Evaluation and Validation of Two Approaches to User Profiling

F. Esposito, G. Semeraro, S. Ferilli, M. Degemmis,
N. Di Mauro, T.M.A. Basile, and P. Lops

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

Abstract. In the Internet era, huge amounts of data are available to everybody, in every place and at any moment. Searching for relevant information can be overwhelming, thus contributing to the user's sense of information overload. Building systems for assisting users in this task is often complicated by the difficulty in articulating user interests in a structured form - a profile - to be used for searching. Machine learning methods offer a promising approach to solve this problem. Our research focuses on supervised methods for learning user profiles which are predictively accurate and comprehensible.
The main goal of this paper is the comparison of two different approaches for inducing user profiles, respectively based on Inductive Logic Programming (ILP) and probabilistic methods. An experimental session has been carried out to compare the effectiveness of these methods in terms of classification accuracy, learning and classification time, when coping with the task of learning profiles from textual book descriptions rated by real users according to their tastes.

1 Introduction

The ever increasing popularity of the Internet has led to a huge increase in the number of Web sites and in the volume of available on-line data. Users are swamped with information and have difficulty in separating relevant from irrelevant information. This leads to a clear demand for automated methods able to support users in searching the extremely large Web repositories in order to retrieve relevant information with respect to users' individual preferences. The problem complexity could be lowered by the automatic construction of machine processable profiles that can be exploited to deliver *personalized* content to the user, fitting his or her personal interests.

Personalization has become a critical aspect in many popular domains such as e-commerce, where a user explicitly wants the site to store information such as preferences about himself or herself and to use this information to make recommendations. Exploiting the underlying one-to-one marketing paradigm is essential to be successful in the increasingly competitive Internet marketplace.

Recent research on intelligent information access and recommender systems has focused on the content-based information recommendation paradigm: it requires textual descriptions of the items to be recommended [6].

In general, a content-based system analyzes a set of documents rated by an individual user and exploits the content of these documents to infer a model or profile that can be used to recommend additional items of interest.

The user's profile is built and maintained according to an analysis that is applied to the contents of the documents that the user has previously rated. For example, a user profile can be a text classifier able to distinguish between interesting and uninteresting documents. In recent years, text categorization, which can be defined as the content-based assignment of one or more predefined categories to text, has emerged as an application domain to machine learning techniques. Many approaches that suggest the construction of classifiers using induction over preclassified examples have been proposed [13]. These includes numerical learning, such as Bayesian classification [4] or symbolic learning like in [8]. In [5] are presented empirical results on text categorization performance of two inductive learning algorithms, one based on Bayesian classifiers and the other on decision trees. They attempt to study the effect that characteristics of text have on inductive learning algorithms, and what are the performance of purely learning-based methods. They found that feature selection mechanisms are of crucial importance, due to the fact that the primary influence on inductive learning applied to text categorization is the large number of features that natural language provides. In this paper we present a comparison between two different learning strategies to infer models of users' interests from text: an ILP approach and a naïve bayes method. Motivation behind our research is the realization that user profiling and machine learning techniques can be used to tackle the *relevant information problem* already described.

The application of text categorization methods to the problem of learning user profiles is not new: the LIBRA system [7] makes content-based book recommending by applying a naïve Bayes text categorization method to product descriptions in Amazon.com. A similar approach, adopted by Syskill & Webert [10], tracks the users browsing to formulate user profiles. The system identifies informative words from Web pages to be used as boolean features and learns a naïve Bayesian classifier to discriminate interesting Web pages on a particular topic from uninteresting ones. The authors compare six different algorithms from machine learning and information retrieval on the task and they find that the naïve Bayesian classifier offers several advantages over other learning algorithms. They also show that the Bayesian classifier performs well, in terms of both accuracy and efficiency. Therefore, we have decided to use the naïve Bayesian classifier as the default algorithm in our Item Recommender system because it is very fast for both learning and predicting, which are crucial factors in learning user profiles. The learning time of this classifier is linear in the number of examples and its prediction time is independent of the number of examples. Moreover, our research aims at comparing this technique with a symbolic approach able to induce profiles that are more readable from a human understandability viewpoint.

Experiments reported in this paper evaluated the effects of the ILP and the Bayesian methods in learning intelligible profiles of users' interests. The experiments were conducted in the context of a content-based profiling system for virtual bookshop on the World Wide Web. In this scenario, a client side utility has been developed in order to download documents (book descriptions) for a user from the Web and to capture users feedback regarding his liking/disliking on the downloaded documents. Then this knowledge can be exploited by the two different machine learning techniques so that when a trained system encounters a new document it can intelligently infer whether this new document will be liked by the user or not. This strategy can be used to make recommendations to the user about new books. The experiments reported here investigate also the effect of using different representations of the profiles.

The structure of the remainder of the paper is as follows: Section 2 describes the ILP system INTHELEX and its main features, while the next section introduces Item Recommender, the system that implements a statistical learning process to induce profiles from text. Then a detailed description of the experiments is given in Section 4, along with an analysis of the results by means of a statistical test. Section 5 presents how user profiles can be exploited for personalization purposes. Finally, Section 6 draws some general conclusions.

2 INTHELEX

INTHELEX (INcremental THEory Learner from EXamples) [3] is a learning system for the induction of hierarchical theories from positive and negative examples which focuses the search for refinements by exploiting the Object Identity [14] bias on the generalization model (according to which terms denoted by different names must be distinct). It is fully and inherently incremental: this means that, in addition to the possibility of taking as input a previously generated version of the theory, learning can also start from an empty theory and from the first available example; moreover, at any moment the theory is guaranteed to be correct with respect to all of the examples encountered thus far. This is a fundamental issue, since in many cases deep knowledge about the world is not available. 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, which are unnegligible issues for learning user profiles. Indeed, generally users' accesses to a source of information are distributed in time, and the system is not free to choose when to start the learning process because a theory is needed since the first access of the user. Hence, for each new access the system tries to assess the validity of the theory (if available) with respect to this new observation. INTHELEX can learn simultaneously various concepts, possibly related to each other, and is based on a closed loop architecture — i.e. the learned theory correctness is checked on any new example and, in case of failure, a revision process is activated on it, in order to restore completeness and consistency.

INTHELEX learns theories expressed as sets of Datalog^{OI} clauses (function free clauses to be interpreted according to the Object Identity assumption). It adopts a full memory storage strategy — i.e., it retains all the available examples, thus the learned theories are guaranteed to be valid on the whole set of known examples — and it incorporates two inductive operators, one for generalizing definitions that reject positive examples, and the other for specializing definitions that explain negative examples. Both these operators, when applied, change the set of examples the theory accounts for.

The logical architecture of INTHELEX is organized as in Figure 1. A set of examples of the concepts to be learned, possibly selected by an Expert, is provided by the Environment. 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 at hand, and whose head lists all the classes for which the observed object is a positive example and all those for which it is a negative one (in this case the class is negated). 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. It is important to note that a positive example for a concept is not considered as a negative example for all the other concepts (unless it is explicitly stated). The set of all examples 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.

The Rule Generalizer is activated when a positive example is not covered, and a revised theory is obtained in one of the following ways (listed by decreasing priority) such that completeness is restored:

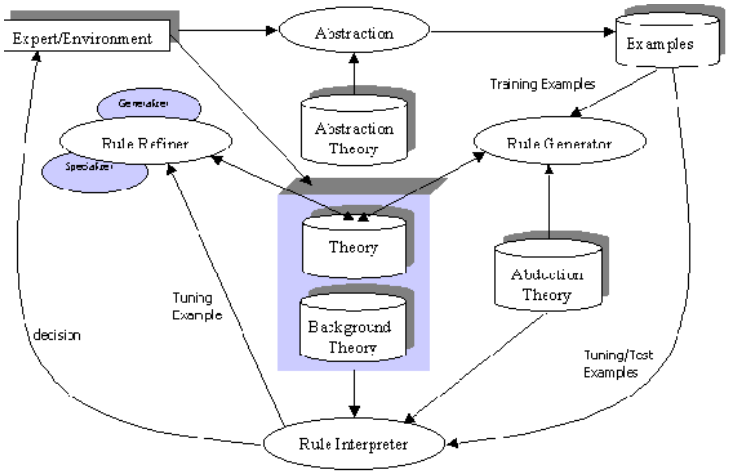

Fig. 1. INTHELEX architecture

- replacing a clause in the theory with one of its generalizations against the problematic example;
- adding a new clause to the theory, obtained by properly turning constants into variables in the problematic example;
- adding the problematic example as a positive exception.

While as regards the Rule Specializer, on the other hand, when a negative example is covered, the system outputs a revised theory that restores consistency by performing one of the following actions (by decreasing priority):

- adding positive literals that are able to characterize all the past positive examples of the concept (and exclude the problematic one) to one of the clauses that concur to the example coverage;
- adding a negative literal that is able to discriminate the problematic example from all the past positive ones to the clause in the theory by which the problematic example is covered;
- adding the problematic example as a negative exception.

An exception contains a specific reference to the observation it represents, as it occurs in the tuning set; new incoming observations are always checked with respect to the exceptions before the rules of the related concept. This does not lead to rules which do not cover any example, since exceptions refer to specific objects, while rules contain variables, so they are still applicable to other objects than those in the exceptions.

It is worth noting that INTHELEX never rejects examples, but always refines the theory. Moreover, it does not need to know *a priori* what is the whole set of concepts to be learned, but it learns a new concept as soon as examples about it are available.

2.1 Learning user profiles with INTHELEX

We were led by a twofold motivation to exploit INTHELEX on the problem of learning user profiles. First, its representation language (First-Order Logic) is more suitable than numeric/probabilistic approaches to obtain intuitive and human readable rules, which are a highly desirable feature in order to understand the user preferences. Second, incrementality is an undeniable requirement in the given task, since new information on a user is available each time he issues a query, and it would be desirable to be able to refine the previously generated profile instead of completely rejecting it and learning a new one from scratch. Moreover, a user's interests and preferences might change in time, a problem that only incremental systems are able to tackle.

INTHELEX is specifically designed to learn first-order logic theories. In particular, it is suitable when the descriptions of the concepts to be learned are not flat, i.e. they include not only the properties of the objects but also relations between them. However, as we will see later, in the given environment a user profile is described by a list of attributes with an associated value, which corresponds to a propositional representation rather than a first-order one. Hence, in this case the full potentiality of INTHELEX is not entirely exploited, and this should be taken into account when evaluating results.

A further problem that arises in this type of learning task is due to the lack of precise mental schemas in the user for rating a book. Indeed, in many cases, the choice of a book relies on the presence of details appearing in only a few descriptions (e.g., the name of the favourite author). In such a situation, learning a definition for the mental schema of a user becomes more difficult and the resulting profile will be imprecise. This problem is more evident when the system is provided with few users' preferences. On the contrary, when the user's accesses are more frequent, it should (hopefully) be easier to find what is the main trend of the user.

Since INTHELEX is not currently able to handle numeric values, it was not possible to learn preference rates in the continuous interval $[0, 1]$ like in the probabilistic approach. Thus, a discretization was needed. Instead of learning a definition for each of the 10 possible votes, we decided to learn just two possible classes of interest: "likes", describing that the user likes a book, and its opposite "not(likes)". Specifically, the former (positive examples) encompasses all rates ranging from 6 to 10, while the latter (negative examples) included all the others (from 1 to 5). It is worth noting that such a discretization step is not in charge of the human supervisor, since a proper abstraction operator embedded in INTHELEX can be exploited for carrying out this task. Moreover, it has a negligible computational cost, since each numeric value is immediately mapped onto the corresponding discretized symbolic value.

2.2 Representation of Profiles

Each book description is represented in terms of three components by using predicates `slot_title(b,t)`, `slot_author(b,au)`, and `slot_annotation(b,an)`, in-

dicating, respectively, that the book '`b`' contains a title, an author and an annotation, where the objects '`t`', '`au`' and '`an`' are, respectively, the title, author and annotation of the book '`b`'. Any word in the book description is represented by a predicate corresponding to its stem, and linked to both the book itself and the single slots in which it appears. For instance, predicate `prolog(slot_title,stp)` indicates that object '`stp`' has stem 'prolog' and is contained in slot '`slot_title`'; in such a case, also a literal `prolog(book)` is present to say that stem 'prolog' is present in the book description.

Also the number of occurrences of each word in each slot was represented by means of the following predicates: `occ_1`, `occ_2`, `occ_m`, `occ_12`, `occ_2m`. A predicate `occ_X(Y)` indicates that term Y occurs X times, while a predicate `occ_XY(Z)` indicate that the term Z occurs from X to Y times. Again, such a 'discretization' was needed because numeric values cannot be dealt with in INTHELEX. Note that all the predicates representing intervals to which the value to be represented belongs must be used to represent it; thus, many such predicates can be needed to represent the occurrences of a term. For instance, if a term occurs once, then it occurs also from 1 to 2 (`occ_12`) times and from 1 to m (`occ_1m`) times. Figure 2 shows an example for the class `likes`. Given the specific value in the example, all the intervals to which it belongs are automatically added by the system by putting this information in the background knowledge and exploiting its *saturation* operator. Predicates describing intervals are needed to obtain generalizations based also on the number of word's occurrences in a book. In particular, if a word w occurs once in a description d and twice in a description d', the possible generalizations of the number of occurrences are `occ_12`, `occ_1m`.

3 Item Recommender

ITR (ITem Recommender) [2] is a system able to recommend items based on their textual descriptions. It implements a probabilistic learning algorithm to classify texts, the naïve Bayes classifier. Naïve Bayes has been shown to perform competitively with more complex algorithms and has become an increasingly popular algorithm in text classification applications [10, 7].

The prototype is able to classify text belonging to a specific category as interesting or uninteresting for a particular user. For example, the system could learn the target concept "*textual descriptions the user finds interesting in the category Computer and Internet*".

Bayesian reasoning provides a probabilistic approach to inference. It is based on the assumption that the quantities of interest are governed by probabilistic distributions and that optimal decision can be made by reasoning about these probabilities together with observed data.

In the learning problem, each instance (item) is represented by a set of *slots*. Each slot is a textual field corresponding to a specific feature of an item.

```
likes(book_501477998) :-
     slot_title(book_501477998, slott),
     practic(slott, slottitlepractic),
     occ_1(slottitlepractic),
     occ_12(slottitlepractic),
     occ_1m(slottitlepractic),
     prolog(slott, slottitleprolog),
     occ_1(slottitleprolog),
     occ_12(slottitleprolog),
     occ_1m(slottitleprolog)
     slot_authors(book_501477998, slotau),
     l_sterling(slotau, slotauthorsl_sterling),
     occ_1(slotauthorsl_sterling),
     occ_12(slotauthorsl_sterling),
     occ_1m(slotauthorsl_sterling),
     slot_annotation(book_501477998, slotan),
     l_sterling(book_501477998),
     practic(book_501477998),
     prolog(book_501477998).
```

Fig. 2. First-Order Representation of a Book

The text in each slot is a collection of words (a bag of word, BOW) processed taking into account their occurrences in the original text. Thus, each instance is represented as a vector of BOWs, one for each slot.

Moreover, each instance is labelled with a discrete rating (from 1 to 10) provided by a user, according to his or her degree of interest in the item.

According to the Bayesian approach to classify natural language text documents, given a set of classes $C = \{c_1, c_2, \ldots, c_{|C|}\}$, the conditional probability of a class c_j given a document d is calculated as follows:

$$P(c_j|d) = \frac{P(c_j)}{P(d)} P(d|c_j)$$

In our problem, we have only 2 classes: c_+ represents the positive class (user-likes, corresponding to ratings from 6 to 10), and c_- the negative one (user-dislikes, ratings from 1 to 5). Since instances are represented as a vector of documents, (one for each BOW), and assumed that the probability of each word is independent of the word's context and position, the conditional probability of a category c_j given an instance d_i is computed using the formula:

$$P(c_j|d_i) = \frac{P(c_j)}{P(d_i)} \prod_{m=1}^{|S|} \prod_{k=1}^{|b_{im}|} P(t_k|c_j, s_m)^{n_{kim}} \tag{1}$$

where $S = \{s_1, s_2, \ldots, s_{|S|}\}$ is the set of slots, b_{im} is the BOW in the slot s_m of the instance d_i, n_{kim} is the number of occurrences of the token t_k in b_{im}.

In (1), since for any given document, the prior $P(d_i)$ is a constant, this factor can be ignored if the only interest concerns a ranking rather than a probability estimate. To calculate (1), we only need to estimate the probability terms $P(c_j)$ and $P(t_k|c_j, s_m)$, from the training set, where each instance is weighted according to the user rating r:

$$w_+^i = \frac{r-1}{9}; \qquad w_-^i = 1 - w_+^i \tag{2}$$

The weights in (2) are used for weighting the occurrence of a word in a document. For example, if a word appears n times in a document d_i, it is counted as occurring $n{\cdot}w_+^i$ in a positive example and $n{\cdot}w_-^i$ in a negative example. Weights are used for estimating the two probability terms according to the following equations:

$$\hat{P}(c_j) = \frac{\sum_{i=1}^{|TR|} w_j^i}{|TR|} \tag{3}$$

$$\hat{P}(t_k|c_j, s_m) = \frac{\sum_{i=1}^{|TR|} w_j^i n_{kim}}{\sum_{i=1}^{|TR|} w_j^i |b_{im}|} \tag{4}$$

In (4), n_{kim} is the number of occurrences of the term t_k in the slot s_m of the i^{th} instance, and the denominator denotes the total weighted length of the slot s_m in the class c_j. Therefore, $\hat{P}(t_k|c_j, s_m)$ is calculated as a ratio between the weighted occurrences of the term t_k in slot s_m of class c_j and the total weighted length of the slot.

The final outcome of the learning process is a probabilistic model used to classify a new instance in the class c_+ or c_-. The model can be used to build a personal profile including those words that turn out to be most indicative of the user's preferences, according to the value of the conditional probabilities in (4).

In the specific context of book recommendations, instances in the learning process are the book descriptions. ITR represents each instance as a vector of three BOWs, one BOW for each slot. The slots used are: *title*, *authors* and *textual annotation*. Each book description is analyzed by a simple pattern-matcher that extracts the words, the *tokens* to fill each slot. Tokens are obtained by eliminating stopwords and applying stemming. Instances are used to train the system: occurrences of terms are used to estimates probabilities as described in Equations (3) and (4). An example ITR profile is given in figure 3.

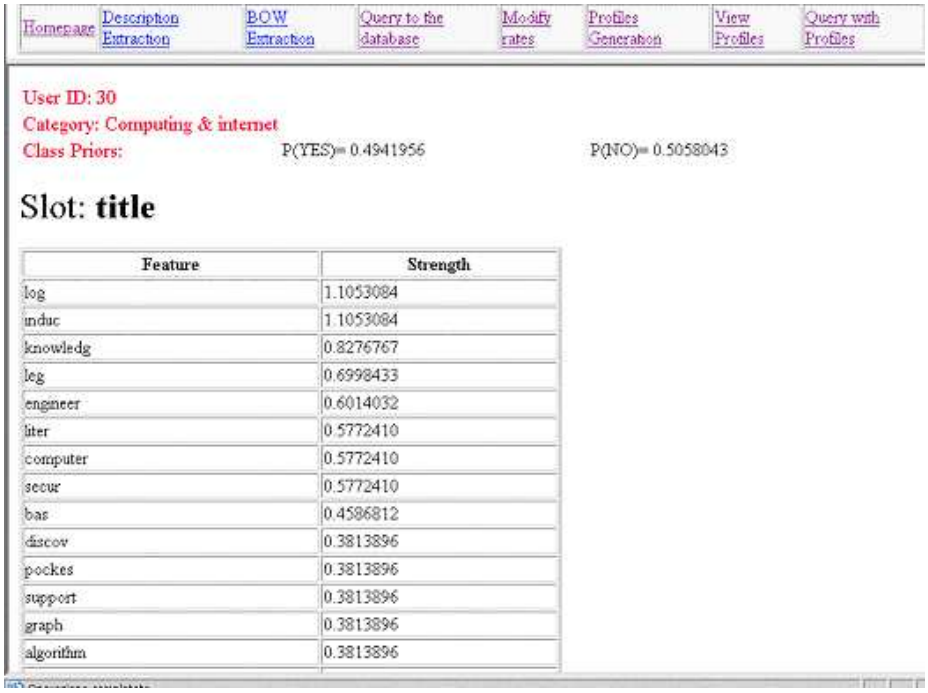

Fig. 3. An example of ITR user profile

4 Experimental Sessions

In this section we describe results from experiments using a collection of textual book descriptions rated by real users according to their tastes. The goal of the experiment has been the comparison of the methods implemented by INTHELEX and ITR in terms of classification accuracy, learning and classification time, when coping with the task of learning user profiles.

The presented experiments are preliminary and should be seen as a baseline study. A new, intensive experimental session will be performed on the EachMovie data set (`http://research.compaq.com/SRC/eachmovie/`), that contains 2811983 numeric ratings (entered by 72916 users) for 1628 different movies.

4.1 Design of the experiments

Eight book categories were selected at the Web site of a virtual bookshop. For each book category, a set of book descriptions was obtained by analyzing Web pages using an automated extractor and stored in a local database. Table 1 describes the extracted information. For each category we considered:

– *Book descriptions* - number of books extracted from the Web site belonging to the specific category;

Table 1. Database information

Category	Book descr.	Books with annotation	Avg. annotation length
Computing & Int.	5378	4178 (77%)	42.35
Fiction & lit.	5857	3347 (57%)	35.71
Travel	3109	1522 (48%)	28.51
Business	5144	3631 (70%)	41.77
SF, horror & fan.	556	433 (77%)	22.49
Art & entert.	1658	1072 (64%)	47.17
Sport & leisure	895	166 (18%)	29.46
History	140	82 (58%)	45.47
Total	22785	14466	

Table 2. Number of books rated by each user in a given category

UserID	**Category**	**Rated books**
37	SF, Horror & Fantasy	40
26	SF, Horror & Fantasy	80
30	Computer & Internet	80
35	Business	80
24c	Computer & Internet	80
36	Fiction & literature	40
24f	Fiction & literature	40
33	Sport & leisure	80
34	Fiction & literature	80
23	Fiction & literature	40

- *Books with annotation* - number of books with a textual annotation (slot annotation not empty);
- *Avg. annotation length* - average length (in words) of the annotations;

Several users have been involved in the experiments: each user were requested to choose one or more categories of interest and to rate 40 or 80 books (in the database) in each selected category, providing 1-10 discrete ratings. In this way, for each user a dataset of 40 or 80 rated books was obtained (see Table 2).

On each dataset a 10-fold cross-validation was run and several metrics were used in the testing phase. In the evaluation phase, the concept of *relevant book* is central. A book in a specific category is considered as relevant by a user if his or her rating is greater than 5. This corresponds in ITR to having $P(c_+|d_i) \geq 0.5$, calculated as in equation (1), where d_i is a book in a specific category. Symmetrically, INTHELEX considers as relevant books covered by the inferred theory. Classification effectiveness is measured in terms of the classical Information Retrieval (IR) notions of *precision* (Pr), *recall* (Re) and *accuracy* (Acc), adapted to the case of text categorization [11]. *Precision* is the proportion of items classified as relevant that are really relevant, and *recall* is the proportion of relevant

Table 3. Performance for ITR and INTHELEX on 10 different users

	Precision		Recall		Accuracy	
UID	ITR	INTHELEX	ITR	INTHELEX	ITR	INTHELEX
37	0,767	0,967	0,883	0,5	0,731	0,695
26	0,818	0,955	0,735	0,645	0,737	0,768
30	0,608	0,583	0,600	0,125	0,587	0,488
35	0,651	0,767	0,800	0,234	0,725	0,662
24c	0,586	0,597	0,867	0,383	0,699	0,599
36	0,783	0,9	0,783	0,3	0,700	0,513
24f	0,785	0,9	0,650	0,35	0,651	0,535
33	0,683	0,75	0,808	0,308	0,730	0,659
34	0,608	0,883	0,490	0,255	0,559	0,564
23	0,500	0,975	0,130	0,9	0,153	0,875
Mean	0,679	0,828	0,675	0,4	0,627	0,636
	(0,699)	(0,811)	(0,735)	(0,344)	(0,68)	(0,609)

Table 4. Learning and Classification times (msec) for ITR and INTHELEX on 10 different users

	Learning Time		Classification Time	
UID	ITR	INTHELEX	ITR	INTHELEX
37	3,738	3931,0	0,851	15,0
26	5,378	8839,0	0,969	20,0
30	8,561	51557,0	1,328	53,0
35	9,289	30338,0	1,423	55,0
24c	7,502	29780,0	1,208	44,0
36	5,051	12317,0	0,894	19,0
24f	4,532	18448,0	0,848	19,0
33	5,820	14482,0	0,961	25,0
34	7,592	73708,0	1,209	42,0
23	4,951	1859,0	0,845	20,0
Mean	6,2414	24525,9	1,0536	31,2

items that are classified as relevant; *accuracy* is the proportion of items that are correctly classified as relevant or not.

As regards training and classification times, we tested the algorithms on a 2.4 GHz Pentium IV running Windows 2000.

4.2 Discussion

Table 3 shows the average precision, recall and accuracy of the models learned in the 10 folds for each user. The last row reports the mean values, averaged on all users. Since the average performance for ITR is very low for user 23, we decided to have a deeper insight into the corresponding training file, and noted that all examples were positive, thus indicating possible noise in the data. This led us to recompute the metrics neglecting this user, thus obtaining the results reported in parentheses.

```
likes(A) :-
  learn(A),
  mach(A),
  intellig(A),
  slot_title(A, F),
  slot_authors(A, G),
  slot_annotation(A, B),
  intellig(B, C),
  learn(B, D),
  occ_12(D),
  mach(B, E),
  OCC_12(E).
```

Fig. 4. Rule learned by INTHELEX

In general, INTHELEX provides some performance improvement over ITR. In particular, it can be noticed that INTHELEX produces very high precision even on the category "SF, horror & fantasy", taking into account the shortness of the annotations provided for books belonging to this category. This result is obtained both for user 26, who rated 80 books, and for user 37, who rated only 40 books. Moreover, classification accuracy obtained by INTHELEX is slightly better than the one reached by ITR. On the other hand, ITR yields a better recall than INTHELEX for all users except one (user 23).

For pairwise comparison of the two methods, the nonparametric Wilcoxon signed rank test was used [9], since the number of independent trials (i.e., users) is relatively low and does not justify the application of a parametric test, such as the t-test. In this experiment, the test was adopted in order to evaluate the difference in effectiveness of the profiles induced by the two systems according to the metrics pointed out in Table 3. Requiring a significance level $p < 0.05$, the test revealed that there is a statistically significant difference in performance both for Precision (in favor of INTHELEX) and for Recall (in favor of ITR), but not as regards Accuracy.

Going into more detail, as already stated, ITR performed very poorly only on user 23, whose interests turned out to be very complex to be captured by the probabilistic approach. Actually, all but one rates given by such a user were positive (ranging between 6 and 8), that could be the reason for such a behaviour. With respect to the complete dataset of all users, the accuracy calculated on the subset of all users except user 23 becomes statistically significant in favor of ITR.

Table 4 reports the results about training and classification time of both systems. Training times vary substantially across the two methods. ITR takes an average of 6,2414 msec to train a classifier for a user when averaged over all 10 users. Training INTHELEX takes more time than ITR, but this is not a real problem because profiles can be learnt by batch processes without noise for users. In user profiling application, it is important to quickly classify new instances,

Table 5. Interests of user 39 and user 40 in books belonging to the category "Computing and Internet"

UserID	Interests
39	machine learning, data mining, artificial intelligence
40	web programming, XML, databases, e-commerce

for example to provide users with on-line recommendations. Both methods are very fast in this regard.

In summary, the probabilistic approach seems to have better recall, thus showing a trend to classify unseen instances as positive; on the contrary, the first-order approach tends to adopt a more cautious behavior, and classify new instances as negative. Such a difference is probably due to the approach adopted: learning in INTHELEX is data-driven, thus it works bottom-up and keeps in the induced definitions as much information as possible from the examples. This way, requirements for new observations in order to be classified as positive are more demanding, and few of them pass; on the other hand, this ensures that those that fulfill the condition are actually positive instances.

Another remark worth noting is that theories learned by the symbolic system are very interesting from a human understandability viewpoint, in order to be able to explain and justify the recommendations provided by the system. Figure 4 shows one such rule, to be interpreted as "the user likes a book if its annotation contains stems *intellig*, *learn* (1 or 2 times) and *mach* (1 or 2 times)". Anybody can easily understand that this user is interested in books concerning artificial intelligence and, specifically, machine learning.

The probabilistic approach could be used in developing recommender systems exploiting the *ranked list* approach for presenting items to the users. In this scheme, users specifies their needs in a form and the system presents a usually long list of results, ordered by their predicted relevance (the probability of belonging to the class). On the other hand, the ILP approach could be adopted in situations when the system transparency is a critical factor and it is important to provide an explanation of why a recommendation was made.

From what said above, it seems that the two approaches compared in this paper have complementary *pros and cons*, not only as regards the representation language, but also as concerns the predictive performances. This naturally leads to think that some cooperation could take place between the two in order to reach higher effectiveness of the recommendations. For instance, since the probabilistic theories have a better recall, they could be used for selecting which items are to be presented to the user. Then, some kind of filtering could be applied on them, in order to present to the user first those items that are considered positive by the symbolic theories, that are characterized by a better precision.

Fig. 5. Profile of user 39.

5 Exploiting profiles to personalize recommendations

In this section, we present an example of how the learned profiles can be exploited to provide Web users with personalized recommendations, delivered using the ranked list approach. In particular, we analyze a usage scenario of the ITR system, in which two users with different interests in books belonging to the category "Computing and Internet" submit the same query to the ITR search engine. Table 5 reports the explicit interests of the two users. Figure 5 and 6 depict the profiles of both users inferred by ITR, and shows some keywords in the slot title, which are indicative of user preferences. When a user submits a query q, the books b_i in the result set R_q are ranked by the classification value $P(c_+|b_i)$, $b_i \in R_q$, computed according to Equation (1). The exact posterior probabilities are determined by normalizing $P(c_+|b_i)$ and $P(c_-|b_i)$, so that their sum is equal to 1. The result set retrieved by ITR in response to the query q= *"programming"*, submitted by user 39, is presented in Figure 7. The first book displayed is *"Expert Systems in Finance and Accounting"*, in accordance with the interests contained in the user profile. In fact, the profile of user 39 contains, in the slot title, stemmed keywords (*"intellig"*, *"artific"*, *"system"*) that reveal the interest of the user in systems exploiting artificial intelligence methods, like expert systems. Conversely, if another user submits the same query, the books in the result set are ranked in a different way, due to the fact that this user has a different profile (user 40 in Figure 6). In this case, the system recommends *"Java Professional Library"* (the first book in the ranked list) (Figure 8), because the stemmed keywords (*"java"*, *"databas"*, *"xml"*, *"program"*, *"jdbc"*) in the slot title of the profile indicate well known technologies for web developers. Again, the

Fig. 6. Profile of user 40.

advice provided by the systems seems to be indicative of the interests supplied by the users.

These scenarios highlight the effect of the personalization on the search process, that is the dependence of the result set on the profile of the user who issued the query. Although the query personalization scenarios presented here suggest a use of the probabilistic profiles for content-based filtering of the search results, thus adopting a passive recommendation strategy, they can be used also for active recommendation. For example, the profile of a user could be used to identify a set of N items that will be of interest to the user in each category of the catalogue (top-N recommendation problem) [12]. Then, the N top-scored items in a category could be recommended when the user is browsing items in that category. As regards the rule-based profiles, since they do not provide a recommendation score, but only a binary judgement (likes/dislikes), they are more suitable for refining the recommendations from among a candidate set, such as a ranked list. To sum up, although this has been designed as a baseline study, it is worth drawing attention to the key finding highlighted by the study: ILP and probabilistic techniques are complementary for the task of learning user profiles from text and could be combined for active or passive recommendation. In our opinion, a cascade hybridization method [1] is the best way to integrate the two approaches. In this technique, the probabilistic profile of a user is exploited first to produce a coarse ranking of candidates, and then the symbolic profile refines the recommendations from among the candidate set, also explaining and justifying the recommendations provided by the system.

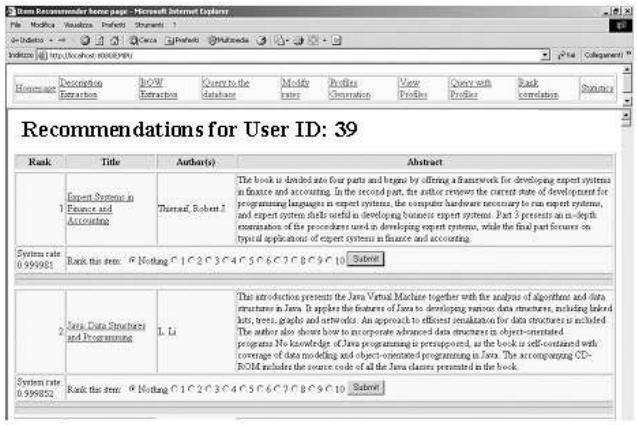

Fig. 7. Books recommended by ITR to user 39, who issued the query "programming".

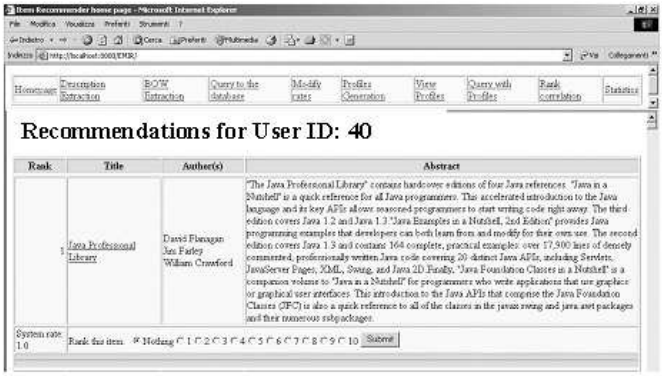

Fig. 8. Books recommended by ITR to user 40, who issued the query "programming".

6 Conclusions

Research presented in this paper has focused on methods for learning user profiles which are predictively accurate and comprehensible. Specifically, an intensive comparison between an ILP and a probabilistic approach to learning models of users' preferences was carried out. Experimental results highlight the usefulness and drawbacks of each one, that can suggest possible ways of combining the two approaches in order to offer better support to users accessing e-commerce virtual shops or other information sources. In particular, we suggest a simple possible way of obtaining a cascade hybrid method. In this technique, the probabilistic approach could be employed first to produce a coarse ranking of candidates and the ILP approach could be used to refine the recommendations from among the candidate set.

Currently we are working on the integration in INTHELEX of techniques able to manage numeric values, in order to treat in a more efficient way numerical features of instances, and hence to obtain theories with a more fine grain size.

References

[1] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[2] M. Degemmis, P. Lops, G. Semeraro, and F. Abbattista. Extraction of user profiles by discovering preferences through machine learning. In M. A. Klopotek, S. T. Wierzhon, and K. Trojanowski, editors, *Information Systems: New Trends in Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 69–78. Springer, 2003.

[3] F. Esposito, G. Semeraro, N. Fanizzi, and S. Ferilli. Multistrategy Theory Revision: Induction and abduction in INTHELEX. *Machine Learning*, 38(1/2):133–156, 2000.

[4] D.D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[5] D.D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, US, 1994.

[6] D. Mladenic. Text-learning and related intelligent agents: a survey. *IEEE Intelligent Systems*, 14(4):44–54, 1999.

[7] R.J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the 5^{th} ACM Conference on Digital Libraries*, pages 195–204, San Antonio, US, 2000. ACM Press, New York, US.

[8] I. Moulinier and J.G. Ganascia. Confronting an existing machine learning algorithm to the text categorization task. In *IJCAI, Workshop on New approaches to Learning for Natural Language Processing*, Montral, 1995.

[9] M. Orkin and R. Drogin. *Vital Statistics*. McGraw-Hill, New York, 1990.

[10] M. Pazzani and D. Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27(3):313–331, 1997.

[11] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

[12] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology*. East West University, Bangladesh, 2002.

[13] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[14] 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.