User Model User-Adap Inter (2007) 17:217–255
DOI 10.1007/s11257-006-9023-4

ORIGINAL PAPER

A content-collaborative recommender that exploits WordNet-based user profiles for neighborhood formation

Marco Degemmis · Pasquale Lops · Giovanni Semeraro

Received: 31 October 2005 / Accepted in revised form: 26 September 2006 /
Published online: 22 March 2007

Abstract Collaborative and content-based filtering are the recommendation techniques most widely adopted to date. Traditional collaborative approaches compute a similarity value between the current user and each other user by taking into account their rating style, that is the set of ratings given on the *same* items. Based on the ratings of the most similar users, commonly referred to as *neighbors*, collaborative algorithms compute recommendations for the current user. The problem with this approach is that the similarity value is only computable if users have common rated items. The main contribution of this work is a possible solution to overcome this limitation. We propose a new content-collaborative hybrid recommender which computes similarities between users relying on their content-based profiles, in which user preferences are stored, instead of comparing their rating styles. In more detail, user profiles are clustered to discover current user neighbors. Content-based user profiles play a key role in the proposed hybrid recommender. Traditional keyword-based approaches to user profiling are unable to capture the semantics of user interests. A distinctive feature of our work is the integration of linguistic knowledge in the process of learning *semantic* user profiles representing user interests in a more effective way, compared to classical keyword-based profiles, due to a sense-based indexing. Semantic profiles are obtained by integrating machine learning algorithms for text categorization, namely a naïve Bayes approach and a relevance feedback method, with a word sense disambiguation strategy based exclusively on the lexical knowledge stored in the WordNet lexical database. Experiments carried out on a content-based extension of the EachMovie dataset show an improvement of the accuracy of sense-based profiles with respect to keyword-based ones, when coping with the task of classifying movies as interesting

M. Degemmis (✉) · P. Lops · G. Semeraro
Department of Informatics, University of Bari, Via E, Orabona, 4-70126, Bari, Italy
e-mail: degemmis@di.uniba.it

P. Lops
e-mail: lops@di.uniba.it

G. Semeraro
e-mail: semeraro@di.uniba.it

(or not) for the current user. An experimental session has been also performed in order to evaluate the proposed hybrid recommender system. The results highlight the improvement in the predictive accuracy of collaborative recommendations obtained by selecting like-minded users according to user profiles.

Keywords User modeling · Collaborative filtering · Content-based filtering · Hybrid recommenders · Machine learning · Neighborhood formation in recommender systems · WordNet

1 Introduction

Internet is open to anyone who can access it and everyday there are tons of information generated online. The Web search environment, however, is not ideal. The existence of a large quantity of information, in combination with the dynamic and heterogeneous nature of the Web, makes retrieval a hard task for the average user. As a consequence, the requirement of the development of techniques able to help users navigate large information spaces or search the Web to satisfy their information needs, is more than evident. Web personalization is the process of customizing a Web site to the needs and preferences of specific users, taking into account the knowledge acquired from the analysis of the users' navigational behavior, in correlation with other information collected in the Web context, such as web pages' content. Nowadays many web sites embody recommender systems as a way of personalizing their content for users (Resnick and Varian 1997). Recommender systems have the effect of guiding users in a personalized way to interesting or useful objects in a large space of possible options (Burke 2002). Recommendation algorithms use input about a customer's interests to generate a list of recommended items. At Amazon.com, recommendation algorithms are used to personalize the online store for each customer, for example showing programming titles to a software engineer and baby toys to a new mother (Linden et al. 2003). Among different recommendation techniques that have already been put forward in studies on this matter, the *content-based* and the *collaborative filtering* approaches are the most widely adopted to date.

Systems implementing the content-based recommendation approach analyze a set of documents, usually textual descriptions of the items previously rated by an individual user, and build a model or profile of user interests based on the features of the objects rated by that user (Mladenic 1999). The profile is exploited to recommend new items of interest.

Collaborative recommenders differ from content-based ones in that user opinions are used instead of content. They gather user ratings about objects and store them in a centralized or distributed database. To provide user X with recommendations, the system computes the *neighborhood* of that user, i.e. the subset of users that have a taste similar to X. Similarity in taste is computed using the similarity of ratings for objects that were rated by both users. The system then recommends objects that users in X's neighborhood indicated to like, provided that they have not yet been rated by X.

Each type of filtering methods has its own weaknesses and strengths (Shardanand and Maes 1995; Balabanovic and Shoham 1997; Lee 2001). The main advantage of collaborative filtering over content-based methods is that any item can be recommended regardless of its content. Movies, images, art and text items are all represented by the users' opinions and thus can be recommended by the same system. This also means that collaborative filtering can recommend things from different genres. Lets assume that there is a user who likes thrillers but not science fiction. In a content-based approach the user will only get thrillers recommended, whereas in a collaborative approach, a science fiction movie can be recommended as well, if there are enough science fiction lovers that also like thrillers.

Terveen and Hill (2001) claim three essentials are needed to support collaborative filtering: Many people must participate (increasing the likelihood that any one person will find other users with similar preferences); there must be an easy way to represent user interests in the system; the algorithms must be able to match people with similar interests. These three elements are not that easy to develop, and produce the main shortcomings of collaborative filtering systems (Balabanovic and Shoham 1997; Lee 2001):

- Sparsity problem. The number of ratings obtained from users is usually very small compared to the number of ratings that must be predicted. Sparsity mainly has a negative effect on the predictions for the active user because it affects the selection of the neighbors. If the similarity between users is only determined using the ratings given to co-rated items, the computation becomes harder in case of extremely sparse user-item matrix. Thus, adopting a better strategy to find correlated users, even if they did not rate the same items, could have a positive effect on predictions.
- Scalability problem. Collaborative filtering systems require an increasing amount of computational resources as the number of users and items grows.
- Lack of transparency problem. Collaborative systems today are *black boxes* which give advice but cannot be questioned. However, this is not just a problem in collaborative filtering, but it is particularly evident in this kind of systems because collaborative predictions may be harder to explain than predictions made by some content-based models (Herlocker et al. 2000).

These problems have limited acceptance of collaborative systems in all but low-risk content domains since they are untrustworthy for high-risk ones. Many researchers have tried to combine different techniques in different ways in order to obtain *hybrid recommender systems*. Most frequently, collaborative filtering is combined with content-based filtering. The key challenge is how to combine the two types of filters to compensate the drawbacks of each single approach. In this paper we propose a hybrid recommender that tries to address the shortcomings of collaborative filtering systems listed above.

1.1 Motivations and contributions

Although collaborative filtering is the most fully explored technique, a number of hybrids based on this technique remains unexplored. According to Burke's classification of hybrid recommender systems (Burke 2002), no proposal of *content-based/collaborative feature augmentation hybrid* can be found in literature. As he pointed out, augmentation is attractive because it offers a way to improve the performance of a core recommendation technique without modifying it. Additional functionality is added by intermediaries that can use other techniques to augment the data itself. Augmentation is realized by sequencing two recommenders: The features used by the second recommender include the output of the first one.

Our idea is that a potential improvement of collaborative recommendations may come from new strategies for neighborhood formation that overcome the aforementioned limitations due to the classic and widely adopted Pearson's correlation coefficient (Herlocker et al. 1999). If the neighborhood formation process selects people with similar tastes, the chances are great that the items that are highly evaluated by that group will also be appreciated by the advice-seeker. According to this idea, the main contribution of this paper is the design of a *content-based/collaborative feature augmentation hybrid*. A system like this exploits user profiles learned by a content-based recommender to improve the *neighborhood formation* step within the process of producing collaborative recommendations through a nearest-neighbor (a kind of memory-based) algorithm.

In the discussion of the proposed hybrid technique, we will take into account different issues:

– *Combination of content and collaborative recommendation techniques*. What are the main advantages of combining these techniques into a feature augmentation hybrid? Which is the scale of improvement over pure collaborative filtering?
– *Integration of knowledge about the user interests for driving the process of producing recommendations*. Which content-based technique should be adopted to produce user profiles that are useful for the neighborhood formation process? How could the concept of *similarity between users* be extended by using the knowledge stored in the profiles instead of ratings in order to compute neighbors?
– *Use of semantic approaches in learning user profiles*. Is it sufficient to represent user interests by keywords or is a semantic approach based on *concepts* more accurate? How does this decision affect the performance of the hybrid system?

The contribution of the work can be summarized as follows:

– Presentation of a new *content-based/collaborative feature augmentation hybrid* recommender system not yet explored in the literature. Section 3 describes the main features of the hybrid recommender. In particular, Sect. 3.3 highlights the advantages of the proposed approach when facing the problems of sparsity, scalability and lack of transparency. Results of a detailed experimental session (Sect. 3.2) show that the hybrid approach outperforms the pure collaborative approach (7% MAE reduction on average). A case in which the effectiveness of the proposed method is reduced is also presented.
– Definition of a neighborhood formation process which exploits user profiles, inferred through content-based methods, to group users having similar preferences. Profiles are built by a relevance feedback approach able to learn both interests and *dis*interests of users. The choice of this approach is motivated by an experimental evaluation described in Sect. 2.5 that aims at comparing this learning strategy with the Bayesian learning approach described in Sect. 2.4. We will show in Sect. 3.1 how similarity between users is computed by a clustering algorithm that groups users according to their profiles. The concept of user similarity is extended so that the constraint of considering users dissimilar if they did not rate common items is removed. The main advantage compared to Pearson's correlation coefficient is that users might be considered *similar* not only if they like or dislike the *same* items, but also if they like or dislike *similar* ones, according to the content descriptions of the items (e.g. movies directed by the same director, or with similar plot or with the same star in the cast), due to the form of inferred user profiles (Sect. 2.3.1).
– Presentation of a context disambiguation method based on general linguistic knowledge that produces sense-based (concept-based) profiles representing user interests

in a more effective way, in opposition to classical keyword-based profiles. Experiments reported in Sect. 2.5 show that migrating from words to concepts produces a classification accuracy improvement for both content-based methods presented in the paper (+2% of $F1$ improvement for the relevance feedback approach, +8% for the probabilistic approach). Finally, we will show that by integrating sense-based profiles in the proposed hybrid system, the predictive accuracy of collaborative recommendations can be increased.

1.2 Related work

There have been several attempts to combine content information with collaborative filtering in order to obtain hybrid recommender systems, and there are different ways to do that (Adomavicius and Tuzhilin 2005). A possibility is to implement separate methods and combine the predictions, as in the *P-Tango* system (Claypool et al. 1999). It initially gives equal weights to both recommenders, but gradually adjusts the weightings as predictions about user ratings are confirmed or not. The system keeps the two filtering approaches separate and this allows to benefit from individual advantages. The implicit assumption is that the relative value of the different techniques is more or less uniform across the space of possible items. We know that is not always so—for example, a collaborative recommender will be weaker for those items with a small number of raters.

Another strategy is to incorporate content-based characteristics into a collaborative approach (and vice versa). Pazzani (1999) proposes the *collaboration via content*, that uses a prediction scheme similar to the standard collaborative filtering, in which similarity among users is not computed on provided ratings, but rather on the content-based profile of each user. The underlying intuition is that like-minded users are likely to have similar content-based models, and that this similarity relation can be detected without requiring overlapping ratings. The main limitation of Pazzani's approach is that the similarity of users is computed using Pearson's correlation coefficient between content-based weight vectors.

Melville et al. (2002) propose the *Content-Boosted Collaborative Filtering* approach, that exploits a content-based predictor to enhance existing user data and then provides personalized suggestions through collaborative filtering. The content-based predictor is applied to each row of the initial user-item matrix, corresponding to each user, and gradually generates a pseudo user-item matrix that is a full dense matrix used for performing collaborative filtering. The prediction task is treated as a text categorization problem.

Soboroff and Nicholas (1999) use latent semantic indexing to create a collaborative view of a collection of user profiles represented as term vectors.

LIBRA (Mooney and Roy 2000) makes content-based recommendations of books based on data found in Amazon.com, using a naïve Bayes text classifier. The text data used by the system includes *related authors* and *related titles*, that Amazon generates using collaborative algorithms. These features were found to make a significant contribution to the quality of recommendations.

Other approaches construct a general unifying model that incorporates content-based and collaborative characteristics, as in (Basu et al. 1998), where the authors propose the use of content-based and collaborative characteristics (e.g., the age or gender of users or the genre of movies) in a single rule-based classifier.

The strategy we adopted to design our hybrid recommender is inspired by the work of authors that combine collaborative and content-based approaches by (1) learning and maintaining user profiles based on content analysis using different techniques, and (2) directly comparing the resulting profiles to determine similar users in order to make collaborative recommendations (Adomavicius et al. 2005).

The choice of an effective content-based method is crucial in designing an efficient content-collaborative hybrid recommender. Our work was mainly inspired by:

- *Syskill & Webert* (Pazzani and Billsus 1997), that suggests learning user profiles as Bayesian classifiers;
- *ifWeb* (Asnicar and Tasso 1997), that supports users in document searching by maintaining user profiles which store both interests and explicit *dis*interests;
- *SiteIF* (Magnini and Strapparava 2001), which exploits a sense-based representation to build a user profile as a semantic network whose nodes represent senses of the words in documents requested by the user;
- *Fab* (Balabanovic and Shoham 1997), which adopts a Rocchio (1971) relevance feedback method to create and update user personal models (selection agents) that are directly compared to determine similar users for collaborative recommendations.

According to these successful works, we conceived our content-based systems as text classifiers able (1) to deal with a sense-based document representation and (2) to distinguish between interests and *dis*interests of users. The strategy we propose to shift from a keyword-based document representation to a sense-based one is *to integrate lexical knowledge in the indexing step of training documents*.

Several methods have been proposed to accomplish this task. In (Rodriguez et al. 1997), WordNet is used to enhance neural network learning algorithms. This approach makes use of synonymy alone and involves a manual word sense disambiguation (WSD) step, whereas our work exploits both synonymy and hypernymy and is completely automatic.

Scott and Matwin (1998) suggested including WordNet information at the feature level by expanding each word in the training set with *all* the synonyms for it in WordNet, including those available for each sense, in order to avoid a WSD process. This approach has shown a decrease of effectiveness in the obtained classifier, mostly due to the word ambiguity problem. The work by Scott and Matwin suggests that some kind of disambiguation is required.

More recent works tried to investigate whether embedding WSD in document classification tasks improves classification accuracy.

Hotho et al. (2003) used WordNet-based WSD and feature weighting to achieve improvements of clustering results: They showed positive effects when background knowledge stored in WordNet is included into text clustering.

Bloedhorn and Hotho (2004) compared three strategies to map words to senses: No WSD, most frequent sense as provided by WordNet, WSD based on context. They found positive results on the Reuters 25178, the OSHUMED and the FAODOC corpora. The approach presented in (Rosso et al. 2004) shows that the use of a supervised WSD algorithm slightly improves the error-percentage of a k-NN classifier. In (Theobald et al. 2004), a WSD algorithm based on the general concept of Extended Gloss Overlaps is used and classification is performed by a Support Vector Machine classifier applied to the two largest categories of the Reuters 25178 corpus and two Internet Movie Database movie genres.[1] The relevant outcome of this work is that, when the training set is small, the use of WordNet senses combined with words improves the performance of the classifier. Also in a more recent work (Mavroeidis et al. 2005), the authors provided a sound experimental evidence of the quality of their approach for embedding WSD in classification tasks, especially when the training sets are small.

2 WordNet-based user profiles

Due to the impressive amount of the available text data, there has been a growing interest in augmenting traditional information filtering and retrieval approaches with machine learning and text mining techniques, that induce a structured model of the interests of a user, the *user profile*, from text documents (Mladenic 1999). These methods typically require users to label documents by assigning a relevance score, and automatically infer profiles exploited in the filtering/retrieval process to rank documents according to the user preferences. Some information access scenarios cannot be solved through straightforward matching of queries and documents represented by keywords. For example, a user who wants to retrieve "interesting news stories" cannot easily express this form of information need as a query suitable for search engines. In order to find relevant information in these problematic information scenarios, a possible solution could be to develop methods for discovering concepts that characterize documents the user has already labeled as relevant. Traditional keyword-based approaches are unable to capture the *semantics* of the user interests. They are primarily driven by a string-matching operation: If a string, or some morphological variant, is found in both the profile and the document, a match is made and the document is considered as relevant. String matching suffers from problems of:

- POLYSEMY, the presence of multiple meanings for one word (e.g. the noun "Bat" as a nocturnal mouselike mammal, or squash racket);
- SYNONYMY, multiple words have the same meaning (e.g. the verbs "make", "manufacture" and "produce" all refer to the production of items).

The result is that, due to synonymy, relevant information can be missed if the profile does not contain the exact keywords in the documents while, due to polysemy, wrong documents could be deemed relevant. These problems call for alternative methods able to learn more accurate profiles that capture concepts expressing user interests from relevant documents. These *semantic* profiles will contain references to concepts defined in lexicons or ontologies. Although they clearly require additional knowledge and processing, methods for learning *semantic* profiles have potentially a number of advantages: For example, if a user likes documents about *robotics* and *machine learning*, a method with the ability to identify these concepts and to have access to the proper concept hierarchy could infer that the user is interested in *artificial intelligence*. Not only this would be a natural suggestion to the user, but it might also be useful in quickly capturing his/her real preferences and suggesting what additional information might be of interest. Moreover, the descriptions of the identified key concepts could help make the profile more intelligible to the user, which in turn could help establish trust. We propose a strategy integrating a WSD algorithm based on WordNet (Miller

[1] IMDb, http://www.imdb.com

1990; Fellbaum 1998) with both a relevance feedback and a naïve Bayes method to induce *semantic user profiles* (Degemmis 2005). Machine learning techniques, generally used in the task of inducing content-based profiles, are those that are well-suited for text categorization (Sebastiani 2002). In the machine learning approach to text categorization, an inductive process automatically builds a text classifier by learning from a set of *training documents* (documents labeled with the categories they belong to), the features of the categories. We consider the problem of learning user profiles as a binary text categorization task: Each document has to be classified as interesting or not with respect to the user preferences. Therefore, the set of categories is $C = \{c_+, c_-\}$, where c_+ is the positive class (user-likes) and c_- the negative one (user-dislikes). We present a method able to learn profiles for content-based filtering. The accuracy of the keyword-based profiles inferred by this method will be compared to semantic user profiles obtained by the same method, but exploiting an indexing procedure based on WordNet.

2.1 Document representation: words and meanings

In the case of text categorization, the selection of appropriate document features is usually referred to as document representation. Many document representations have appeared in previous studies (Yang and Pedersen 1997) and most of them are based on the use of the words occurring in a document. In the classical *bag of words* (BOW) model, each feature used to represent a document corresponds to a single word found in the document. Such a representation ignores important aspects of a document, such as the structure.

We adopt a document representation that can be exploited as a starting point for building a more accurate profile of a user's interests, that we call *semantic user profile* because it is based on the senses of the words found in the training documents. Here "word sense" is used as a synonym of "word meaning". Word meanings provide more information about the content of a document than words themselves. The filtering phase could take advantage of the word senses to recommend new items (documents) with high semantic relevance compared to the user profile. There are two crucial issues to address: First, a repository for word senses has to be identified. Second, any implementation of a sense-based text classifier must solve the problem that, while words occur in a document meanings do not, since they are often hidden in the context. Therefore, a procedure is needed for assigning senses to words: The task of WSD consists in determining which of the senses of an ambiguous word is invoked in a particular use of the word (Manning and Schütze 1999).

As for sense repository, we have adopted WordNet (version 1.7.1), a large lexical database for English, which is freely available online[2] and has been extensively used in NLP research (Stevenson 2003). WordNet was designed to establish connections between four types of Parts of Speech (POS): Noun, verb, adjective, and adverb. The basic building block for WordNet is the SYNSET (SYNONYM SET), which represents a specific meaning of a word. The specific meaning of one word under one type of POS is called a sense. Synsets are equivalent to senses, which are structures containing sets of words with synonymous meanings (words that are interchangeable in some contexts). Each synset has a gloss, a short textual description that defines the concept represented by the synset. For example, the words *night*, *nighttime* and *dark* constitute

[2] http://wordnet.princeton.edu

Table 1 The WordNet Lexical Matrix: F_1 and F_2 are synonyms; F_2 is polysemous

Word Meanings	Word Forms			
	F_1	F_2	...	F_n
M_1	$E_{1,1}$	$E_{1,2}$		
M_2		$E_{2,2}$		
...				
M_m				$E_{m,n}$

a single synset that has the following gloss: "the time after sunset and before sunrise while it is dark outside".

The WordNet lexical matrix (Table 1) describes the mapping between forms and meanings. Word forms are imagined to be listed as headings for the columns, word meanings as headings for the rows. An entry in a cell of the matrix implies that the form in that column can be used (in an appropriate context) to express the meaning in that row. Thus, entry $E_{1,1}$ implies that word form F_1 can be used to express word meaning M_1. If there are two entries in the same column, the word form is polysemous; if there are two entries in the same row, the two word forms are synonyms (relative to a context).

The word meaning M_1 in Table 1 can be represented by simply listing the word forms that can be used to express it: $\{F_1, F_2, ...\}$ (here and later, the curly brackets, '{' and '}', surround the sets of synonyms that are used to identify definitions of lexicalized concepts). Synsets are connected through a series of relations: Antonymy (opposites), hyponymy/hypernymy (IS-A), meronymy (PART-OF), etc. We addressed the WSD problem by proposing an algorithm based on semantic similarity between synsets computed by exploiting the hyponymy relation, which is used to organize the lexicon into a hierarchical structure. A concept represented by the synset $\{x, x', ...\}$ is said to be a hyponym of the concept represented by the synset $\{y, y', ...\}$ if native English speakers accept sentences constructed from such frames as "An x is a (kind of) y", like {maple} is a hyponym of {tree}, and {tree} is a hyponym of {plant}.

Figure 1 shows the output of WordNet when a user requests all the hypernyms for the noun senses of the word form *bat*. WordNet shows 5 noun senses for *bat*, and each corresponding synset is displayed followed by all the synsets that appear above it in the hypernym hierarchy. The figure shows only the hierarchy of sense 1.

The WSD procedure is fundamental for obtaining a synset-based vector space representation that we called *bag of synsets* (BOS). In this model, a document is represented as a synset vector rather than a word vector. Another key feature of the approach is that each document is represented by a set of *slots*, where each slot is a textual field corresponding to a specific feature of the document, in an attempt to take into account also the structure of documents. For example, in our application scenario, in which documents are movie descriptions, we selected five slots to represent movies:

1. *title*—the title of the movie;
2. *cast*—the list of the names of the actors appearing in the movie;
3. *director*—name(s) of the director(s) of the movie;
4. *summary*—a short text that presents the main parts of the story;
5. *keywords*—a list of words describing the main topics of the movie.

The text in each slot is represented according to the BOS model by counting occurrences of a synset separately in the slots in which it appears. More formally, assume that

```
bat, chiropteran - (nocturnal mouselike mammal with
forelimbs modified to form membranous wings and anatomical
adaptations for echolocation by which they navigate)
 => placental, placental mammal, eutherian, eutherian
    mammal - (mammals having a placenta; all mammals
    except monotremes and marsupials)
     => mammal - (any warm-blooded vertebrate having the
     skin more or less covered with hair; young are born
     alive except for the small subclass of monotremes and
     nourished with milk)
        => vertebrate, craniate - (animals having a bony
        or cartilaginous skeleton with a segmented
        spinal column and a large brain enclosed in a
        skull or cranium)
          => chordate - (any animal of the phylum Chordata
          having a notochord or spinal column)
            => animal, animate being, beast, brute,
            creature, fauna - (a living organism
            characterized by voluntary movement)
              => organism, being - (a living thing
              that has (or can develop) the ability to act
              or function independently)
                => living thing, animate thing - (a
                living (or once living) entity)
                   => object, physical object - (a
                   tangible and visible entity; an
                   entity that can cast a shadow; "it
                   was full of rackets, balls and
                   other objects")
                     => entity - (that which is
                     perceived or known or inferred to
                     have its own distinct existence
                     (living or nonliving))
```

Fig. 1 The hierarchy of sense 1 of the word "bat" obtained from WordNet (version 1.7.1)

we have a collection of N documents. Let m be the index of the slot, for $n = 1, 2, \ldots, N$, the nth document is reduced to five bags of synsets, one for each slot:

$$d_n^m = \langle t_{n1}^m, t_{n2}^m, \ldots, t_{nD_{nm}}^m \rangle$$

where t_{nk}^m is the kth synset in slot s_m of document d_n and D_{nm} is the total number of synsets appearing in the mth slot of document d_n. For all n, k and m, $t_{nk}^m \in V_m$, which is the vocabulary for the slot s_m (the set of all different synsets found in slot s_m). Document d_n is finally represented in the vector space by five synset-frequency vectors:

$$f_n^m = \langle w_{n1}^m, w_{n2}^m, \ldots, w_{nD_{nm}}^m \rangle$$

where w_{nk}^m is the weight of the synset t_k in the slot s_m of document d_n and can be computed in different ways: It can simply be the number of times synset t_k appears in the slot s_m or a more complex TF-IDF score.

2.2 A WordNet-based algorithm for WSD

The goal of a WSD algorithm is to associate the appropriate meaning or sense s to a word w in document d, by exploiting its *window of context* (or more simply *context*) C, that is a set of words surrounding w. The sense s is selected from a predefined set of

```
The    white    cat    is     hunting     the    mouse    (1)
The/DT white/JJ cat/NN is/VBZ hunting/VBG the/DT mouse/NN (2)
The/DT white/JJ cat/NN be/VB  hunt/VB     the/DT mouse/NN (3)
       white/JJ cat/NN        hunt/VB            mouse/NN (4)
```

Fig. 2 The preprocessing of sentence "The white cat is hunting the mouse". Each token is labeled with a tag describing its lexical role in the sentence. NN = noun, singular; VB = verb, base form; VBZ = verb, is; VBG = verb, gerund form; JJ = adjective; DT = determinative

possibilities, usually known as *sense inventory*. In the proposed algorithm, the sense inventory is obtained from WordNet. For example, let us consider the document *d*: "The white cat is hunting the mouse". The text in *d* is processed by two basic phases:

1. tokenization, POS tagging and lemmatization;
2. synset identification with WSD.

Figure 2 shows how *d* is represented in each substep of the first phase. The original sentence is (1) tokenized and, for each token, part of speech ambiguities are solved (2). Reduction to lemmas (3) (for example, verbs are turned to their base form) is performed before deleting stopwords (4).

As for lemmatization and POS tagging the MontyLingua natural language processor[3] for English has been adopted. Document *d*, after step (4) in Fig. 2, is the input for the synset identification phase. The core idea behind the proposed WSD algorithm is to disambiguate *w* by determining the degree of *semantic similarity* among candidate synsets for *w* and those of each word in *C*. Thus, the proper synset assigned to *w* is the one with the highest similarity for its specific context of use.

Several measures of similarity or relatedness are used to determine the degree of semantic similarity, or, more generally, relatedness, between two words based on their relative position in a concept hierarchy like WordNet, and possibly augmented with corpus-based information (Budanitsky and Hirst 2001). A crucial point is therefore the choice of a suitable similarity measure, by taking into account the specificness of the user profiling task we are addressing. In the following part of this paper, we discuss the choice of the semantic similarity adopted in the WSD algorithm, before describing in further detail the complete procedure.

2.2.1 The semantic similarity measure

A natural way to evaluate semantic similarity in a taxonomy is to measure the distance between the nodes corresponding to the items being compared. The shorter the path from one node to another, the more similar they are. In this work, semantic similarity is computed by means of the Leacock-Chodorow (1998) measure, which is based on the length of the path between concepts in an IS-A hierarchy. The idea behind this measure is that similarity between synsets *a* and *b* is inversely proportional to the distance between them in the WordNet *is-a* hierarchy, measured by the number of nodes in the shortest path (the path having minimum number of nodes) from *a* to *b*. The similarity is computed in the proposed WSD algorithm (see Algorithm 1) by the function `SinSim` (lines 24–29): The path length N_p is scaled by the depth D of the hierarchy, where depth is defined as the length of the longest path from a leaf node to the root node of the hierarchy.

[3] http://web.media.mit.edu/hugo/montylingua

In a study conducted by Patwardhan et al. (2003), a detailed analysis of the performance of several similarity measures is performed using a variety of different sources to determine the semantic relatedness of words. The main finding of the study is that measures combining the structure of WordNet with information content values taken from corpora provided better results compared to measures that rely only on the concept hierarchy structure or information content values. Information content of a concept is a measure of the specificity of a concept in a hierarchy. It is usually estimated by counting the frequency of that concept in a *large* corpus. If sense-tagged text is available, frequency counts of concepts can be attained directly, since each concept will be associated with a unique sense. If sense tagged text is not available (which is the usual situation), it will be necessary to adopt an alternative counting scheme. For example, Resnik (1998) suggests counting the number of occurrences of a word in a corpus, and then dividing that count by the number of different senses associated with that word. This value is then assigned to each concept.

In our case, disambiguation is performed for the specific task of building a user profile. Therefore, the corpus that should be adopted to estimate the frequency of concepts is the set of documents on which the user provided ratings. It is unreasonable to assume that this corpus is annotated with senses or that it is sufficiently large to perform an alternative counting scheme as the one suggested by Resnik.

These problems do not allow the adoption of measures based on corpus frequencies and lead us to rely on an approach exclusively based on the knowledge coming from WordNet.

2.2.2 The WSD procedure

In this section the WSD procedure based on the Leacock-Chodorow measure is described. Throughout the section, the sentence *"The white cat is hunting the mouse"* is used as the reference example to explain the steps of the WSD procedure. Let w = "cat" be the word to be disambiguated. The procedure starts by defining the context C of w as the set of words in the same slot of w having the same POS as w. In this case, the only *noun* in the sentence is "mouse", then $C = \{mouse\}$. Next, the algorithm identifies both the sense inventory for w, that is X = `{01789046: feline mammal, 00683044: computerized axial tomography,...}`, and the sense inventory X_j for each word w_j in C. Thus, X_j = `{01993048: small rodents, 03304722: a hand-operated electronic device that controls the coordinates of a cursor, ... }`. The sense inventory T for the whole context C is given by the union of all X_j (in this case, as C has a single word, then $X_j = T$). After this step, we measure the similarity of each candidate sense $s_i \in X$ to the one of each sense $s_h \in T$ and then the sense assigned to w is the one with the highest similarity score. In the example, `SinSim(01789046: feline mammal, 01993048: small rodents) = 0.806` is the highest similarity score, thus w is interpreted as "feline mammal".

Each document is mapped into a list of WordNet synsets following three steps:

1. each monosemous word w in a slot of a document d is mapped into the corresponding WordNet synset;
2. for each pair of words ⟨*noun,noun*⟩ or ⟨*adjective,noun*⟩, a search in WordNet is made to verify if at least one synset exists for the bigram $\langle w_1, w_2 \rangle$. In the positive case, Algorithm 1 is applied to the bigram, otherwise it is applied separately to w_1

Algorithm 1 The WordNet-based WSD algorithm

procedure WSD(w, d) $\triangleright$ finds the proper synset of a polysemous word w in document d
$C \leftarrow \{w_1, \dots, w_n\}$ $\triangleright$ C is the context of w. For example, $C = \{w_1, w_2, w_3, w_4\}$ is a window with radius=2, if the sequence of words $\{w_1, w_2, w, w_3, w_4\}$ appears in d
$X \leftarrow \{s_1, \dots s_k\}$ $\triangleright$ X is sense inventory for w, that is the set of all candidate synsets for w returned by WordNet
$s \leftarrow null$ $\triangleright$ s is the synset to be returned
$score \leftarrow 0$ $\triangleright$ $score$ is the similarity score assigned to s with respect to the context C
$T \leftarrow \emptyset$ $\triangleright$ T is the set of all candidate synsets for all words in C
for all $w_j \in C$ **do**
if $POS(w_j) = POS(w)$ **then** $\triangleright$ $POS(y)$ is the part-of-speech of y
$X_j \leftarrow \{s_{j1}, \dots s_{jm}\}$ $\triangleright$ X_j is the set of m possible senses for w_j
$T \leftarrow T \cup X_j$
end if
end for
for all $s_i \in X$ **do**
for all $s_h \in T$ **do**
$score_{ih} \leftarrow$ SINSIM(s_i, s_h) $\triangleright$ computing similarity scores between s_i and every synset $s_h \in T$
if $score_{ih} \geq score$ **then**
$score \leftarrow score_{ih}$
$s \leftarrow s_i$ $\triangleright$ s is the synset $s_i \in X$ having the highest similarity score with respect to the synsets in T
end if
end for
end for
return s
end procedure
function SINSIM(a, b) $\triangleright$ The similarity of the synsets a and b
$N_p \leftarrow$the number of nodes in path p from a to b
$D \leftarrow$maximum depth of the taxonomy $\triangleright$ $D = 16$ in WordNet 1.7.1
$r \leftarrow -log(N_p/2D)$
return r
end function

and w_2; in both cases all words in the slot are used as the context C of the word(s) to be disambiguated;

3. each polysemous unigram w is disambiguated by Algorithm 1, using all words in the slot as the context C of w.

As an example, Fig. 3 shows a fragment of the BOS representation for the document in Fig. 4. To improve readability, in addition to the synset unique identifier (and the number of occurrences for the synset), the natural language description of the synset (as provided by WordNet) is shown.

Our hypothesis is that the proposed indexing procedure helps to obtain profiles able to recommend documents semantically closer to the user interests. The difference compared to keyword-based profiles is that synset unique identifiers are used instead of words. The main advantages of a synset-based document representation are:

1. each ambiguous term in the document is disambiguated, therefore allowing its correct interpretation and consequently a higher precision in the user model construction (e.g. if a user is interested in financial news, a document containing the word "bank" in the context of geography will not be relevant);
2. synonym words belonging to the same synset can contribute to the user profile definition by referring to the same concept. If two synonyms appear in the same

```
title: {shining-the work of making something shine by
        polishing it; "the shining of shoes provided a
        meager living"-434048: 1.0}
director: {stanley kubrick-United States filmmaker
           (born in 1928)-9111534: 1.0}
cast: {}
summary: {male-(biology) being the sex (of plant or
         animal) that produces gametes (spermatozoa)
         that perform the fertilizing function in
         generation; "a male infant";
         "a male holly tree"-1432909: 1.0,
         novelist-someone who writes novels-8492863: 2.0,
         ...

keywords: {extrasensory perception-apparent power to
           perceive things that are not present to the
           senses-6047688: 1.0,
           freeze-be cold-00075821: 1.0,
           death-the event of dying or departure from
           life-06904072: 1.0;
           ...}
```

Fig. 3 The bag of synsets representation of the movie “The Shining”

```
title:  The Shining
director: Stanley Kubrick
cast:  Jack Nicholson, Shelley Duvall, Danny Lloyd,
       Scatman Crothers, Barry Nelson, Philip Stone,
       Joe Turkel, Anne Jackson, Tony Burton, Lia Beldam,
       Billie Gibson, Barry Dennen...
summary:  A male novelist is having writer's block.
          He, his wife, and his young son become
          the care-takers of a haunted hotel so
          he can go back to writing again. Once they
          start meeting the ghosts, they talk to
          them by ''shining'' (telepathic conversation)...
keywords: extrasensory-perception, freeze-to-death, bar,
          axe-murder,psychological-drama, child-in-peril,
          whiskey, murder, winter...
```

Fig. 4 The five slots corresponding to description of the movie “The Shining”

slot of a document, in the corresponding BOW we count *one* occurrence for each word form; conversely, in the BOS we count *two* occurrences of the corresponding synset. For example, both “bank” and “bank transaction” bring evidence for financial documents, improving *recall* in retrieval or categorization tasks;

3. recognition of bigrams as concepts. For example, if words “artificial” and “intelligence” occur in the same slot of a document, in the corresponding BOW we count one occurrence for each word; in the BOS, we count only one occurrence of the synset “{05766061} *artificial intelligence, AI*—(*the branch of computer science*...)”.

In the next sections, we will describe different learning algorithms used for the task of acquiring user profiles: Two algorithms, namely the Rocchio (1971) relevance feedback and naïve Bayes (Mitchell 1997), have been adapted for such a task. Two different prototypes have been developed in order to implement the proposed approaches: *RocchioProfiler* and *ITem Recommender* (ITR). The final goal was to evaluate the effectiveness of the above mentioned methods in learning intelligible profiles of user interests.

2.3 A relevance feedback method for learning WordNet-based profiles

In the Rocchio algorithm, documents are represented with the vector space model and the major heuristic component is the TFIDF word weighting scheme (Rocchio 1971):

$$\text{TFIDF}(t_k, d_j) = \underbrace{\text{TF}(t_k, d_j)}_{\text{TF}} \cdot \underbrace{\log \frac{N}{n_k}}_{\text{IDF}} \tag{1}$$

where N is the total number of documents in the training set and n_k is the number of documents containing the term t_k. $\text{TF}(t_k, d_j)$ computes the frequency of t_k in document d_j. Learning combines vectors of positive and negative examples into a prototype vector $\vec{c}$ for each class in the set of classes C. The method computes a classifier $\vec{c_i} = \langle \omega_{1i}, \ldots, \omega_{|T|i} \rangle$ for category c_i (T is the *vocabulary*, that is the set of distinct terms in the training set) by means of the formula:

$$\omega_{ki} = \beta \cdot \sum_{\{d_j \in POS_i\}} \frac{\omega_{kj}}{|POS_i|} - \gamma \cdot \sum_{\{d_j \in NEG_i\}} \frac{\omega_{kj}}{|NEG_i|} \tag{2}$$

where ω_{kj} is the TFIDF weight of the term t_k in document d_j, POS_i and NEG_i are the set of positive and negative examples in the training set for the specific class c_i, β and γ are control parameters that allow setting the relative importance of *all* positive and negative examples. To assign a class $\widetilde{c}$ to a document d_j, the similarity between each prototype vector $\vec{c_i}$ and the document vector $\vec{d_j}$ is computed and $\widetilde{c}$ will be the c_i with the highest value of similarity. We propose a modified version of this method *able to manage documents structured in slots and represented by WordNet synsets*. As reported in Sect. 2.1, each document d_j is represented in the vector space model by five synset-frequency vectors:

$$f_j^m = \langle w_{j1}^m, w_{j2}^m, \ldots, w_{jD_{jm}}^m \rangle$$

where D_{jm} is the total number of different synsets appearing in the mth slot of document d_j and w_{jk}^m is the weight of the synset t_k in the slot s_m of document d_j, computed according to a synset weighting strategy described in (Degemmis et al. 2005). Here, we will not report the details of the strategy because they do not add anything to the discussion of the proposed hybrid technique.

2.3.1 Synset-based profiles

Given a user u and a set of rated movies in a specific genre G (e.g. *Comedy*), the aim is to learn a profile able to recognize movies liked by the user in that genre. In B2C e-commerce, often items are grouped in a fixed number of categories. For example, at Amazon.com, books are organized in many subject categories or DVDs are subdivided into genres. However, users are hardly ever interested in all these categories, and their preferences focus only on a small subset. Our choice is to exploit this categorization of items in the profile leaning process: For a specific user u, we maintain separate profiles, one for each subject category he/she provided some feedback (ratings). Both the content-based systems proposed in this work are conceived as text classifiers. Thus, if all the items rated in different categories were included in a unique training set, the performance of the learned classifiers would be affected by the fact that learning relies on examples that might be very dissimilar.

Learning consists in inducing one prototype vector for *each slot*: These five vectors will represent the user profile. Each prototype vector could contribute in a different way to the calculation of the similarity between the vectors representing a movie and the vectors representing the user profile. More formally, we compute one prototype vector $\overrightarrow{p_i^m} = \langle \omega_{1i}^m, \ldots, \omega_{|V_m|i}^m \rangle$ for each slot s_m and for each class c_i (c_+ and c_-, user-likes and user-dislikes, respectively, V_m is the vocabulary for slot s_m, that is the set of all distinct synsets appearing in slot s_m) by using the ratings given by the user on movie descriptions in genre G. In other words, the method builds two profiles for user u and genre G: The positive profile (composed by 5 prototypes $\overrightarrow{p_+^m}$, corresponding to the slots) is learned from positive examples, the negative profile ($\overrightarrow{p_-^m}$) is learned from negative examples. Each rating $r_{u,j}$ on the document d_j is a discrete judgment ranging from 1 to 6 used to compute the coordinates of the vectors in both the positive and the negative user profile:

$$\omega_{ki}^m = \sum_{\{d_j \in POS_i\}} \frac{w_{jk}^m \cdot r'_{u,j}}{|POS_i|} \tag{3}$$

$$\omega_{ki}^m = \sum_{\{d_j \in NEG_i\}} \frac{w_{jk}^m \cdot r'_{u,j}}{|NEG_i|} \tag{4}$$

where $r'_{u,j}$ is the normalized value of $r_{u,j}$ ranging between 0 and 1 (respectively corresponding to $r_{u,j} = 1$ and 6), $POS_i = \{d_j \in T_r | r_{u,j} > 3\}$, $NEG_i = \{d_j \in T_r | r_{u,j} \leq 3\}$, and w_{jk}^m is the weight of the synset t_k in the slot s_m of document d_j, computed as in (Degemmis et al. 2005). Equations 3 and 4 differ from the classical formula in the fact that the parameters β and γ are substituted by the ratings $r'_{u,j}$ that give a different weight to each document in the training set. The similarity between a profile and a movie is obtained by computing five partial similarity values between each pair of corresponding vectors $\overrightarrow{p_i^m}$ and $\overrightarrow{d_j^m}$. A weighted average of the five values is computed, assigning a different weight α_m to underline the importance of a slot in classifying a movie. In our experiments, we used $\alpha_1 = 0.10$ (title), $\alpha_2 = 0.15$ (director), $\alpha_3 = 0.15$ (cast), $\alpha_4 = 0.25$ (summary) and $\alpha_5 = 0.35$ (keywords). The values α_m were defined according to experiments not reported in the paper as too lengthy. Each experiment consisted in a run of the Rocchio algorithm by using a different selection of α_m values. Here, with the term *run* we intend executing all the experimental sessions on the 10 "Genre" EachMovie datasets described in Table 4 and whose results are reported in Table 5. The α_m values reported here are those that allowed to obtain the best predictive accuracy, corresponding to results in Table 5. Since the user profile is composed by both the positive and the negative profiles, we compute two similarity values, one for each profile. The document d_j is considered interesting only if the similarity value of the positive profile is higher than the similarity of the negative one.

Figure 5 depicts an example of semantic profile induced by RocchioProfiler. The difference compared to the same keyword-based profile (Fig. 6) is that synset unique identifiers are used instead of words.

Fig. 5 An example of a synset-based profile

```
Selected Category: 1 - Action
PROFILE FOR: 4772 / 1
Generation date :  29-Nov-2004  3:25:16
**********     PROFILE     **********
**********     CLASS: LIKES
---------------------------------------------
SLOT: title

SYNSET                                  Weight
13711152 -  the condition of being
susceptible to harm or injury           0.0051
2780280 - soft fine feathers            0.0023
...
...
...
Slot title length = 0.004647657465
---------------------------------------------
SLOT: summary

SYNSET                                  Weight
09409641 - the male ruler of
an empire                               0.0183
...

Slot summary length = 0.0199832422
---------------------------------------------
**********     CLASS: DISLIKES
---------------------------------------------
SLOT: title

SYNSET                                 Weight
08030730 - a region marked off for
administrative or other purposes       0.0011
...
...
Slot title length = 0.0000369802
---------------------------------------------
```

2.4 A Naïve Bayes method for user profiling

Naïve Bayes is a probabilistic approach to inductive learning. The learned probabilistic model estimates the *a posteriori* probability of document d_i belonging to class c_j, $P(c_j|d_i)$. This estimation is based on three probability values: The a priori probability, $P(c_j)$, i.e. the probability of observing a document in class c_j, $P(d_i|c_j)$, that is the probability of observing document d_i given c_j, and $P(d_i)$, the probability of observing the instance d_i. Using these probabilities, Bayesian classifiers apply Bayes theorem to calculate $P(c_j|d_i)$. To classify a document d_i, the class with the highest probability is selected. As a working model for the naïve Bayes classifier, the multinomial event model (McCallum and Nigam 1998) is adopted:

$$P(c_j|d_i) = P(c_j) \prod_{w \in V_{d_i}} P(t_k|c_j)^{N(d_i,t_k)} \tag{5}$$

Fig. 6 An example of a keyword-based profile

```
Selected Category: 1 - Action
PROFILE FOR: 4772 / 1
Generation date :  7-Sep-2004  11:35:36
**********     PROFILE     **********
**********     CLASS: LIKES
---------------------------------------------
SLOT: title

TOKEN                 Weight
danger                0.0012
dawn                  0.1497
fiction               0.2799
romance               0.2995
...
Slot title length = 0.007632205931562852
---------------------------------------------
SLOT: summary

TOKEN                 Weight
emperor               0.0159
boss                  0.0359
butch                 0.0667
time                  0.1194
...
Slot summary length = 0.004352457510245179
...

---------------------------------------------
**********     CLASS: DISLIKES
---------------------------------------------
SLOT: title

TOKEN                 Weight
territory             0.0996
video                 0.0019
lies                  0.0500
...
..
---------------------------------------------
```

where $N(d_i, t_k)$ is defined as the number of times word or token t_k appeared in document d_i. Notice that rather than getting the product of all distinct words in the corpus, V, we only use the subset of the vocabulary, V_{d_i}, containing the words that appear in the document d_i. A key step in implementing naïve Bayes is estimating the word probabilities $P(t_k|c_j)$. We use Witten-Bell (1991) smoothing that sets $P(t_k|c_j)$ as follows:

$$P(t_k|c_j) = \begin{cases} \dfrac{N(t_k, c_j)}{V_{c_j} + \sum_i N(t_i, c_j)} & \text{if } N(t_k, c_j) \neq 0 \\ \dfrac{V_{c_j}}{V_{c_j} + \sum_i N(t_i, c_j)} \dfrac{1}{V - V_{c_j}} & \text{if } N(t_k, c_j) = 0 \end{cases} \tag{6}$$

where $N(t_k, c_j)$ is the count of the number of times t_k occurs in the training data for class c_j, and $|V_{c_j}|$ is the total number of unique synsets in class c_j. ITR implements the above described method to classify documents as interesting or uninteresting for a particular user. In order to compare the results obtained by RocchioProfiler and ITR, we train both systems on the same dataset, the EachMovie dataset, in which ratings from 72,916 users were recorded on a discrete 6-point scale from 1 to 6 (see Sect. 2.5 for a detailed description of the dataset). An instance labeled with a rating r, $1 \leq r \leq 3$, belongs to class c_- (user-dislikes); if $4 \leq r \leq 6$ then the instance belongs to class c_+ (user-likes). Each rating r was normalized to obtain values ranging between 0 and 1:

$$w_+^i = \frac{r-1}{\mathrm{MAX}-1}; \qquad w_-^i = 1 - w_+^i \tag{7}$$

where MAX is the maximum rating that can be assigned to an instance.

In the collection, movies are grouped by genre (categories). ITR learns a profile of the movies preferred by a user in a specific category or genre G, as for RocchioProfiler. Thus, given a user u and a set of rated movies in a specific category of interest, the system learns a profile able to recognize movies liked by u in that category. Since each instance is encoded as a vector of documents, one for each BOS, Eq. 5 becomes:

$$P(c_j|d_i) = \frac{P(c_j)}{P(d_i)} \prod_{m=1}^{|S|} \prod_{k=1}^{|b_{im}|} P(t_k|c_j, s_m)^{n_{kim}} \tag{8}$$

where $S = \{s_1, s_2, \ldots, s_{|S|}\}$ is the set of slots, b_{im} is the BOS in the slot s_m of the instance d_i, n_{kim} is the number of occurrences of the synset t_k in b_{im}. To calculate (8), we only need to estimate $P(c_j)$ and $P(t_k|c_j, s_m)$. The weights in (7) are used for weighting the occurrences of a synset in a document and to estimate the probability terms from the training set TR. The prior probabilities of the classes are computed according to the following equation:

$$\hat{P}(c_j) = \frac{\sum_{i=1}^{|\mathrm{TR}|} w_j^i + 1}{|\mathrm{TR}| + 2} \tag{9}$$

Witten-Bell estimates in (6) have been modified by taking into account that documents are structured into slots and that word occurrences are weighted according to Eq. 7:

$$\hat{P}(t_k|c_j, s_m) = \begin{cases} \dfrac{N(t_k, c_j, s_m)}{V_{c_j} + \sum_i N(t_i, c_j, s_m)} & \text{if } N(t_k, c_j, s_m) \neq 0 \\ \dfrac{V_{c_j}}{V_{c_j} + \sum_i N(t_i, c_j, s_m)} \dfrac{1}{V - V_{c_j}} & \text{if } N(t_k, c_j, s_m) = 0 \end{cases} \tag{10}$$

where $N(t_k, c_j, s_m)$ is the count of the weighted occurrences of the word t_k in the training data for class c_j in the slot s_m, V_{c_j} is the total number of unique words in class c_j, and V is the total number of unique words across all classes. $N(t_k, c_j, s_m)$ is computed as follows:

$$N(t_k, c_j, s_m) = \sum_{i=1}^{|\mathrm{TR}|} w_j^i n_{kim} \tag{11}$$

In (11), n_{kim} is the number of occurrences of the term t_k in the slot s_m of the ith instance. The sum of all $N(t_k, c_j, s_m)$ in the denominator of Eq. 10 denotes the total weighted length of the slot s_m in the class c_j. In other words, $\hat{P}(t_k|c_j, s_m)$ is estimated as

Table 2 Number of features used to represent movies

Slot	# Features BOW	# Features BOS
Title	3,080	2,516
Cast	46,568	352
Director	3,559	120
Summary	77,015	67,217
Keywords	42,074	37,785
Total	172,296	107,990

a ratio between the weighted occurrences of synset 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 that includes those words that turn out to be most indicative of the user preferences, according to the value of the conditional probabilities in (10).

2.5 Experimental evaluation of synset-based profiles

The goal of this phase is to evaluate whether the new synset-based versions of ITR and RocchioProfiler actually improve the performance with respect to the keyword-based versions of the systems. For this purpose, two experimental sessions have been conducted, one for each system. Finally, the results obtained by synset-based profiles produced by both systems have been compared. The documents in the EachMovie dataset have been disambiguated using Algorithm 1, obtaining a reduction of the number of features used to represent movies (the reduction is roughly 38%—see Table 2). This result is mainly due to three reasons:

- WordNet is able to recognize only a few proper names, thus many actors and directors have not been recognized;
- the WSD procedure is able to recognize bigrams like “artificial intelligence” or “white house”;
- obviously, synonym words have been represented by the same synset.

2.5.1 The EachMovie dataset

The experimental work has been carried out on a collection of 1,628 textual descriptions of movies rated by 72,916 real users, the EachMovie dataset.[4] The movies are rated on a 6-point scale that was mapped linearly into the interval [0,1]. The original dataset does not contain any information about the content of the movies. The content information for each movie was collected from the Internet Movie Database (see Fig. 7) using a simple crawler that, following the IMDb link provided in the original dataset, collects information from the various links of the main URL. In particular the crawler gathers the *Title*, the *Director*, the *Genre*, that is the category of the movie, the list of *Keywords*, the *Summary* and the *Cast*. Figure 8 reports an example of summary related to the movie “Young Frankenstein”. The retrieved content data is provided in a CSV (comma separated value) text file. After appropriate preprocessing (the operations performed on each document are listed in Table 3), the content is organized and stored in a relational database.

[4] EachMovie dataset no longer available for download: http://www.cs.umn.edu/Research/GroupLens/

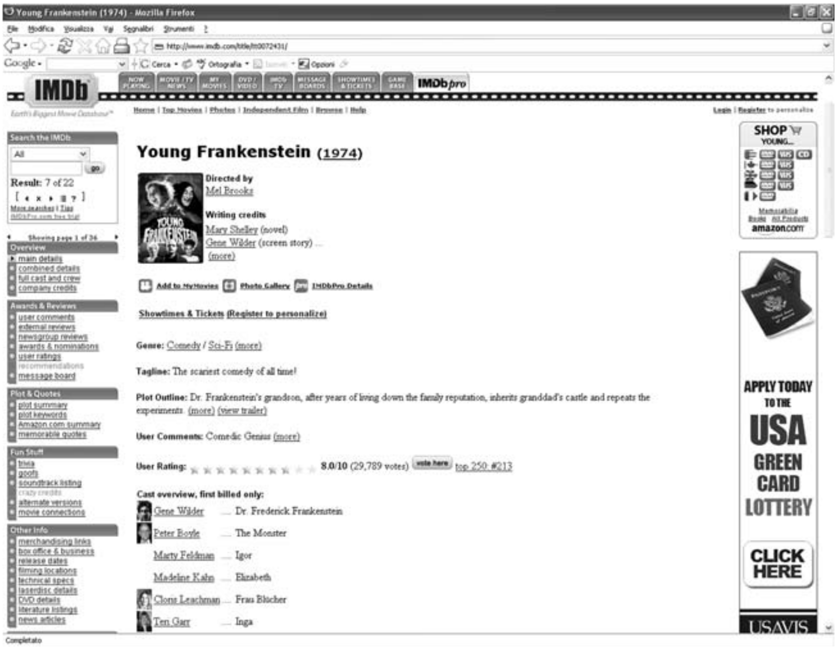

Fig. 7 The web page of the movie "Young Frankenstein" on the Internet Movie Database

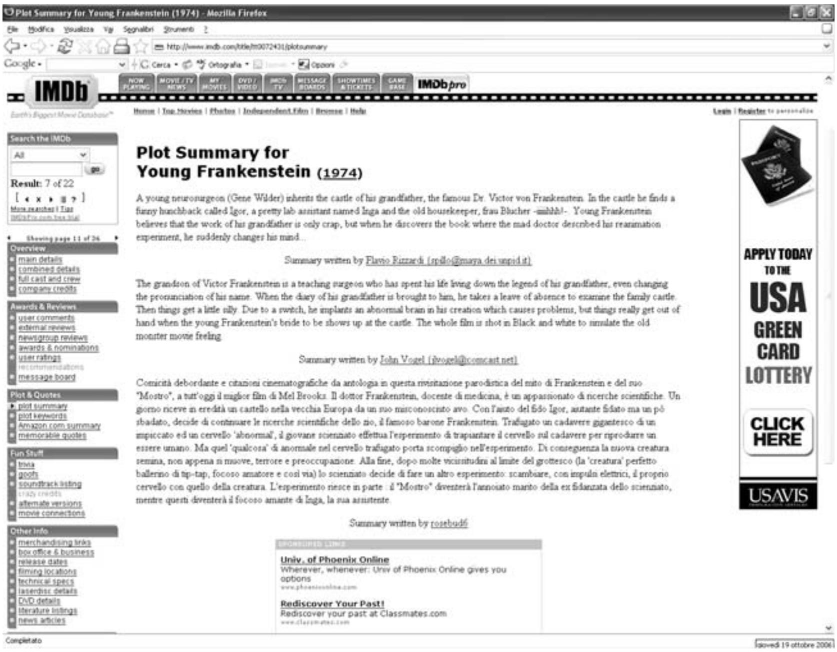

Fig. 8 Summary web page of the movie "Young Frankenstein"

Table 3 Preprocessing operations performed on the EachMovie dataset

Slot	Tokenization	Stopword elimination	Stemming
Title	X		
Cast	X		
Director	X		
Summary	X	X	X
Keywords	X	X	X

Table 4 10 'Genre' datasets obtained from the original EachMovie dataset

Id Genre	Genre	Number ratings	% POS	% NEG
1	Action	4,474	72	28
2	Animation	1,103	57	43
3	Art_Foreign	4,246	76	24
4	Classic	5,026	92	8
5	Comedy	4,714	63	37
6	Drama	4,880	76	24
7	Family	3,808	64	36
8	Horror	3,631	60	40
9	Romance	3,707	73	27
10	Thriller	3,709	72	28
		39,298	72	28

The content of slots *title*, *cast* and *director* was only tokenized because we observed that the process of stopword elimination produced some unexpected results: For example, slots containing exclusively stopwords, such as "*It*" or "*E.T.*", became empty. Moreover, it does not make sense to apply stemming and stopword elimination on proper names.

Movies are divided into different genres: *Action*, *Animation*, *Classic*, *Art_Foreign*, *Comedy*, *Drama*, *Family*, *Horror*, *Romance*, *Thriller*.

For each genre or category, a set of 100 users was randomly selected among users that rated n items in that movie category, $30 \leq n \leq 100$ (only for genre 'animation', the number of users that rated n movies was 33, due to the low number of movies in that genre). In this way, for each category, a dataset of at least 3,000 triples (user, movie, rating) was obtained (at least 990 for 'animation'). Table 4 summarizes the data used for the experiments. The number of movies rated as positive and negative is balanced in genre datasets 2, 5, 7, 8 (55–70% positive, 30–45% negative), while is unbalanced in genre datasets 1, 3, 4, 6, 9, 10 (over 70% positive).

2.5.2 Design and results of the experiments

As our content-based profiling systems are conceived as text classifiers, their effectiveness is mainly evaluated by classification accuracy measures *precision* and *recall* (Herlocker et al. 2004). Precision (Pr) is defined as the number of relevant selected items divided by the number of selected items. Recall (Re) is defined as the number of relevant selected items divided by the total number of relevant items available. For the evaluation of recommender systems, they have been used in (Billsus and Pazzani 1998; Basu et al. 1998; Sarwar et al. 2000a,b). $F1$ measure, a combination of precision and recall, is also used:

$$F1 = \frac{2 \times Re \times Pr}{Pr + Re}$$

$F1$ has been used to evaluate recommender systems in (Sarwar et al. 2000a,b). These classification measures do not consider predictions and their deviations from actual ratings, they rather compute the frequency with which a recommender system makes correct or incorrect decisions about whether an item is good. We adopted these measures because in this phase we are interested in measuring how relevant a set of recommendations is for the active user. Rank accuracy metrics measure the ability of a recommender system to produce a recommended ordering of items that matches how the user would have ordered the same items. In our study, we adopted the Normalized Distance-based Performance Measure (NDPM) originally proposed by Yao (1995) to compare the ranking imposed by the user ratings with the classification scores given by both RocchioProfiler (the similarity score for the class *likes*) and ITR (the a-posteriori probability of the class *likes*). Values range from 0 (agreement) to 1 (disagreement). The adoption of both classification accuracy and rank accuracy metrics gives us the possibility to evaluate both whether the systems are able to recommend good items and how these items are ranked. For example, even if the top ten items ranked by the systems were relevant, a rank accuracy metric might give a low value because the best item is actually ranked 10th.

In all the experiments, a movie description d_i is considered as *relevant* by a user if the rating is greater or equal to 4. RocchioProfiler considers an item as relevant if the similarity score for the class *likes* is higher than the one for the class *dislikes*, while ITR considers an item as relevant if the a-posteriori probability of the class *likes* is greater than 0.5. We executed one experiment for each user in the dataset: The ratings of each specific user and the content of the rated movies have been used for learning the user profile and measuring its predictive accuracy, using the aforementioned measures. Each experiment consisted of:

1. selecting ratings of the user and the content of the movies rated by that user;
2. splitting the selected data into a training set *Tr* and a test set *Ts*;
3. using *Tr* for learning the corresponding user profile;
4. evaluating the predictive accuracy of the induced profile on *Ts*, using the aforementioned measures.

The methodology adopted for obtaining *Tr* and *Ts* was the 10-fold cross validation (Kohavi 1995). Table 5 reports the results obtained over all 10 genres by RocchioProfiler. We notice a 2% improvement on average in precision of the BOS model over the BOW one. In more detail, the BOS model significantly outperforms the BOW one on datasets 3 (+8%), 7 (+6%), 8 (+5%). Only dataset 2 showed no improvement. This is probably due both to the low number of ratings and to the specific features of the movies, in most cases stories, that makes the disambiguation difficult. Also recall (+3%) and $F1$-measure (+2%) obtained by the BOS model are improved over those obtained by the BOW model. In particular, a significant improvement of recall was observed again on dataset 3 (+7%), 7 (+5%), 8 (+6%). This could be an indication that the improved results are independent from the distribution of positive and negative examples in the datasets: Datasets 7 and 8 are balanced, while dataset 3 is unbalanced. NDPM has not been improved, but it remains acceptable. It could be noted, from the NDPM values, that the relevant/not relevant classification is improved without improving the ranking. This situation could be explained by the example in

Table 5 Comparison between BOW-generated profiles and BOS-generated profiles obtained by RocchioProfiler

Id Genre	Precision		Recall		F1		NDPM	
	BOW	BOS	BOW	BOS	BOW	BOS	BOW	BOS
1	0.74	0.75	0.84	0.86	0.76	0.79	0.46	0.44
2	0.65	0.64	0.70	0.70	0.68	0.63	0.34	0.38
3	0.77	0.85	0.80	0.87	0.77	0.84	0.46	0.48
4	0.92	0.94	0.94	0.96	0.93	0.94	0.45	0.43
5	0.67	0.69	0.72	0.75	0.67	0.70	0.44	0.46
6	0.78	0.79	0.84	0.87	0.80	0.81	0.45	0.45
7	0.68	0.74	0.79	0.84	0.73	0.77	0.41	0.40
8	0.64	0.69	0.78	0.84	0.69	0.73	0.42	0.44
9	0.75	0.76	0.83	0.85	0.76	0.77	0.48	0.48
10	0.74	0.75	0.84	0.85	0.77	0.78	0.45	0.44
Avg.	0.74	0.76	0.81	0.84	0.76	0.78	0.44	0.44

Table 6 Example of situation in which classification is improved without improving ranking

Item	R_{u}	R_{A}	R_{B}
I1	6 (1)	0.65 (2)	0.65 (2)
I2	5 (2)	0.62 (3)	0.60 (3)
I3	5 (3)	0.75 (1)	0.70 (1)
I4	4 (4)	0.60 (4)	0.45 (5)
I5	4 (5)	0.43 (6)	0.42 (6)
I6	3 (6)	0.55 (5)	0.55 (4)
I7	3 (7)	0.40 (7)	0.40 (7)
I8	2 (8)	0.30 (8)	0.30 (8)
I9	1 (9)	0.25 (9)	0.25 (9)
I10	1 (10)	0.20 (10)	0.20 (10)

Table 6, in which each column reports the ratings of the items and the corresponding position in the ranking (in brackets).

Let R_{u} be the ranking imposed by the user u on a set of 10 items, let R_{A} be the ranking computed by A, let R_{B} be the ranking computed by method B (ratings ranging between 1 and 6—classification scores ranging between 0 and 1). An item is considered as relevant if the rating is greater than 3 (symmetrically, the score is greater than 0.5). Method A has a better classification accuracy compared to method B (Recall = 4/5, Precision = 4/5 vs. Recall = 3/5, Precision = 3/4). NDPM is almost the same for both methods because the two rankings R_{A} and R_{B} are very similar. The difference is that I4 is ranked above I6 in R_{A} whilst I6 is ranked above I4 in R_{B}. Thus, the general conclusion is that method A has improved the classification of items whose score (and ratings) is close to the relevant/not relevant threshold, thus items for which the classification is highly uncertain.

In our experiments, NDPM compared the ranking set by the user ratings and the similarity score for the class c_+: Further investigations will be carried out to define a better ranking score for computing NDPM, that will also take into account the negative part of the profile. A Wilcoxon signed ranked test ($p < 0.05$) has been performed to validate the results (Orkin and Drogin 1990). We considered each experiment as a single trial for the test. The test confirmed that there is a statistically significant

Table 7 Comparison between BOW-generated profiles and BOS-generated profiles obtained by ITR

Id Genre	Precision		Recall		$F1$		NDPM	
	BOW	BOS	BOW	BOS	BOW	BOS	BOW	BOS
1	0.70	0.74	0.83	0.89	0.76	0.80	0.45	0.45
2	0.51	0.57	0.62	0.70	0.54	0.61	0.41	0.39
3	0.76	0.86	0.84	0.96	0.79	0.91	0.45	0.45
4	0.92	0.93	0.99	0.99	0.96	0.96	0.48	0.48
5	0.56	0.67	0.66	0.80	0.59	0.72	0.46	0.46
6	0.75	0.78	0.89	0.92	0.81	0.84	0.46	0.45
7	0.58	0.73	0.67	0.83	0.71	0.79	0.42	0.42
8	0.53	0.72	0.65	0.89	0.58	0.79	0.41	0.43
9	0.70	0.77	0.83	0.91	0.75	0.83	0.49	0.49
10	0.71	0.75	0.86	0.91	0.77	0.81	0.48	0.48
Avg.	0.67	0.75	0.78	0.88	0.73	0.81	0.45	0.45

difference in favor of the BOS model compared to the BOW one as regards precision, recall and $F1$-measure, and that the two models are equivalent in defining the ranking of the preferred movies with respect to the score for the class “likes”.

The results of the comparison between the profiles obtained from documents represented using the two indexing approaches by ITR are reported in Table 7. We can notice a significant improvement of BOS over BOW both in precision (+8%) and recall (+10%). The BOS model outperforms the BOW one specifically on datasets 5 (+11% of precision, +14% of recall), 7 (+15% of precision, +16% of recall), 8 (+19% of precision, +24% of recall). Only on dataset 4 (Classic) we have not observed any improvement, probably because precision and recall are already very high, thus there is not much room for improvement. The above mentioned results could be interpreted as an indication that the improved results depend on the balanced distribution of positive and negative examples in the dataset (see Table 4). For NDPM, in this case it also remains stable, even if classification accuracy was improved. Also for ITR results, a Wilcoxon signed ranked test has been performed, requiring a significance level $p < 0.05$. The test confirmed that there is a statistically significant difference in favor of the BOS model compared to the BOW model as regards precision, recall and $F1$-measure, and that the two models are equivalent in defining the ranking of the preferred movies with respect to the score for the class “likes”.

2.5.3 The final choice

The main aim of the experiments was to verify which is the most suitable technique to be integrated in the hybrid recommender. Notice that “the most suitable technique” does not mean only “the technique that learns the most accurate profiles”. We should also verify which technique represents the user interests in the most effective way, by taking into account that profiles will not be used for item predictions, but for discovering users having similar preferences. First of all, we observed that both techniques significantly improve their overall accuracy when shifting from BOW to BOS: +2% $F1$ improvement for RocchioProfiler, +8% $F1$ improvement for ITR. Therefore, the first design choice is to integrate a method that learns *synset-based profiles* (in the evaluation of the hybrid system we will also perform an experiment that aims at confirming this choice). In the following part of the paper, we compare the results obtained by the

BOS versions of ITR and RocchioProfiler. RocchioProfiler performs slightly better than ITR as regards NDPM (RocchioProfiler: 0.44 vs. ITR: 0.45, even if the difference is not statistically significant) and also obtains a better precision (+1%).

Given the large number of items that a user has to choose from (e.g. items in a large repository such as the Amazon.com catalogue), we felt it is important to achieve a high level of precision, thereby making it more likely that an item selected from the set returned by the system will be liked. *Trust* is a keyword in giving recommendations: The system should minimize false positive errors. Moreover, as observed before, we can reasonably assume that the level of precision reached by RocchioProfiler does not depend on the distribution of the ratings in the training set, because improvements are obtained both on balanced and unbalanced datasets. This is a point in favor of RocchioProfiler.

On the other hand, ITR outperforms RocchioProfiler in recall (+3%). The level of recall achieved by the BOS version of ITR is mainly determined by the improvement observed on datasets 5, 7, 8 (+14%, +16%, +24% respectively). These results are always achieved on balanced datasets. The problem is that a balanced distribution of ratings in the training set is not always guaranteed. We interpreted this result as an evidence that RocchioProfiler is able to find interests in a more effective way than ITR, because it has a more "stable" behavior with respect to distribution of ratings in the dataset.

These motivations convinced us to use RocchioProfiler in the proposed new hybrid method.

3 A novel hybrid recommender based on user profiles

The classical trend in collaborative filtering is represented by memory-based algorithms (Resnick et al. 1994; Shardanand and Maes 1995; Breese et al. 1998; Nakamura and Abe 1998; Delgado and Ishii 1999). These methods first compute the similarity between users by directly comparing their preference ratings. The correlation between two users is an indicator of the match on the quality of objects assessed by the users in the system. The preference of a user (for an unrated item) is then predicted by summing up the contributions of other users for the same item, and weighted using a user similarity measure.

The introduction of the weights allows a user to take into account the opinions of the "like-minded" users to a further extent. Thus, the recommendation accuracy highly relies on how the underlying similarity measure is defined. To sum up, the main task of collaborative filtering can be seen as the task of predicting the rating for a particular user (henceforth called the *active user*) from a set of user ratings provided by other users in the database.

The main steps of the process of producing collaborative recommendations in nearest-neighbor algorithms are:

1. *Representation of input data:* the input data is a set of ratings of n users on m items. It is usually represented as an $n \times m$ user-item matrix, R, such that $r_{i,j}$ represents the rating assigned by the ith user on the jth item.
2. *Neighborhood Formation:* the neighborhood formation process is the model-building or learning process for a collaborative recommender. Users similar to the active user will form a proximity-based neighborhood with him/her. The main

goal of neighborhood formation is to find, for each user a, an ordered list of l users $N_a = (N_1, N_2, \ldots, N_l)$ so that $a \notin N_a$ and $sim(a, N_1)$ is maximum, $sim(a, N_2)$ is the next maximum and so on.

3. *Recommendation Generation:* the final step in the recommendation process is to produce either a prediction, which will be a numerical value representing the predicted opinion of the active user, or a recommendation, which will be expressed as a list of the *top-N* items that the active user will appreciate the most. In both cases, the result should be based on the neighborhood of users.

Collaborative filtering by itself cannot always guarantee a good prediction. The inaccuracy might increase if the number of people who have a correlation with the *active user* (for which recommendations have to be produced) is very low. Instead of performing content analysis, collaborative filtering systems rely entirely on interest ratings from members of a participating community. Many implementations of collaborative filtering apply some variations of the neighborhood-based prediction algorithm. There is no consensus as to which technique is the most appropriate for what situations. The combination of content and collaborative methods allows for recommendations that go beyond object similarity and that take into account the interests of users that are similar to those of the active user (Schwab et al. 2001).

A first attempt to improve collaborative recommendations by means of behavioral profiles inferred from the analysis of transactional data (browsing and purchasing history of customers of an electronic marketplace) showed promising results (Degemmis et al. 2004). Rules describing the customer behavior were induced and exploited to discover a set of "nearest neighbors" to compute collaborative recommendations for the active user.

The hybrid recommender proposed in this paper extends the process of producing collaborative recommendations in a nearest-neighbor algorithm as follows:

– *Neighborhood Formation:* this process is the core of the proposed hybrid recommender. A bisecting k-means clustering algorithm (Hartigan 1975; Hartigan and Wong 1979; Cutting et al. 1992; Bradley and Fayyad 1998; Larsen and Aone 1999) is applied to the set of user profiles for producing the set of like-minded users. The idea is to partition relevance feedback user profiles (Sect. 2.3) and use partitions as neighborhood N_a. More details are provided in the next section and in (Lops 2005).
– *Recommendation Generation:* the final step is the computation of the prediction $p_{a,j}$ for the active user a on item j, performed according to the classical collaborative filtering formula (Breese et al. 1998), based on the idea that similarities $w_{a,i}$ among a and his/her neighbors i are computed over profiles contained in N_a instead of ratings in R:

$$p_{a,j} = \bar{r}_a + \frac{\sum_{i \in N_a} w_{a,i}(r_{i,j} - \bar{r}_i)}{\sum_{i \in N_a} |w_{a,i}|} \tag{12}$$

where $r_{i,j}$ is the rating of the user i on the item j, and $\bar{r}$ is the average rating of a user. If a strategy to determine a subset X of neighbors from N_a is adopted (as the best-n-neighbors), then similarities $w_{a,i}$ are computed over profiles contained in X rather than profiles in N_a.

3.1 Clustering user profiles for neighborhood formation

Collaborative recommendations are generated taking into account opinions of similar users: A crucial issue is how overlapping user interests can be exploited to improve recommendations. Collaborative techniques take into account opinions of users that rated common items, but users can choose among hundreds of items to rate and new items become available continuously, thus it is likely that overlap of rated items between two users will be minimal in many cases. As a consequence, many correlation coefficients would be computed on just few observations. In the worst case, if users did not rate any common item, then their profiles would result to be not correlated at all, even though as a matter of fact this does not necessarily mean that they were not like-minded. Even more so, the correlation approach induces one single global model of similarity between users, rather than separate models for classes of positive and negative ratings. Current approaches measure whether two user profiles are either positively correlated, or not correlated at all, or negatively correlated. For these reasons, our work extends the concept of correlation by exploiting both parts a user profile consists of (Sect. 2.3.1). In this way, two users turn out *similar* not only if they share preferences, but also if they have *similar* negative tastes, according to the content descriptions of the items (e.g. movies directed by the same director, or with similar plot or with the same star in the cast). Computing similarities between users taking into account both parts of their profiles is likely to select like-minded users more precisely, thus providing better recommendations.

The idea is to aggregate user profiles in a collaborative filtering system by a clustering algorithm. This approach is different from others presented by several authors (Sarwar et al. 2002; Schwab et al. 2001; Ungar and Foster 1998). The added value is that profiles contain additional knowledge about interests of users, and the accuracy of predictions generated taking into account this knowledge should be better. Figure 9 and Algorithm 2 explain the idea. First, a clustering algorithm is applied to the set of user profiles inferred by the content analysis (steps 3 and 4 of Algorithm 2). In the next step, the neighborhood for the active user is defined as the union of clusters that contain the user profile of the active user. The same process of neighborhood selection is applied to both the positive and the negative parts of the user profile (step 5). Clusters obtained by positive parts represent groups of similar users because they share the same interests. Clusters obtained by negative parts represent again groups of similar users because they share common dislikes. Then, predictions are generated using Eq. 12.

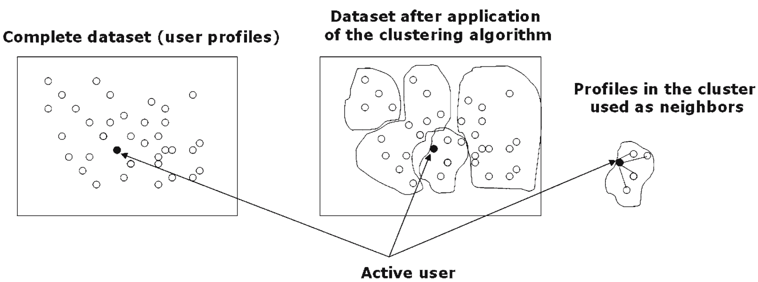

Fig. 9 Neighborhood formation from clustered partitions

Algorithm 2 Neighborhood Formation

1: **Input:** P^+, P^- $\triangleright$ P^+ is the set of positive profiles, that is the the set of profiles representing the positive interests of users, P^- is the set of negative profiles, that is the set of profiles representing the negative interests of users
2: **Output:** N_a $\triangleright$ Neighborhood of the active user a.
3: P^+ is partitioned in $P_1^+, P_2^+, \ldots, P_k^+$, where $P_i^+ \cap P_j^+ = \oslash$, for $1 \leq i,j \leq k$, $i \neq j$, and $P_1^+ \cup P_2^+ \cup \ldots \cup P_k^+ = P$. $\triangleright$ A clustering algorithm is applied to the set of positive profiles representing positive interests in the specific category to produce k partitions.
4: P^- is partitioned in $P_1^-, P_2^-, \ldots, P_k^-$, where $P_i^- \cap P_j^- = \oslash$, for $1 \leq i,j \leq k$, $i \neq j$, and $P_1^- \cup P_2^- \cup \ldots \cup P_k^- = P$. $\triangleright$ A clustering algorithm is applied to the set of negative profiles representing negative interests in the specific category to produce k partitions.
5: If $a \in P_i^+$ and $a \in P_j^-$, $N_a = P_i^+ \cup P_j^-$. $\triangleright$ Determine the *neighborhood* N_a for the active user a as the union of users contained in the cluster of positive profiles (C^+) and the cluster of negative profiles (C^-).

3.2 Experimental evaluation of the novel hybrid recommender

The purpose of the experimental session is to compare results of different neighborhood-based prediction algorithms in order to validate the hypothesis that user profiles are useful to improve the quality of recommendations.

Three different experiments have been performed:

- Experiment 1: Evaluation of a collaborative filtering algorithm using the Pearson's correlation coefficient for the neighborhood formation process.
- Experiment 2: Evaluation of a collaborative filtering algorithm using clusters of *keyword-based* user profiles for the neighborhood formation process.
- Experiment 3: Evaluation of a collaborative filtering algorithm using clusters of *synset-based* user profiles for the neighborhood formation process.

The basic evaluation sequence proceeds as follows. The dataset of users (and their ratings) is divided into a *training set* (the community) and a *test set*. We then iterate through the users in the test set, treating each user as the active user. We divide the ratings for the active user into a set of ratings that we treat as observed, I_a, and a set that we will attempt to predict, P_a. We use the ratings in I_a to predict the ratings in P_a as shown in Eq. 12.

The quality of recommendations is measured in terms of Mean Absolute Error (MAE) and Mean Squared Error (MSE) (Herlocker et al. 2004). MAE measures the average absolute deviation between a predicted rating and the user's true rating. The MAE is measured only for those items, for which user u_i has expressed his opinion. Lower Mean Absolute Errors correspond to more accurate recommender systems. MSE squares the error and more emphasis on large errors is given. The adoption of these measures is justified by the interest in measuring the error of the ratings predicted by the recommender systems with respect to the (true) user ratings. In fact, in the proposed movie recommending scenario, the main goal is to predict the number of stars that a user assigned to each movie, as depicted in Fig. 7.

3.2.1 Experiment 1

Results of this experiment are considered the baseline for a comparison to other methods.

The dataset used in this experiment is the user-item matrix that had 835 rows (i.e., 835 distinct users) and 1,613 columns (movies), obtained by grouping the 10 genres in the EachMovie datasets described in Sect. 2.5.1. The training set/test set split is 80%/20%. We adopted two different protocols:

– *All But 1 Protocol:* The test set P_a for each test user contains a single randomly selected rating and the observed set I_a contains the rest of the ratings. For each user in the test set, predictions were computed for the withheld items by using Eq. 12, where neighbors are selected from the community by using the Pearson's correlation coefficient for measuring the similarity between users.
– *Training/Test Protocol:* 20% of ratings are randomly placed for each test user in the observed set I_a, while the rest of the ratings are placed in the test set P_a. Predictions are computed as in the All But 1 Protocol.

The procedure was repeated 5 times selecting a different test set. This allows running 5 different trials corresponding to a 5-fold cross validation. Table 8 reports MAE and MSE for different sizes of the set of neighbors. The size ALL means that all the users in the user-item matrix are considered as neighbors. Last row of the table reports an average of the values.

3.2.2 Experiment 2

The second experiment measures the accuracy of the hybrid recommender system, where like-minded users are selected from clusters of keyword-based profiles, created using the bisecting k-means algorithm.

The experiment has been performed separately on each genre dataset described in Sect. 2.5.1. We adopted both the same training/test split and the two protocols as in Experiment 1:

1. *All But 1 Protocol*: For each user in the test set, predictions were computed for the withheld items by using Eq. 12. The neighborhood N_a of the active user is formed by Algorithm 2 by taking as input the set of positive and negative profiles of users in the community. The technique adopted to compute the neighbors is the best-n-neighbors (Vozalis and Margaritis 2003), n representing the neighborhood size, which picks out the best n correlates from N_a.
2. *Training/Test Protocol*: Same as in Experiment 1, but neighbors are picked out from N_a by best-n-neighbors.

Table 8 Performance of the collaborative filtering algorithm using the Pearson's correlation coefficient for the neighborhood formation

Neigh. size	Training/Test protocol		All But 1 protocol	
	MAE	MSE	MAE	MSE
20	0.91495986	1.47320112	0.90877139	1.45039610
30	0.91478420	1.46629900	0.90778644	1.44134278
40	0.91614997	1.46526980	0.90903524	1.44062510
50	0.91797904	1.46723864	0.91103786	1.44314138
100	0.93004515	1.49454212	0.92400561	1.47536678
ALL	0.94364049	1.52469950	0.93544843	1.50104554
Avg.	0.92292645	1.48187503	0.91601416	1.45865295

Table 9 Performance of the collaborative filtering algorithm using clusters of keyword-based profiles for the neighborhood formation [Average over all Categories]

Neigh. size	Training/Test protocol			All But 1 protocol	
	# clusters	MAE	MSE	MAE	MSE
20	5	0.89333210	1.44826048	0.88792554	1.50622503
30	3	0.89457071	1.44498053	0.88881724	1.50549011
40	2	0.89545209	1.44597572	0.88992782	1.51359789
50	2	0.89587027	1.44631562	0.89030066	1.51472140
100	1	0.89605402	1.44750222	0.89035761	1.50963302
Avg.		0.89505584	1.44660692	0.88946577	1.50993349

Table 9 reports the neighborhood size, the number of clusters created and errors for both the protocols. K is determined as the ratio between the total number of users in the dataset and the fixed neighborhood size. The neighborhood sizes 20, 30, 40, 50, 100 (and consequently the K values 5, 3, 2, 2, 1) are chosen in order to allow comparison with the results of the baseline presented in Experiment 1. For example, in dataset 'Action' (Table 4) there are 100 users; when the neighborhood size is set to 20, we created 5 clusters (100/20) of positive profiles and 5 clusters of negative profiles. The neighbors of the active user are chosen by using best-n-neighbors technique among the users in the same clusters as the active user. Among these, we do not consider the 20% users belonging to the test set.

3.2.3 Experiment 3

Results of the previous experiment suggest that exploiting user profiles in defining like-minded users, might improve the accuracy of recommendations. The third experiment differs from Experiment 2 only in the fact that clustering is applied to synset-based profiles described in Sect. 2.3.1.

Results averaged over all categories are reported in Table 10. Figures 10–13 summarize results of this experiment and compare MAE and MSE values obtained using the clustering method on the set of keyword-based user profiles and synset-based profiles.

Table 10 Performance of the collaborative filtering algorithm using clusters of synset-based profiles for the neighborhood formation [Average over all Categories]

Neigh. size	Training/Test protocol			All But 1 protocol	
	# clusters	MAE	MSE	MAE	MSE
20	5	0.84433210	1.40726048	0.83692554	1.46222503
30	3	0.86957071	1.41518053	0.85981724	1.46982011
40	2	0.86425209	1.41597572	0.85782782	1.48259789
50	2	0.88477027	1.42471562	0.87850066	1.48672140
100	1	0.87595402	1.41770222	0.85055761	1.47983302
Avg.		0.86777584	1.41616692	0.85672577	1.47623949

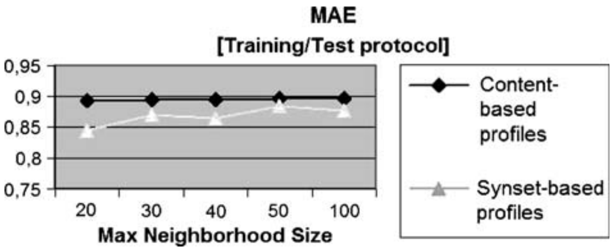

Fig. 10 A comparison of MAE values with the use of keyword-based and synset-based profiles [Training/Test protocol]

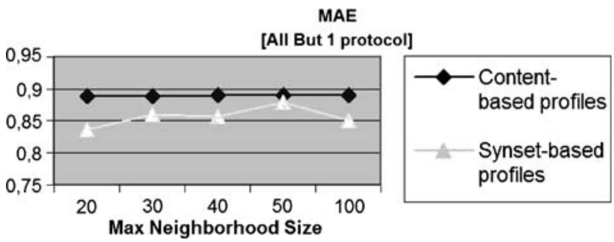

Fig. 11 A comparison of MAE values with the use of keyword-based and synset-based profiles [All But 1 protocol]

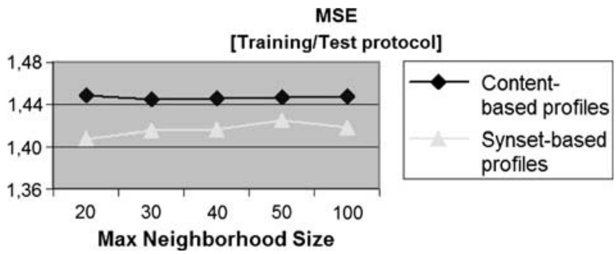

Fig. 12 A comparison of MSE values with the use of keyword-based and synset-based profiles [Training/Test protocol]

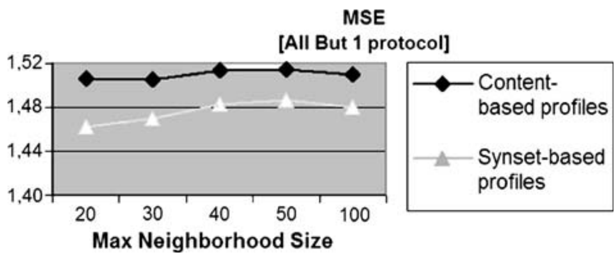

Fig. 13 A comparison of MSE values with the use of keyword-based and synset-based profiles [All But 1 protocol]

3.2.4 Discussion of results

Results obtained by the first experiment correspond to the accuracy of a classic collaborative filtering algorithm that exploits only ratings given by users for computing their similarity and thus recommendations. The most important result we can observe is that recommendations produced are fairly accurate. The method using only the Pearson's correlation coefficient for computing users' similarity is accurate even if it does not use any additional information coming from user profiles (Table 8). This is not surprising since the EachMovie dataset shows a coverage over ratings (percentage of items for which a filtering algorithm can provide predictions or make recommendations) that tends to be higher than 99,97% (Lops 2005), and this might not be representative of real word situations, with very sparse data, with a large portion of cold start users and of items rated just by one user (Massa 2006).

The second and third experiment report results of the hybrid recommender proposed in the paper. We expect that, by aggregating users on the ground of their profiles, a better selection of like-minded users is achieved. Even if there is not much room for improvement, results obtained by using the *training/test* protocol highlight an improvement both for MAE and MSE values achieved by using clusters of synset-based profiles over those obtained using the other techniques. The same improvement has been observed for the MAE in the experiments conducted using the *All But* 1 protocol. A surprising result can be noted by comparing the MSE values registered in all the three experiments using the *All But* 1 protocol. The MSE value of predictions made by collaborative filtering based on Pearson's correlation is lower than the other ones. We analyzed in more detail the results of each single genre dataset and we discovered that the worst performance is on dataset *Animation*. This is mainly due to two reasons: (1) The low number of ratings available; (2) The specificness of the content, as reported in Sect. 2.5.2, where we analyze the poor performance of the content-based classifiers. This is a clear indication of a possible drawback of the proposed hybrid method: When the accuracy of profiles is not adequate (in this case, $F1$ is under 65%), profile-based neighborhood formation is affected by errors, thus a neighborhood formation based on rating style works better.

The other advantage of using clusters is that the process of neighborhood formation is more efficient since clusters are created by an off-line procedure.

Results of the second and third experiment have been compared in order to understand whether the difference between the methods was statistically significant. We compared results obtained for the 10 genre EachMovie datasets, by considering the same neighborhood size. This means that we carried out statistical tests to compare results obtained by systems over the 10 datasets when the neighborhood size is set to 20, 30, 40, 50 and 100. For pairwise comparison of methods, the non-parametric Wilcoxon two-sample paired signed rank test was used, requiring a significance level $\alpha < 0.05$. Results showed that the difference between MAE and MSE values obtained by using keyword-based and synset-based profiles in the neighborhood selection process is statistically significant in favor of synset-based profiles.

These results corroborate the initial hypothesis that a better understanding of users improves recommendations.

3.3 Advantages of the approach

The main advantage of the proposed hybrid technique is the effective strategy adopted for finding better neighbors. In pure collaborative filtering, similarity between users is determined by co-rated items; the novel hybrid recommender computes similarity on synset-based profiles, so users do not need to have co-rated items to be considered similar: An overlap of synsets in positive and/or negative profiles is the only requirement. The effect of using synset-based profiles in finding better neighbors is an improvement of MAE and MSE compared to the use of both Pearson's correlation coefficient and keyword-based profiles.

Furthermore, the proposed hybrid recommender overcomes some shortcomings of pure collaborative filtering systems reported in Sect. 1:

- Sparsity Problem: We interpreted the improvement of MAE and MSE observed from the experiments reported in the previous section as a direct consequence of the neighborhood formation strategy proposed. Even if we did not perform a specific evaluation session devoted to this purpose, our feeling is that this improvement is particularly evident in case of data sparsity, when the strategy based on the Pearson's correlation coefficient is more likely to fail.
- Scalability Problem: The novel hybrid system tries to solve the scalability problem through an off-line clustering process that groups users sharing the same interests or dislikes. When recommendations have to be produced, the selection of a neighborhood for a user is immediate: In order to pick up neighbors, just a unique access to the cluster(s) the user belongs to is needed. Furthermore, the bisecting k-means algorithm used for clustering the set of user profiles is very efficient.
- Lack of Transparency Problem: The use of synset-based profiles to select the neighborhood of users gives the possibility to understand why some users have been selected for producing recommendations. Profiles are explicitly represented by senses instead of words, thus a certain level of system transparency has been added. For example, the concepts in the profiles of the neighbors of the active user could be used to explain to him/her why he/she is considered similar to other users. The idea is that providing the active user with this information is more transparent than giving him/her a list of common rated items or a simple "cryptic" similarity score.

To sum up, the hybrid system proposed has new advanced properties compared to systems presented in Sect. 1.2. To the best of our knowledge, the clustering of synset-based profiles for the process of neighborhood selection is a novel contribution in the area of collaborative filtering systems.

4 Conclusions and future work

Recommender systems facilitate the natural social recommendation behavior and alleviate the pressure of information overload. The traditional collaborative filtering approach to build recommender systems ignores proximities between users if they did not rate any *common* item. In order to overcome this limitation, we proposed a content-collaborative hybrid recommendation approach that:

- Integrates general and shared linguistic knowledge in the process of learning user profiles in order to infer sense-based profiles able to represent user interests in a more effective way compared to classical keyword-based profiles; experiments reported in Sect. 2.5.2 show that migrating from words to concepts produces a classification accuracy improvement for both content-based methods presented in the paper (+2% $F1$ improvement for RocchioProfiler, +8% $F1$ improvement for ITR);

– Exploits sense-based profiles to form neighborhood for the active user in order to discover similarities among users, even if they did not rate any common item. Experimental results reported in Sect. 3.2 highlight the improvement in the accuracy of collaborative recommendations obtained by selecting like-minded users according to sense-based profiles (7% MAE reduction on average).

The general conclusion is that the lexical knowledge approach based on WordNet requires some improvements to learn more accurate semantic profiles because of the unsuccessful recognition of domain entities and of specialized terminology (this is the case of the dataset *Animation*). Thus, an improvement will concern the integration of domain-dependent knowledge sources, such as domain ontologies, in the synset-based linguistic approach, in order to obtain a more powerful knowledge-based approach. Another improvement concerns the comprehensive testing of a more sophisticated word sense disambiguation algorithm, based on the idea of combining different strategies to disambiguate nouns, verbs, adjectives and adverbs, which has already been developed (Semeraro et al. 2007).

We foresee a further possible improvement of our method for selecting like-minded users: In the neighborhood formation process, interests are maintained separated from disinterests, but they weigh equally in Algorithm 2. As people could dislike similar things but could have different interests, Algorithm 2 could be modified by introducing a strategy to weigh interests higher than disinterests.

Finally, in order to understand the effectiveness of the proposed strategy, we are also planning to validate the proposed solution against a large real world dataset derived from Epinions.com, a consumers opinion site where users can review items (such as cars, books, movies, software, ...).

Acknowledgements The authors wish to thank Anna Rowe for her help to correct the manuscript. This research was partially funded by the European Commission under the 6th Framework Programme IST Integrated Project VIKEF No. 507173, Priority 2.3.1.7: Semantic Based Knowledge Systems, http://www.vikef.net and by the DELOS Network of Excellence on Digital Libraries, Priority: Technology-enhanced Learning and Access to Cultural Heritage, Contract No. G038-507618, http://delos.info.

References

Adomavicius, G., Sankaranarayanan R., Sen S., Tuzhilin A.: Incorporating contextual information in recommender systems using a multidimensional approach. ACM Trans. Inf. Sys. **23**(1), 103–145 (2005)

Adomavicius, G., Tuzhilin, A.: Towards the next generation of recommender systems, a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowledge Data Eng. **17**(6):734–749 (2005)

Asnicar, F., Tasso, C.: ifWeb: a prototype of user model-based intelligent agent for documentation filtering and navigation in the word wide web. In: Tasso, C., Jameson, A., Paris, C.L. (eds.) Proceedings of the First International Workshop on Adaptive Systems and User Modeling on the World Wide Web, Sixth International Conference on User Modeling, pp. 3–12. Chia Laguna, Sardinia, Italy (1997)

Balabanovic, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Commun. ACM **40**(3), 66–72 (1997)

Basu, C., Hirsh, H., Cohen, W.: Recommendation as classification: using social and content-based information in recommendation. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98) and of the Tenth Conference on Innovative Applications of Artificial Intelligence (IAAI-98), pp. 714–720. Menlo Park, AAAI Press (1998)
Billsus D., Pazzani, M.J.: Learning collaborative information filters. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 46–54. Morgan Kaufmann, San Francisco, CA (1998)
Bloedhorn, S., Hotho, A.: Boosting for Text Classification with Semantic Features. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Mining for and from the Semantic Web Workshop, pp. 70–87. Seattle, WA, USA, (2004)
Bradley, P.S., Fayyad, U.M.: Refining initial points for K-means clustering. In: Shavlik, J. (ed.) Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98), pp. 91–99. California, Morgan Kaufmann (1998)
Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Cooper, G.F., Moral, S. (eds.) Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, pp. 43–52. Morgan Kaufmann (1998)
Budanitsky, A., Hirst, G.: Semantic distance in WordNet: an experimental, application-oriented evaluation of five measures. In: Proceedings of the Workshop on WordNet and other Lexical Resources, Second Meeting of the North American Chapter of the Association for Computational Linguistics, pp. 29–34. Pittsburgh, PA (2001)
Burke, R.: Hybrid recommender systems: survey and experiments. User Model. User-Adapted Interaction **12**(4), 331–370 (2002)
Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M.: Combining content-based and collaborative filters in an online newspaper. In: Proceedings of ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation. Berkeley, California, USA, ACM Press, New York, NY, USA (1999)
Cutting, D., Karger, D., Pedersen, J., Tukey, J.: Scatter/gather: a cluster based approach to browsing large document collection. In: Proceedings of the Fifteenth ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 318–329, Copenhagen, Denmark, ACM Press, New York, NY, USA (1992)
Degemmis, M.: Learning User Profiles from Text for Personalized Information Access. Ph.D. thesis, Department of Informatics, University of Bari (2005)
Degemmis, M., Lops, P., Semeraro, G.: WordNet-based Word Sense Disambiguation for Learning User Profiles. In: Proceedings of the Second European Web Mining Forum, ECML/PKDD 2005, pp. 16–27. Porto, Portugal, (2005)
Degemmis, M., Lops, P., Semeraro, G., Costabile, M., Guida, S., Licchelli, O.: Improving collaborative recommender systems by means of user profiles. In: Karat, C.-M., Blom, J., Karat, J. (eds.) Designing personalized user experiences in eCommerce, pp. 253–274. Kluwer Academic (2004)
Delgado, J., Ishii, N.: Memory-based weighted-majority prediction for recommender systems. In: Proceedings of the ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation. Berkeley, California, USA, ACM Press, New York, NY, USA (1999)
Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
Hartigan, J.: Clustering Algorithms. John Wiley & Sons, New York, NY (1975)
Hartigan, J., Wong, M.: Algorithm AS136: a k-means clustering algorithm. Appl. Stat. **28**, 100–108 (1979)
Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 230–237. Berkeley, California, USA, ACM Press New York, NY, USA. (1999)
Herlocker, J.L., Konstan, J.A., Riedl, J.: Explaining collaborative filtering recommendations. In: Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work, pp. 241–250. Philadelphia, Pennsylvania, United States, ACM Press New York, NY, USA. (2000)
Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004)
Hotho, A., Staab, S., Stumme, G.: Wordnet improves text document clustering. In: Proceedings of the Semantic Web Workshop at SIGIR 2003, 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Toronto, Canada, ACM Press New York, NY, USA (2003)

Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 1137–1145. San Mateo, CA: Morgan Kaufmann (1995)

Larsen, B., Aone, C.: Fast and Effective text mining using linear-time document clustering. In: Chaudhuri, S. Madigan, D. (eds.) Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 16–22. N.Y., ACM Press, (1999)

Leacock, C., Chodorow, M.: Combining local context and WordNet similarity for word sense identification. In: Fellbaum, C. (ed.) WordNet: An Electronic Lexical Database, pp. 266–283. MIT Press. (1998)

Lee, W.S.: Collaborative learning for recommender systems. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 314–321. Morgan Kaufmann, San Francisco, CA (2001)

Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comp. **7**(1), 76–80 (2003)

Lops, P.: Hybrid recommendation techniques based on user profiles. Ph.D. thesis, Department of Informatics, University of Bari (2005)

Magnini, B., Strapparava, C.: Improving user modelling with content-based techniques. In: Proceedings of the Eighth International Conference on User Modeling, pp. 74–83. Sonthofen, Germany, Springer (2001)

Manning, C., Schütze, H.: Foundations of statistical natural language processing, Chapt. 7: Word Sense Disambiguation, pp. 229–264. The MIT Press, Cambridge, US (1999)

Massa, P.: Trust-aware decentralized recommender systems. Ph.D. thesis, International Doctorate School in Information and Communication Technologies, University of Trento (2006)

Mavroeidis, D., Tsatsaronis, G., Vazirgiannis, M., Theobald, M., Weikum, G.: Word sense disambiguation for exploiting hierarchical thesauri in text classification. In: Proceedings of the Ninth European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), vol. 3721 of Lecture Notes in Computer Science, pp. 181–192. Porto, Portugal, Springer (2005)

McCallum, A., Nigam, K.: A comparison of event models for naïve Bayes text classification. In: Proceedings of the AAAI/ICML-98 Workshop on Learning for Text Categorization, pp. 41–48. AAAI Press (1998)

Melville, P., Mooney, R.J., Nagarajan, R.: Content-Boosted collaborative filtering for improved recommendations. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI-02), pp. 187–192. Menlo Parc, CA, USA, AAAI Press (2002)

Miller, G.: WordNet: an on-line lexical database. Int. J. Lexicogr. **3**(4), 235–312. (Special Issue) (1990)

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

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

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

Nakamura, A., Abe, N.: Collaborative filtering using weighted majority prediction algorithms. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 395–403. Morgan Kaufmann (1998)

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

Patwardhan, S., Banerjee, S., Pedersen, T.: Using measures of semantic relatedness for word sense disambiguation. In: Gelbukh, A.F. (ed.) Computational Linguistics and Intelligent Text Processing, Fourth International Conference, CICLing 2003, Proceedings, vol. 2588 of Lecture Notes in Computer Science, pp. 241–257. Springer (2003)

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

Pazzani, M.J.: A Framework for collaborative, content-based and demographic filtering. Artificial Intelligence Rev. **13**(5–6), 393–408 (1999)

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, pp. 175–186. Chapel Hill, North Carolina, ACM Press New York, NY, USA (1994)

Resnick, P., Varian, H.: Recommender systems. Commun. ACM **40**(3), 56–58 (1997)

Resnik, P.: WordNet and class-based probabilities. In: Fellbaum, C. (ed.) WordNet: An Electronic lexical database, pp. 239–263, MIT Press (1998)

Rocchio, J.: Relevance feedback information retrieval. In: Salton, G. (ed.) The SMART Retrieval System – Experiments in Automated Document Processing, pp. 313–323. Prentice-Hall, Englewood Cliffs, NJ (1971)
Rodriguez, M.d.B., Gomez-Hidalgo, J.M., Diaz-Agudo, B.: Using WordNet to complement training information in text categorization. In: Second International Conference on Recent Advances in NLP, pp. 150–157 (1997)
Rosso, P., Ferretti, E., Jimenez, D., Vidal, V.: Text categorization and information retrieval using WordNet synsets. In: Sojka, P., Pala, K., Smrž, P., Fellbaum, C., Vossen, P. (eds.) Proceedings of the Second International WordNet Conference, pp. 299–304. Masaryk University Brno, Czech Republic (2004)
Sarwar, B.M., Karypis, G., Konstan, J., Reidl, J.: Recommender systems for large-scale E-Commerce: scalable neighborhood formation using clustering. In: Proceedings of the Fifth International Conference on Computer and Information Technology (ICCIT). Dhaka, Bangladesh (2002)
Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Analysis of recommendation algorithms for E-commerce. In: ACM Conference on Electronic Commerce, pp. 158–167. Minneapolis, Minnesota, USA, (2000a)
Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Application of dimensionality reduction in recommender systems: a case study. In: Proceedings of the WebKDD 2000 Workshop at the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'00). Boston, MA (2000b)
Schwab, I., Kobsa, A., Koychev, I.: Learning User Interests through Positive Examples using Content Analysis and Collaborative Filtering. Draft from Fraunhofer Institute for Applied Information Technology, Germany (2001)
Scott, S., Matwin, S.: Text classification using WordNet hypernyms. In: Harabagiu, S. (ed.) COLING-ACL Workshop on Usage of WordNet in NLP Systems, pp. 45–51. Somerset, New Jersey, Association for Computational Linguistics (1998)
Sebastiani, F.: Machine learning in automated text categorization. ACM Comp. Surveys **34**(1), 1–47 (2002)
Semeraro, G., Degemmis, M., Lops, P., Basile, P.: Combining learning and word sense disambiguation for intelligent user profiling. In: Twentieth International Joint Conference on Artificial Intelligence, 2007. Hyderabad, India. (Forthcoming) (2007)
Shardanand, U., Maes, P.: Social information filtering: algorithms for automating/word of mouth. In: Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, vol. 1, pp. 210–217. Denver, Colorado, United States (1995)
Soboroff, I., Nicholas, C.: Combining content and collaboration in text filtering. In: IJCAI'99 Workshop: Machine Learning for Information Filtering, pp. 86–91. Stockholm, Sweden (1999)
Stevenson, M.: Word Sense Disambiguation: The Case for Combinations of Knowledge Sources. CSLI Publications, Stanford, CA, USA (2003)
Terveen, L., Hill, W.: Human-computer collaboration in recommender systems, pp. 223–242. In: Carroll, J. (ed.) HCI on the new Millennium, Addison Wesley (2001)
Theobald, M., Schenkel, R., Weikum, G.: Exploting structure, annotation, and ontological knowledge for automatic classification of XML data. In: Proceedings of the Seventh International Workshop on Web and Databases, pp. 1–6. Maison de la Chimie, Paris, France (2004)
Ungar, L., Foster, D.: Clustering methods for collaborative filtering. In: Proceedings of the Workshop on Recommendation Systems. AAAI Press, Menlo Park California (1998)
Vozalis, E., Margaritis, K.G.: Analysis of recommender systems algorithms. In: Proceedings of the Sixth Hellenic European Conference on Computer Mathematics and its Applications (HERCMA). Athens, Greece (2003)
Witten, I., Bell, T.: The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression. IEEE Trans. Inf. Theory **37**(4), 1085–1094 (1991)
Yang, Y. Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Fisher, D.H. (ed.) Proceedings of ICML-97, Fourteenth International Conference on Machine Learning, pp. 412–420. Nashville, US, Morgan Kaufmann Publishers, San Francisco, US (1997)
Yao, Y.Y.: Measuring retrieval effectiveness based on user preference of documents. J. Am. Soc. Inf. Sci. **46**(2), 133–145 (1995)

Authors' vitae

Dr. Marco Degemmis is Research Assistant at the Department of Informatics, University of Bari (Italy). He completed his Ph.D. in Informatics in 2005 at the University of Bari, Under the supervision of Prof. Giovanni Semeraro, with a thesis on "Learning User Profiles from Text for Personalized Information Access". His current research interest include natural language processing, machine learning techniques for text categorization, information retrieval, and personalized information filtering.

Dr. Pasquale Lops is Assistant Professor at the Department of Informatics, University of Bari (Italy). He completed his Ph.D in 2005 at the University of Bari, under the supervision of Prof. Giovanni Semmeraro, with a dissertation on "Hybrid Recommendation Techniques based on User Profiles". His primary interests lie in the areas of machine learning, recommender systems, digital libraries, user modeling and universal access. He is particularly interested in enabling computer access for all.

Prof. Giovanni Semeraro is Associate Professor at the Department of Informatics, University of Bari (Italy), where he teaches programming languages, formal languages and compilers and enterprise knowledge management". His research activity mainly concerns machine learning, sematic web and personalization. His research interests include logical and algebraic foundations of machine learning for inductive reasoning, extraction of dynamic user profiles, web and usa ge mining, revision of logical theories and application of machine learning techniques to user modeling and digital libraries. He has published over 150 papers in international journals, books and conference proceedings, and is the author of a textbook on formal language theory.