b); first and last page of a DIF censorship decision (c-d); newspaper articles (e-f).

The challenge comes from the low layout quality and standard of such material, which results in a considerable amount of noise in its description. The layout quality is often affected by manual

annotations, stamps that overlap 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 should account for a profitable use of automated reasoning capabilities in INTHELEX, such as abduction and abstraction (described in the following sections). While the former can make the system more flexible in the absence of particular layout components, due to the typist's style, the latter can help in focusing on layout patterns that are meaningful to the identification of the interesting ones, neglecting less interesting details.

INTHELEX: AN INCREMENTAL LEARNING SYSTEM

INTHELEX (INcremental THEory Learner from EXamples) is a learning system embedded in the overall architecture as a learning component. In a nutshell, this is an ILP system that is able to induce conceptual descriptions, in the form of *hierarchical* logic theories, from positive and negative examples.

Some of its features (see (Esposito et al., 2000a) for a more complete introduction to this system) can be briefly summed up as follows:

- it is based on the *Object Identity assumption*, according to which terms denoted by different names within a formula must refer to different objects of the domain; such an assumption, which is adopted in the database theory or even in the event calculus, often corresponds to human intuition, while allowing the search space to fulfill suitable properties affecting efficiency and effectiveness of the learning process (Semeraro et al., 1998);
- it learns theories expressed as sets of Datalog^{OI} clauses (Semeraro et al., 1998) from examples expressed in the same language;
- it can learn simultaneously *multiple concepts*, possibly related to each other according to a given hierarchy (recursion is not allowed);
- it is *fully incremental*: In addition to the possibility of refining a previously generated version of the target concepts, learning can also start from an empty theory;
- it is a *closed loop* learning system – i.e. a system in which the learned theory is checked for validity on any new example; in case of failure, a revision process is activated in order to restore completeness and consistency;
- it retains all the processed examples, so as to guarantee validity of the learned theories on all of them.

Incremental learning is necessary when either incomplete information is available at the time of initial theory generation, or the nature of the concepts evolves dynamically. Both cases are 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 hence require the whole set of observations to be available from the beginning. In a closed loop process, feedback on performance is used to activate the theory revision phase (Becker, 1985).

The logical architecture of INTHELEX is organized as in Figure 2. 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 abstracted and stored in the base of processed examples, then exploited by the Rule Generator 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 Rule Interpreter checks the validity of the theory against new available examples, also abstracted and stored in the example base, taking the set of inductive hypotheses and a tuning/test example as input and producing a decision. The Critic/Performance Evaluator compares such a decision to the correct one. In the case of incorrectness on a tuning example, it can locate the cause of the wrong decision

and choose the proper kind of correction, firing the theory revision process. In this way, tuning examples are exploited incrementally by the Rule Refiner to modify incorrect hypotheses according to a data-driven strategy. The Rule Refiner consists of two distinct modules, a Rule Specializer and a Rule Generalizer, which attempt to correct hypotheses that are too weak or too strong, respectively. 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. Both the Rule Generator and the Rule Interpreter may exploit abduction to hypothesize facts that are not explicitly present in the observations.

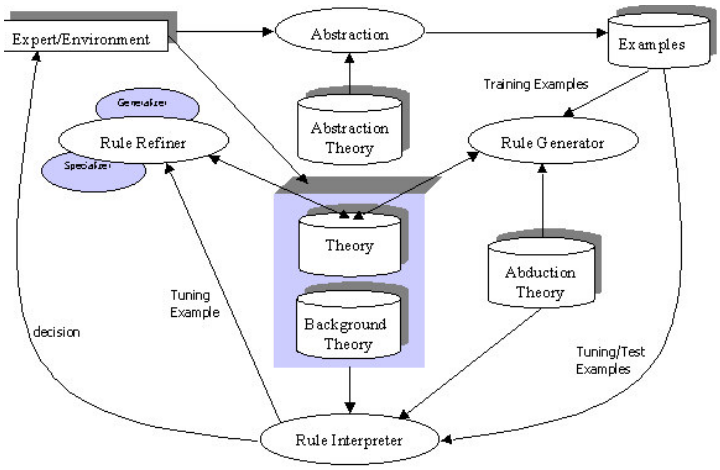


Figure 2 INTHELEX Architecture

INTHELEX is developed in SICStus PROLOG and is currently available in binary format for i586 DOS-based platforms (`http://lacam.di.uniba.it:8000/systems/inthelex/`). It is composed of various modules, corresponding to the logical functions of the system, plus a number of libraries. For each learning problem, the user must provide related information in a number of files.

MULTISTRATEGY LEARNING IN INTHELEX

Initially Machine Learning research focused on single-strategy methods that apply a primary type of inference and/or computational mechanism, but more recently the limitations of these methods has led to exploiting/combining various, different, complementary learning strategies. This reproduces the typical ability of humans to apply a great variety of learning strategies depending on the particular situation and problem faced. An underlying theoretical framework for integrating different learning strategies is the Inferential Learning Theory (ILT), presented in (Michalski, 1994).

Another peculiarity of INTHELEX is the integration of multistrategy operators that may help solve the theory revision problem by pre-processing the incoming information (Ferilli, 2000). 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. On the other hand, 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 though the perspectives are opposite, both aim at reducing the computational effort required to learn a correct theory, with respect to the incoming examples. More details on the theoretical foundations of the cooperation of these strategies in our environment are given in (Esposito et al., 2000c).

Induction

INTHELEX incorporates two inductive refinement operators, one for generalizing hypotheses that reject positive examples, and the other for specializing hypotheses that explain negative examples. It exploits a (possibly empty) previous theory, a graph describing the dependence relationships among concepts, 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 the base of processed examples. Then, 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 to which the example refers is chosen by the system for generalization. The lgg_{OI} of this clause and the example is computed (Semeraro et al., 1998), 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 the lgg_{OI}'s 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 the lgg_{OI}. 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 SLD-derivation of the example, INTHELEX tries to specialize one at the lowest possible level in the dependency graph by adding one (or more) positive literal(s) to it, 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 the case of failure in all 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 or which the example is an instance. 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.

Deduction

INTHELEX requires the observations to be expressed only in terms of a set of 0-level predicates that has been chosen to make up the description language for the given learning problem. Such predicates are basic, in the sense that they have no definition: This is to ensure uniformity of the example descriptions. Nevertheless, since the system is able to handle a hierarchy of concepts, combinations of basic predicates might identify higher level concepts that are 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 example 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. So, any time a new example is considered, a preliminary saturation phase can be performed, adding 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, which is based on a model (and proof)-theory that takes into account this assumption (Esposito et al., 2001) and hence turns out to be different from the standard semantics used for first order representations.

Given a set of terms T, a substitution σ is an *OI-substitution w.r.t. T* if and only if for all t_1, t_2 in T: $t_1 \neq t_2$ implies $t_1\sigma \neq t_2\sigma$. In this setting, an interpretation I is an *OI-model* for a clause C if and only if for all ground OI-substitutions γ it holds that $I \cap C\gamma \neq \emptyset$; I is an OI-model for a set of clauses Σ if and only if it is an OI-model for all clauses in Σ. Now, a set of clauses Σ *OI-implies* a clause C ($\Sigma \models_{OI} C$) if and only if all OI-models I for Σ are also OI-models for C. A sound and refutation-

complete proof-theory has been built upon this semantics, by defining notions of OI-unifiers, OI-resolution and OI-derivation ($\vdash_{OI}$). It holds that:

Theorem (*Subsumption Theorem*) *Let Σ be a finite set of clauses and C be a clause. Then $\Sigma \models_{OI} C$ iff a clause D exists such that $\Sigma \vdash_{OI} D$ and D θ_{OI}-subsumes C where θ_{OI}-subsumption is a variant of classical θ-subsumption that fulfills the Object Identity assumption (Semeraro et al., 1998).*

Differently from abstraction (see the following), 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. This is fundamental if some concepts to be learnt are related, since their definition could not be stable yet. Hence one cannot afford to drop the source from which deductions were made in order to be able to recover from incorrect deductions made because of wrong rules.

Abduction

Induction and abduction are both important strategies for performing hypothetical reasoning (i.e., inferences from incomplete information). The purpose of induction is to infer regularities and laws (from a certain number of significant observations) that may be valid for the whole population. Abduction was defined by Peirce as the hypothesis of some facts that, together with a given theory, could explain a given observation.

The inductive core of INTHELEX was augmented with abduction capabilities (Esposito et al., 2000a) to help to manage situations where not only the set of all observations is partially known, but each observation could also be incomplete. Thus, in our system, abduction is preliminarily used to generate suitable or relevant background data on which theory induction is based.

According to the framework proposed in (Lamma et al., 2000), an *abductive logic program* (or *theory*) is a triple $T = <P, A, IC>$ where:

- *P* is a normal logic program (Lloyd, 1987), which in our case is the theory that is being learned;
- *A* is a set of abducible predicates (called *abducibles*);
- *IC* is a set of *integrity constraints* in the form of denials, each corresponding to a combination of literals that is not allowed to occur.

Abducibles are the predicates on which assumptions (*abductions*) can be made: They are meant to carry all the incompleteness of the domain (if it were possible to complete these predicates then the theory would be correctly described). Integrity constraints provide indirect information on them and, since several explanations may hold for this problem setting, are also exploited to encode preference criteria for selecting the best ones.

The proof procedure implemented in INTHELEX starts from a goal *G* and a set of initial assumptions and returns a set of consistent hypotheses (abduced literals) by exploiting intertwined *abductive* and *consistency derivations*. Intuitively, an abductive derivation is the standard Logic Programming derivation suitably extended in order to consider abducibles. As soon as an abducible atom δ is encountered, it is added to the current set of hypotheses, provided that any integrity constraint containing δ is satisfied. To check the fulfillment of integrity constraints with respect to δ, a consistency derivation is started. Every integrity constraint containing δ is considered satisfied if the goal obtained by removing δ from it fails. In the consistency derivation, when an abducible is encountered, an abductive derivation for its complement is started in order to prove its falsity, so that the constraint is satisfied.

Abstraction

Abstraction is another pervasive activity in human perception and reasoning. When we focus on 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 (Zucker, 1998), concept representation deals with entities belonging to three different levels. Underlying any source of experience there is the *world W*, where *concrete* objects (the 'real things') reside. It is not directly known, since any observer's access is mediated by his *perception* of it *P(W)*. The percepts reality consists of the effect 'physical' stimuli have on the observer. To be available over time, these stimuli must be memorized in an organized *structure S*, i.e. an *extensional* representation of the perceived world, in which stimuli related to each other are stored together. Finally, to reason on the perceived world and communicate with other agents, a *language L* is needed, that describes it *intensionally*. Thus, a *reasoning context* is defined as $R = (P(W),S,L)$. Here situations are supposed not to change in time. If we assume that the three levels of *R* are generated upwards (i.e., *P(W)* is the source of information, that is recorded into *S* and then described by *L*), modifications to the structure and language are just a consequence of differences in the perception of the world. This may occur for a number of reasons (e.g., the medium used and the focus-of-attention), even though the world itself does not change. Thus, abstraction takes place at the world-perception level *P(W)* 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 contains information for performing the shift specified by the abstraction operators. In INTHELEX, it is assumed that the abstraction theory is already given (i.e. it has not to be learned by the system), and that the system automatically applies it to the learning problem at hand before processing the examples. The implemented abstraction operators allow the system to replace a number of components with 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 a certain kind of object.

INTHELEX REPRESENTATIONS

The learning problem solved by INTHELEX can be formulated as follows:
Given

- a set of concepts $C_1, C_2, \ldots, C_r$ to be learned, whose possible dependencies are represented through a graph;
- a set of tagged observations *O* referring to positive and negative instances of the concepts;
- a background knowledge *BK* (optional) supposed to be correct and hence not modifiable;
- a logical theory *T* (optional) for concepts $C_1, C_2, \ldots, C_r$, that is complete and consistent with respect to a set of past examples *E*;
- an abstraction theory *AT* (optional, described later);
- a set *A* of abducibles and a set *IC* of integrity constraints for abduction (optional, described later);
- a set of parameters according to which learning must be performed;

Find

- a (refinement *T'* of the) logical theory *T*, that is complete and consistent with respect to both *O* and *E*.

In the following, the formalism used by INTHELEX to represent the data and results that it manages is presented through simple examples. The kind of theories it handles naturally led to the choice of First-Order Predicate Logic to express observations and rules. In particular, for the sake of uniformity, Horn function-free clauses in Prolog form were used.

Examples are definite ground Horn clauses, whose body describes the observation by means of only basic non-negated predicates of the representation language adopted for the problem in hand, and whose head is the tag that defines the class to which the observed object belongs. They should be interpreted as "the head is true *since* the literals in the body are true".

```
bicycle(b) :- has_pedals(b,p),part_of(b,wb1),part_of(b,wb2),
      circular(wb1),has_rim(wb1),has_tire(wb1),
      circular(wb2),has_rim(wb2),has_tire(wb2).
```

is a positive example for the concept `bicycle/1`, to be interpreted as "*b* is a bicycle *since* it has pedals *p* and two circular parts (*wb1* and *wb2*) that have a rim and a tire". According to the same intuitive approach, negative examples differ from positive ones because of their head being negated.

```
not(bicycle(m)) :- has_engine(m),part_of(m,wm1),part_of(m,wm2),
      circular(wm1),has_rim(wm1),has_tire(wm1),
      circular(wm2),has_rim(wm2),has_tire(wm2).
```

to be interpreted as "*m* is not a bicycle *since* it has an engine and two circular parts (*wm1* and *wm2*) that have a rim and a tire", is a negative example for the same concept. When learning many concepts, it is possible that a single observation, i.e. the description of an object/situation, stands as an example or a counterexample for more than one concept. Since teachers typically describe each observation just once and then tag it in all possible ways, the system allows the specification in the head of a list of all the classifications that can be associated to the observation described in the body. For instance,

```
[not(monocycle(bb)),bicycle(bb),not(motorcycle(bb))] :-
      has_saddle(bb,s),has_pedals(bb,p),has_frame(bb,f),
      part_of(bb,w1),circular(w1),has_rim(w1),has_tire(w1),
      part_of(bb,w2),circular(w2),has_rim(w2),has_tire(w2).
```

means that "An object *bb* that has a saddle *s*, pedals *p*, a frame *f* and two circular parts, *w1* and *w2*, that have a rim and a tire, is not a monocycle nor a motorcycle, but it is a bicycle". This reflects the human way of thinking, but does not affect the system's inherent incrementality. Indeed, single classifications are processed separately, in the order they appear in the list, so that the teacher can still decide which concepts should be taken into account first and which should be taken into account later.

Inference rules that make up a theory are expressed as Horn definite and linked clauses, whose body describes the properties that an observation must fulfil in order to classify it as an instance of the concept reported in the head. Hence, they should be interpreted as "If an object satisfies the conditions in the body *then* it is an instance of the class in the head". Their body may include negated literals (coming from the specialization operator) to be interpreted according to Negation As Failure (it is not possible to prove that they are true).

```
bicycle(X) :-
      part_of(X,W1),part_of(X,W2),wheel(W1),wheel(W2),
      has_pedals(X,P),not(has_engine(X)).
```

is to be interpreted as "If an object *X* has two parts that are wheels, *W1* and *W2*, it has pedals *P*, and it has no engine, *then* it is a bicycle".

The theory also includes *exceptions*, each one referring to a specific example. A negative exception expresses the fact that a classification does *not* hold, which is expressed by "*!, fail*" in the clause body. A positive exception expresses the fact that a classification *does* hold:

```
bicycle(m') :- !,fail.
bicycle(b').
```

"*m'* is not a bicycle"; "*b'* is a bicycle". The background knowledge is expressed in the same way as the rules, but the body of its clauses starts with a "*true*" predicate in order to recognize them and prevent them from being modified by the refinement operators.

```
wheel(X) :- true,circular(X),has_rim(X,C),has_tire(X,T).
```

is to be interpreted as "An object *X* is a wheel if it is circular, it has a rim *C* and a tire *T*". The relations among the concepts are expressed in a dependency graph represented as a set of clauses like the following:

```
bicycle(X) :- wheel(X),mechanics(X,Y).
mechanics(X,Y) :- bicycle_chain(X,Y,Z),front_gear(X,Y),rear_gear(X,Y).
```

"concept `bicycle/1` depends on concepts `wheel/1` and `mechanics/2`"; "concept `mechanics/2` depends on concepts `bicycle_chain/3`, `front_gear/2` and `rear_gear/2`". Note that in the graph the variables are used as placeholders to indicate the arity of concepts.

Given the above setting, since concept 'bicycle' depends on concept 'wheel', and in the example description for bicycle *bb* it is possible to recognize two wheels (*w1* and *w2*), according to the definition in the background knowledge, the example can be saturated in order to explicitly represent such a situation in the following way:

```
[not(monocycle(bb)),bicycle(bb),not(motorcycle(bb))] :-
      has_pedals(bb,p),has_saddle(bb,s),has_frame(bb,f),
      part_of(bb,w1),wheel(w1),circular(w1),has_rim(w1),has_tire(w1),
      part_of(bb,w2),wheel(w2),circular(w2),has_rim(w2),has_tire(w2).
```

To abduce literals, the system needs to know the integrity constraints that the hypothesized information must fulfil. Two examples of integrity constraints follow:

```
[circular(X),square(X)].
[has_double_seat(X,Y,Z),has_pedals(X,W),has_engine(X,T)].
```

The former means that "either *X* is circular or *X* is square" (but it cannot be both at the same time). The latter indicates that the specified events cannot all hold at the same time (at least one of them must not hold): for instance, an object may have a double seat and pedals, but no engine (tandem); a double seat and an engine, but no pedals (motorcycle), etc.

Lastly, each instance of an abstraction operator is represented in the abstraction theory as a clause in the following form:

$[l_1, l_2, \ldots, l_n] :- b_1, b_2, \ldots, b_m.$ $\quad n \geq 0, m > 0,$

meaning that any occurrence of the whole combination of literals $b_1, b_2, \ldots, b_m$ in an observation must be replaced by the literals $l_1, l_2, \ldots, l_n$ that represent their abstraction. Note that, unlike the dependency graph representation, in this case possible correspondences in the arguments of the literals are meaningful.

EXPERIMENTS

Some experiments were run to test the improvement due to the addition of abduction and abstraction in the process of learning definitions for the classes of COLLATE documents introduced above (see Figure 1): FAA registration cards (`faa_registration_card`) and DIF

censorship decisions (`dif_censorship_decision`). Specifically, the dataset consisted of 34 documents for the class `faa_registration_card`, 19 documents for the class `dif_censorship_decision` and 61 reject documents, obtained from newspaper articles and DIF registration cards.

The first order descriptions of these documents, needed to run the learning system, were automatically generated through a pre-processing phase carried out by the system WISDOM++ (Esposito et al., 2000b). 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, by means of Machine Learning techniques. Figure 3 shows (a) the original document just loaded in WISDOM++; (b) the layout structure of the document identified after the pre-processing phase; and finally (c) the superimposition of the original document on the extracted layout, that evidences the precision obtained in singling out the document layout blocks. 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). A detailed list of all the descriptors used, along with the corresponding domains, is reported in Table 1. The description length of the documents for class `faa_registration_card` ranges between 40 and 379 literals (144 on average); for class `dif_censorship_decision` it ranges between 54 and 263 (215 on average).

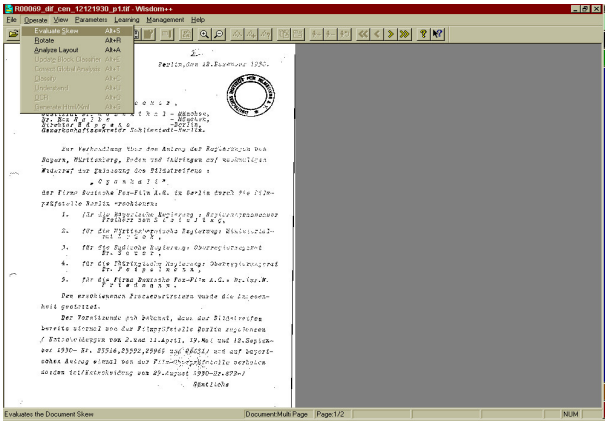
Each document was considered a positive example for the class to which it belongs, and a negative example for the other class to be learned; reject documents were considered negative examples for both classes. Definitions for each class were learned, starting from the empty theory and with all the negative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

Various experiments were performed on both classes, whose results (all averaged on the 10 runs) are reported in Table 2 (as regards `faa_registration_card`) and in Table 3 (concerning `dif_censorship_decision`). For each case, the following data are reported:

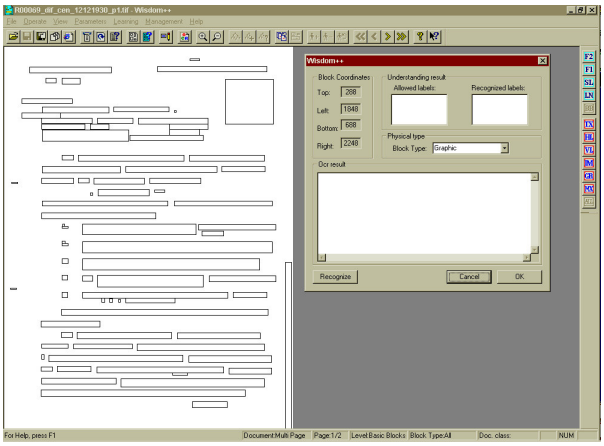
- *Clauses:* number of clauses (i.e. alternative definitions for the concept) in the learned theory;
- *Length:* number of literals making up the clauses;
- *Lgg:* number of generalizations needed to obtain the theory;
- *Runtime:* computational time required to learn the theory (expressed in seconds);
- predictive accuracy rate computed on the test set: *Accuracy* concerns the whole test set, *Pos* refers to positive examples only, and *Neg* to negative examples only.

In the following paragraphs the experimental results will be discussed in more detail, and we will try to justify them.

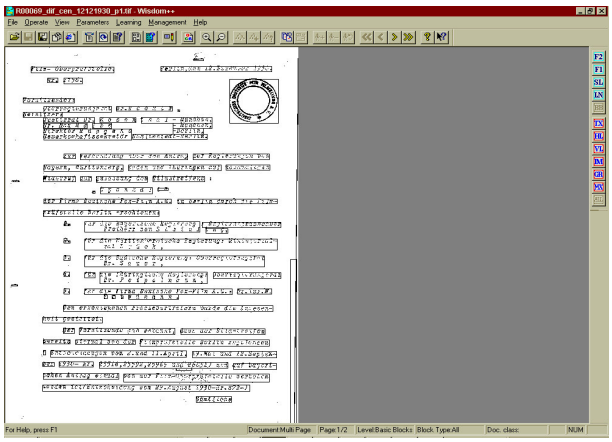
The first problem is that INTHELEX is currently unable to handle numeric descriptors, whereas WISDOM++ expresses the block dimensions and positions in the form of numeric values (number of pixels). Hence, a discretization was needed to assign a symbolic label representing an interval to each specific value. When INTHELEX had not been extended with abstraction, we had to perform such a transformation through a purposely implemented routine. Now, we are able to delegate this task to the system itself, by exploiting one of its abstraction operators (the one acting on the grain size of the descriptors). Comparing the first two rows of Table 2 with the same rows in Table 3, that report performance in both cases, we can note that there is no loss in the ‘automatic abstraction’



a



b



c

Figure 3 Original image (a); result of pre-processing step (b); superimposition of the original image with the identified layout blocks (c).

case (second row) with respect to the 'hand-made' one (first row). Actually, there is a slight improvement (except in the computational time for the first class), probably due to abstraction changing the ordering of the literals in the observations. Indeed, INTHELEX does not currently embed any heuristic to choose specific literals on which to apply its refinement operators, and hence depends on the order in which they are provided in the observations. Thus, all the subsequent experiments were run by charging INTHELEX in the numeric discretization phase by abstraction.

Such experiments checked the effects of exploiting the multistrategy capabilities of INTHELEX on the previously obtained baseline.

Table 1. Description language for representing COLLATE documents

Predicate	**Domain**
image_length(doc,length)	Integer: (1 .. 5000)
image_width(doc,width)	Integer: (1 .. 4000)
width(block,width)	Integer: (1..640)
height(block,height)	Integer: (1..890)
x_pos_centre(block,pos)	Integer: (1..640)
Y_pos_centre(block,pos)	Integer: (1..875)
type_of(block,type)	Nominal: text, hor_line, image, ver_line, graphic, mixed
Part_of(page,block)	Boolean: true if page contains block
on_top(block1,block2)	Boolean: true if block1 is above block2
to_right(block1,block2)	Boolean: true if block2 is to the right of block1
alignment(block1,block2,alignment)	Nominal: only_left_col, only_right_col, only_middle_col, both_columns,only_upper_row, only_lower_row, only_middle_row, both_rows

Since the available documents are often affected by the presence of speckles, that are identified by WISDOM++ as layout components (and hence appear in the document descriptions), we decided to use abstraction to eliminate them. The underlying rationale is that such a 'cleaning up' would hopefully help the system in at least two respects: first, to focus on significant layout components, that are more discriminant and hence should lead to more characterizing definitions of the concepts; second, to have shorter example descriptions, that could have positive effects on the learning time. Specifically, the abstraction theory considered as speckles all the blocks without a specific type of content (`mixed`) that are short and/or narrow.

Another issue to be faced in the COLLATE project is the low quality of some documents, due to their age and to the absence, in many cases, of a standard layout. While the documents in our dataset were chosen because they had an acceptable quality and a sufficiently standard layout, it is foreseeable that worse documents will be available in the future. To simulate the behaviour of INTHELEX in such a possible scenario, we corrupted part of the documents in the dataset by eliminating part of their description, then we tried to apply abduction in order to overcome the problems due to these missing components. Specifically, incomplete documents were generated by randomly dropping 10% of the description from 30% of the available documents, and then letting INTHELEX use abduction. All the basic predicates in the description language concerning block dimensions, type and position were considered as abducibles, while integrity constraints were set to express the mutual exclusion among layout block sizes, types and positions. INTHELEX was allowed to exploit abduction to hypothesize facts, concerning the above descriptors, only in the case of failure in finding a correct generalization, before adding a new clause to the theory. Abduction makes sense in this environment since the absence of a layout block in a document could be due to the writer not fulfilling the style requirements, and not to the insignificance of that block to a correct definition. In other words, a block should not be dropped from the definition just because a few examples miss it; conversely, integrity constraints avoid superfluous blocks in the first few examples introducing unnecessary blocks that can always be abduced in the future.

The last three rows of Table 2 and Table 3 report the experiments with speckle abstraction only, abduction on the corrupted dataset only, and both, respectively (in the last case, abstraction precedes abduction). Rows involving abduction also report, in parentheses for comparison purposes, the performance that would be obtained on the corrupted dataset without exploiting abduction.

Let us now focus on the experiments concerning the `faa_registration_card` class (see Table 2). Abstraction of speckles cannot, of course, improve the number of clauses (that is already 1). Nevertheless, the shorter example descriptions have a beneficial effect on the learning time, in spite of the greater number of generalizations performed. This is probably due to the absence of speckles in negative examples, that formerly helped to avoid their coverage. The greater difficulty

Table 2. Classification results for class `faa_registration_card`

	Clauses	Length	lgg	Runtime	Accuracy	Pos	Neg
'Manual' Discretization	1	10,5	9,5	11,9	0,981	0,940	1,0
Numeric abstraction	1	13,1	8	19,29	0,991	0,971	1,0
Speckle abstraction	1	12,9	8,7	18,24	0,981	0,940	1,0
Abduction	1,1 (2)	13,1 (40,7)	8,7 (9,7)	123,69 (40,9)	0,991 (0,991)	0,971 (0,971)	1,0 (1,0)
Abduction+ Abstraction	1 (2)	12,5 (42)	8 (9)	90,13 (36,42)	1,0 (0,991)	1,0 (0,971)	1,0 (1,0)

Table 3. Classification results for class `dif_censorship_decision`

	Clauses	Length	lgg	Runtime	Accuracy	Pos	Neg
'Manual' Discretization	1,6	52,3	8	71,92	0,934	0,737	0,989
Numeric abstraction	1	23,3	8,4	49,6	0,972	0,842	1,0
Speckle abstraction	1	23,3	8,6	47,64	0,972	0,842	1,0
Abduction	1,1 (1,2)	28,3 (33,4)	8,7 (8,7)	5667,8 (50,37)	0,953 (0,953)	0,737 (0,737)	1,0 (1,0)
Abduction+ Abstraction	1,1 (1,2)	25,8 (33,4)	9,2 (8,7)	5761,8 (50,37)	0,963 (0,943)	0,789 (0,737)	1,0 (0,989)

in avoiding coverage of negative examples also results in a more specific clause, as shown by the slight decrease in predictive accuracy on positive examples. As to abduction, the experiments prove that it is able to balance the corruption in the examples, even at the cost of a slightly worse number of clauses and lgg's, and of a significant increase in the runtime (due to the need to check consistency for each hypothesized fact). Indeed, predictive accuracy is the same as when only inductive operators are employed, although a great portion of the descriptions is now missing. The benefit becomes more evident, especially as regards the number of clauses and their length, if we compare the performance to what would be obtained on the corrupted dataset without exploiting abduction, as reported in parentheses. Finally, the results of the joint effort of abduction together with abstraction are shown in the last row. The theories learned in this case are, indeed, the best obtained among all cases: they outperform all the previous ones in predictive accuracy (that reaches 100%), and are better than any single strategy as to the number of clauses and lgg's performed. Also the runtime, even though worse than that of abstraction only, is far better than that of abduction only. The average number of literals in each clause is almost stable, with a slight decrease when abstraction is used, signifying that the system was always able to grasp the core concept and that abstraction actually eliminated only superfluous information.

Considering the experiments on the other class, `dif_censorship_decision` (see Table 3), we immediately note that the runtime is always higher than for the previous class, while the predictive accuracy is always lower (even though it is still very high). The former suggests that we are facing an intrinsically harder learning problem (indeed, documents in this class have larger size and a higher number of identified layout components – typically, each row in the document is considered a separate block). The latter may be due to the availability of less examples causing a less refined theory that is not predictive enough, as supported by the number of literals in the definitions. It seems that more characteristics than in the other class must be preserved to find a solution. Indeed, the portion of the original example length that is dropped from the definition ranges from 85,68% to 89,17% (against the 90,91–92,71% for the other class). The comments concerning speckle abstraction, made for the previous class, generally still hold, including the improvement that it can bring, when added to abduction, with respect to abduction alone. On the

other hand, this class seems to be particularly idiosyncratic with respect to abduction. In this respect, it should be noted that the runtime in the last two rows is greatly affected by the search for useful abductions in one fold, as proved by the fact that, if we consider only the remaining 9 folds, the mean decreases to 54,87 and 38,87, respectively. The poor performance of abduction in this case seems to be confirmed by the comparison with the statistics in parentheses, particularly as regards the clause length, for which the other class showed a significant improvement. This could be due to the fact that corrupting the descriptions makes an already difficult problem even harder. Another possible explanation is the fact that, corrupting the description of documents in this class, causes the elimination of many literals. This can be compensated to some extent by the generalizations, that are able to find other common (but probably less characterizing and meaningful) features among the given documents. Such accidental similarities could cause overfitting on the training data, which abduction is not able to compensate during the test phase. This is confirmed by the lower predictive accuracy concentrated in the coverage of positive examples and the fact that negative examples are always rejected.

The difference in effectiveness reached by abduction in the two classes led us to closely examine the phenomenon, in order to understand it and its possible causes better. Table 4 summarizes, for both classes, the average number of examples on which literals have been abduced (first row) and the average number of literals abduced for each of them (second row), both with and without abstraction of speckles. Our expectations were confirmed in that, for the class `dif_censorship_decision`, nearly no abduction was made, which explains why no improvement was obtained with respect to the other cases (including learning without abduction on corrupted examples). More specifically, we found that the only abductions were carried out for the 'computation-intensive' fold, which again suggests that this class seldom requires abduction, and in any case is a hard task. On the contrary, as regards class `faa_registration_card`, abduction improves performance since it succeeds in hypothesizing more literals (2 out of about 13 that make up the definition) and in more examples.

Table 4 Abduction performance

	faa_registration_card		**dif_censorship_decision**	
	w/ abstraction	**w/o abstraction**	**w/ abstraction**	**w/o abstraction**
Examples	1,9	1,6	0,1	0,1
Literals	1,53	1,5	0,2	0,2

Similarly, in Table 5 we have collected information on the effects of eliminating speckles from document descriptions through abstraction. For both kinds of document in the dataset, the number of examples in which abstraction took place is reported (first row), along with the average number of dropped literals (second row). It turns out that only one part of the whole training set suffered from speckles, and in that case a relevant portion of the descriptions was removed. It is noticeable that more speckles were found in `faa_registration_card` documents than in `dif_censorship_decision` ones, even though the average description length of the latter was greater than that of the former, which indicates a better layout quality in the latter. This also means that abstraction cannot lower the document description complexity for the latter class, which makes the other operators work harder.

Table 5 Speckle abstraction performance

	faa_registration_card	**dif_censorship_decision**	**Reject**
Examples	14	9	3
Literals	114	37	140

CONCLUSIONS

Multistrategy approaches to machine learning can help to obtain more efficiency, and are necessary in a number of real-world situations. The incremental system INTHELEX works on first-order logic representations. Its multistrategy learning capabilities have been further enhanced in order to improve effectiveness and efficiency of the learning process, by augmenting pure induction and abduction with abstraction and deduction. INTHELEX is included in the architecture of the EU project COLLATE, in order to learn rules for automated classification of cultural heritage documents dating back to the twenties and thirties.

This paper has presented and discussed some experimental results proving the benefits that the addition of each strategy can bring in the task of document classification. Although the performance obtained exclusively by inductive operators was very good, multistrategy operators contributed to make it more effective and the efficient.

ACKNOWLEDGEMENTS

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

(Becker, 1985) Becker, J. M. 1985. *Inductive Learning of Decision Rules with Exceptions: Methodology and Experimentation*. B.S. Dissertation, Dept. of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, UIUCDCS-F-85-945.

(Brocks et al., 2001) Brocks, H., U. Thiel, A. Stein, and A. Dirsch-Weigand. 2001. Customizable Retrieval Functions Based on User Tasks in the Cultural Heritage Domain, in *Research and Advanced Technology for Digital Libraries*, number 2163 in LNCS, P. Constantopoulos and I.T. Sølvberg, eds. Springer-Verlag, Berlin.

(Esposito et al., 1994) Esposito, F., D. Malerba, and G. Semeraro. 1994. A Multistrategy Learning Approach to Document Recognition. *Applied Artificial Intelligence: an International Journal*, 8:33-84.

(Esposito et al., 1998) Esposito, F., D. Malerba, G. Semeraro, N. Fanizzi, and S. Ferilli. 1998. Adding Machine Learning and Knowledge Intensive Techniques to a Digital Library Service. *International Journal on Digital Libraries*, 2(1):3–19.

(Esposito et al., 2000a) Esposito, F., G. Semeraro, N. Fanizzi, and S. Ferilli. 2000a. Multistrategy Theory Revision: Induction and Abduction in INTHELEX. *Machine Learning Journal*, 38(1/2):133–156.

(Esposito et al., 2000b) Esposito, F., D. Malerba, and F.A. Lisi. 2000b. Machine Learning for Intelligent Processing of Printed Documents. *Journal of Intelligent Information Systems*, 14(2/3):175–198.

(Esposito et al., 2000c)Esposito, F., N. Fanizzi, S. Ferilli and G. Semeraro. 2000c. Abduction and Abstraction in Inductive Learning", *Proceedings of the 5th International Workshop on Multistrategy Learning*, MSL00, Guimaraes, Portugal

(Esposito et al., 2001) Esposito, F., N. Fanizzi, S. Ferilli and G. Semeraro. 2001. OI-implication: Soundness and Refutation Completeness, *Proceedings of the 17th Interlantioanl Joint Conference on Artificial Intelligence*, IJCAI01, Morgan Kaufmann Publishers, B. Nebel, ed., San Francisco, CA, 847–852.

(Ferilli, 2000) 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, Italy.

(Kouzes et al. 1996) Kouzes, R.T., J.D. Myers, and W.A. Wulf. 1996. Collaboratories: Doing Science on the Internet. *IEEE Computer*, 29(8).

(Lamma et al., 2000) Lamma, E., P. Mello, F. Riguzzi, F. Esposito, S. Ferilli, and G. Semeraro (2000). Cooperation of Abduction and Induction in Logic Programming, in *Abductive and Inductive Reasoning: Essays on their Relation and Integration*, A. C. Kakas and P. Flach, eds., Kluwer.

(Lloyd, 1987) Lloyd, J. W. 1987. *Foundations of Logic Programming*, second edition, Springer-Verlag, Berlin.

(Michalski, 1994) Michalski, R. S.. 1994. Inferential Theory of Learning. Developing Foundations for Multistrategy Learning, in *Machine Learning. A Multistrategy Approach*", volume IV, R. S. Michalski and G. Tecuci, eds., Morgan Kaufmann, San Mateo, CA, 3–61.

(Semeraro et al., 1998)Semeraro, G., Esposito, D. Malerba, N. Fanizzi, and S. Ferilli. 1998. A Logic Framework for the Incremental Inductive Synthesis of Datalog Theories, in *Logic Program Synthesis and Transformation*, number 1463 in LNCS, N. E. Fuchs, ed.. Springer-Verlag, Berlin.

(Semeraro et al., 2001) Semeraro, G., S. Ferilli, N. Fanizzi, and F. Esposito. 2001. Document Classification and Interpretation through the Inference of Logic-based Models. in *Research and Advanced Technology for Digital Libraries*, number 2163 in LNCS, P. Constantopoulos and I.T. Sølvberg, eds. Springer-Verlag, Berlin.

(Zucker, 1998) Zucker. J.-D. 1998. Semantic Abstraction for Concept Representation and Learning. *Proceedings of the 4th International Workshop on Multistrategy Learning*, MSL98, R. S. Michalski and L. Saitta, eds., Desenzano del Garda, Italy.