Symbolic and Probabilistic Techniques for Learning User Profiles

T. Basile, M. Degemmis, N. Di Mauro, F. Esposito & S. Ferilli
Dipartimento di Informatica, Università di Bari

ABSTRACT: In the era of Internet, huge amounts of data are available to everybody, in every place and at any moment. As more information becomes available, it becomes increasingly difficult to search for relevant information: the main challenge is to support Web users in order to improve searching among extremely large Web repositories, such as online product catalogues or other generic information sources. Building systems for assisting users in finding relevant information is often complicated by the difficulty in articulating user interests in a form that can be used for searching. Machine learning methods offer a promising approach to solve this problem. Our research focuses on methods for learning user profiles which are predictively accurate and comprehensible. In this paper we present a comparison between an ILP and a probabilistic approach to learning models of users' preferences. Experimental results highlight the usefulness and drawbacks of each one.

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 complexity the problem could be lowered by the automatic construction of machine processable profiles that can be exploited to deliver to the user *personalized* content, 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 (Mladenic 1999).

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.

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. Our experiments evaluated the effects of the two above mentioned 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: we first describe the INTHELEX ILP system and its main features. We next introduce Item Recommender, the system that implements a statistical

learning process to induce profiles from text. Then a detailed description of the experiments is given. Finally, we analyze the results of the experiments by means of a statistical test and we draw some general conclusions.

2 INTHELEX

INTHELEX (INcremental THEory Learner from EXamples) 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 (Semeraro et al. 1998) 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. 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.

A set of examples of the concepts to be learned is provided by an *Expert*, possibly selected from 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 whole set of examples can be subdivided into *tuning* and *test* examples, according to the way in which examples are exploited during the learning process. Specifically, tuning examples, previously classified by the Expert, are exploited to build/refine a theory that is able to explain them. An initial theory can also be provided by the Expert. Subsequently, such a theory, plus the Background Knowledge (if any), are checked against test examples and, in case of incorrectness, the cause of the wrong decision can be located. Test examples are exploited only to check the predictive capabilities of the theory on new observations. Conversely, tuning examples are exploited incrementally to modify incorrect hypotheses according to a data-driven strategy. In particular, when a positive example is not covered, a revised theory is obtained in one of the following ways (listed by decreasing priority) such that completeness is restored:

- 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.

When, on the other hand, 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.

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 desiberable feature in order to understand the user preferences. Second, incrementality is an unnegligible 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.

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.

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)`, indicating that 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(slott, slottitleprolog)` indicates that the object `slottitleprolog` has stem "prolog" and is contained in slot `slott`; in such a case, also a literal `prolog(book)` is present to say that stem "prolog" is present in the book description.

```
likes(501477998) :-
 slot_title(501477998, slott),
  practic(slott, slottitlepractic),
   occ_1(slottitlepractic),
   occ_12(slottitlepractic),
  prolog(slott, slottitleprolog),
   occ_1(slottitleprolog),
   occ_12(slottitleprolog),
 slot_authors(501477998, slotau),
  l_sterling(slotau, slotauthorsl_sterling),
   occ_1(slotauthorsl_sterling),
   occ_12(slotauthorsl_sterling),
 slot_annotation(501477998, slotan),
 l_sterling(501477998),
 practic(501477998),
 prolog(501477998).
```

Figure 1: First-Order Representation of a Book

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 1 shows an example for the class `likes`.

3 ITEM RECOMMENDER

ITR (ITem Recommender) 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 (Mitchell 1997). Naïve Bayes has been shown to perform competitively with more complex algorithms and has become an increasingly popular algorithm in text classification applications (Pazzani and Billsus 1997; Mooney and Roy 2000).

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 rep-

resented by a set of *slots*. Each slot is a textual field corresponding to a specific feature of an item.

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:

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

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 all that is desired is 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*:

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

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:

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

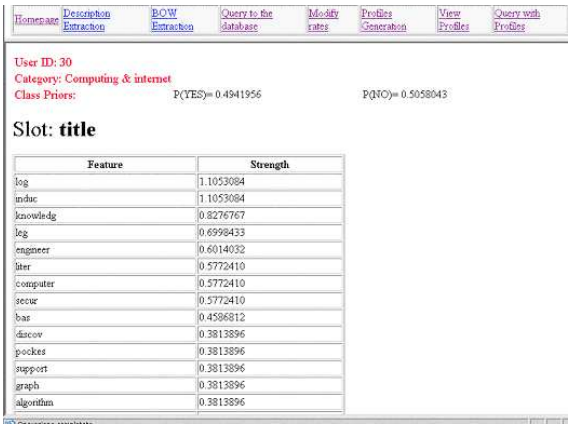

Figure 2: An example of ITR user profile

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

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 3 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: cccurrences of terms are used to estimates probabilities as described in Equations (3) and (4). An example ITR profile is given in figure 2.

4 EXPERIMENTAL SESSIONS

The goal of the experiment was to measure the accuracy of the two different types of user profiles obtained by the two already described methods.

4.1 *Design of the experiments*

Nine book categories were selected at the Web site of a virtual bookshop. For each book category, a set

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 1: Database information

UserID	Category	Rated book
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

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

of book descriptions was obtained by analyzing Web pages using an automated extractor and stored in a local database. Table 4.1 describes the extracted information. For each category we considered:

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

UID	Precision		Recall		Accuracy	
	A1	A2	A1	A2	A1	A2
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 3: Precision, Recall and Accuracy for ITR (A1) and INTHELEX (A2) by a 10-fold cross validation on 10 different users

a specific category. Simmetrically, 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 (Salton and McGill 1983).

4.2 *Discussion*

Table 3 shows the average precision, recall and accuracy of the models learned in the 10 folds for each user. The row headed *Mean* reports the mean values, averaged on all users. For pairwise comparison of the two methods, the nonparametric Wilcoxon signed rank test was used (Orkin and Drogin 1990), 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, it is possible to note that 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 rates given by such a user but one were positive (ranging between 6 and 8), that could be the reason for such a behaviour. This led us to recompute the metrics neglecting this user, thus obtaining the results reported in the last row of Table 3. With respect to the complete dataset of all users, this causes the Accuracy to become statistically significant in favor of ITR, as well.

In summary, the probabilistic approach seems to have better recall, thus showing a trend to classify un-

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

Figure 3: Rule learned by INTHELEX

seen 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 3 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.

In a commercial Web site perspective, the probabilistic behavior should be preferable. It 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. 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.

5 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 integrating the two approaches in order to offer better support to users accessing e-commerce virtual shops or other information sources.

REFERENCES

Mitchell, T. (1997). *Machine Learning*. New York: McGraw-Hill.

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

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

Orkin, M. and R. Drogin (1990). *Vital Statistics*. New York: McGraw-Hill.

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

Salton, G. and M. McGill (1983). *Introduction to Modern Information Retrieval*. New York: McGraw-Hill.

Semeraro, G., F. Esposito, D. Malerba, N. Fanizzi, and S. Ferilli (1998). A logic framework for the incremental inductive synthesis of datalog theories. In N. E. Fuchs (Ed.), *Logic Program Synthesis and Transformation*, Number 1463 in Lecture Notes in Computer Science, pp. 300–321. Springer-Verlag.