Computers in Human Behavior xxx (2015) xxx–xxx

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

A collaborative user-centered framework for recommending items in Online Social Networks

Francesco Colace [a,*], Massimo De Santo [a], Luca Greco [b], Vincenzo Moscato [c], Antonio Picariello [c]

[a] DIEM, University of Salerno, Fisciano, Italy
[b] DIIN, University of Salerno, Fisciano, Italy
[c] DIETI, University of Naples "FEDERICO II", Italy

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Recommender systems
Sentiment analysis

ABSTRACT

Recommender Systems are more and more playing an important role in our life, representing useful tools helping users to find "what they need" from a very large number of candidates and supporting people in making decisions in various contexts: what items to buy, which movie to watch, or even who they can invite to their social network, etc. In this paper, we propose a novel collaborative user-centered recommendation approach in which several aspects related to users and available in Online Social Networks – i.e. *preferences* (usually in the shape of items' metadata), *opinions* (textual comments to which it is possible to associate a sentiment), *behavior* (in the majority of cases logs of past items' observations made by users), *feedbacks* (usually expressed in the form of ratings) – are considered and integrated together with *items' features* and *context information* within a general framework that can support different applications using proper customizations (e.g., recommendation of news, photos, movies, travels, etc.). Experiments on system accuracy and user satisfaction in several domains shows how our approach provides very promising and interesting results.

1. Introduction

In the era of *Big Data* we are assisting to an explosive and amazing increase of digital information, and as a consequence, huge data collections of different nature are widely available to a large population of users. In addition, the widespread diffusion of the most popular social networks, multimedia repositories, news archives, travel websites, e-commerce portals, and so on, has constrained users necessarily to deal with this ocean of information to find "what they need".

In the last decade, *Recommender Systems* have been introduced to facilitate the browsing of such collections, thus realizing the well known transition in the Web from the *search* to the *discovery* paradigm.

Generally, recommender systems help people in retrieving information that match their preferences by recommending products or services from a large number of candidates, and support people in making decisions in various contexts: what items to buy, which movie to watch, which music to listen, what travels to do, or even who they can invite to their social network, just to make some examples. In particular, *Online Social Networks* (either a general-purpose social network, such as Facebook, or a domain-specific recommendation social network, such as Flixster for movie recommendation and Epinions for a wide range of product recommendation) can take the most immediate advantage of recommendation facilities, leveraging in different ways user experiences and interactions within the social community to suggest objects of interest.

Formally, a recommender system deals with a set of *users* $U = \{u_1, \ldots, u_m\}$ and a set of *items* $O = \{o_1, \ldots, o_n\}$. For each pair (u_i, o_j), a recommender can compute a *score* (or a *rank*) $r_{i,j}$ that measures the expected interest of user u_i in item o_j (or the expected utility of item o_j for user u_i), using a *knowledge base* and a *ranking* algorithm that generally could consider different combinations of the following characteristics: (i) user preferences and past behavior, (ii) preferences and behavior of the user community, (iii) items' features and how they can match user preferences, (iv) user feedbacks, and (v) context information and how recommendations can change together with the context.

In our opinion, modern recommending applications have to take into account in some way all the above characteristics to provide useful and reliable recommendations both for virtual and physical environments. To this goal, the last generation of

* Corresponding author.
E-mail addresses: fcolace@unisa.it (F. Colace), desanto@unisa.it (M. De Santo), lgreco@unisa.it (L. Greco), vmoscato@unina.it (V. Moscato), picus@unina.it (A. Picariello).

http://dx.doi.org/10.1016/j.chb.2014.12.011
0747-5632/© 2014 Elsevier Ltd. All rights reserved.

recommender systems is usually composed by one or more of the following components (Ricci, Rokach, Shapira, & Kantor, 2011).

A *pre-filtering* module that selects for each user u_i a subset $O_i^c \subset O$ containing items that are good candidates to be recommended; such items usually match user preferences and needs.

A *ranking* module that assigns w.r.t. user u_i a rank $r_{i,j}$ to each candidate item o_j in O_i^c using the well-known recommendation techniques (i.e., *content-based*, *collaborative filtering* and *hybrid* approaches) that can exploit in several ways items' features and users' preferences, feedbacks (in the majority of cases in terms of *ratings*) and behavior.

A *post-filtering* module that dynamically excludes, for each user u_i, some items from the recommendations' list; in this way, a new set $O_i^f \subseteq O_i^c$ is obtained on the base of user feedbacks and other contextual information (such as data coming from the interactions between the user and the application). Eventually, depending on the applications, recommended items can be arranged in *groups* according to additional constraints.

In this paper, we propose a novel collaborative and *user-centered* approach that provides *social* recommendations, capturing and exploiting users' *opinions* and *sentiments* about items, as a sort of additional ranking criterion. People opinions have always driven human choices and behaviors and an important part of information gathering behavior has always been to find out "what other people think". It has to be considered as a fundamental aspect in the design of a modern recommender system, especially for online social networks.

Thus, in our approach several aspects related to users – i.e., *preferences* (usually coded in the shape of items' metadata), *opinions* (textual comments to which it is possible to associate a particular sentiment), *behavior* (in the majority of cases logs of past items' observations made by users), *feedbacks* (usually expressed in the form of ratings) – are considered and integrated together with *items' features* and *context information* within a general and unique recommendation framework that can support different applications using proper customizations (e.g., recommendation of news, photos, movies, travels, etc.), overcoming problems related to the availability and quality of user profiles and ratings of classical recommendation techniques.

In other words, it is the user with his/her preferences (in the pre-filtering stage) and actions (in the post-filtering stage) to drive the recommendation process towards the real useful items among those that the user community considers the "best ones" (computed in the ranking stage), as in a collaborative learning approach (Colace, De Santo, & Greco, 2014b), where a user "learns from the others" the item utility.

The paper is organized as follows. Section 2 presents some motivating examples for our work, while Section 3 discusses the state of the art of other similar systems focusing on the aspect of social recommendations. Section 4 provides a functional overview of our system and describes the proposed strategy for recommendation. Section 5 illustrates two system customizations in the domains of movies and travels recommendation respectively, and reports some implementation details. Section 6 reports experimental results, and provides a comparison with other recommendation techniques. Finally, Section 7 gives some concluding remarks and discusses future work.

2. Motivating examples

Let us consider two typical scenarios where an effective social recommender system would be desirable.

First, we consider a user that desires to have information about the coming soon movies. In this case, the list of suggested items (that can be easily extracted from portals as *IMDB*) should consider the following information: *user preferences* in terms of movies' metadata (e.g., favorite genre, director, stars, etc.); *item features* (i.e., movies' metadata) and their similarity (e.g., a semantic relatedness based on a movie taxonomy); *user behavior* in terms of the sequence of items that in the past the community of users have observed and positively rated; *user feedbacks* in terms of the user community ratings; *user opinions* in terms of the average sentiment that items have aroused on the user community; *context information* in this case in terms of item features that satisfy the search criteria (e.g., coming soon movies that are showing in theaters near the user) or that have a certain "similarity" with respect to the item user has selected and is currently watching.

We can image a user that prefers the adventure and fantasy genres and has among his/her favorite actors Ian McKellen and Hugh Jackman; the system can initially suggest as first items to watch the X-Men saga movies, together with other titles (the majority will be adventure and fantasy). After the pre-filtering stage, the candidate items matching user preferences are initially ranked on the base of the related social *popularity* (e.g., number of accesses through past users' paths, average rating and sentiment from users' reviews, etc.). Successively if the user chooses to read more information about one of X-Men movies and rates it positively, a list of filtered items is then provided with the most popular movies that are similar in terms of metadata with the X-Men movie. Eventually, if the user chooses to limit the search to the coming soon movies (constraint on a metadata value) and select his/her position as additional context information, all the best movies – according to a social view – matching user preferences and being shown in the next days in theaters near the user will be finally proposed (e.g., "X-Men: Days of Future Past" and "Captain America: The Winter Soldier").

As second typical scenario is a touristic application: let us consider a user that desires to plan her/his vacation in Italy (and in particular in Naples) selecting on the Web some available travel packages. Also in this case, the list of suggested items (that can be easily extracted from portals as *tripadvisor*) should consider: *user preferences* in terms of travel-related content metadata (e.g., for hotels class, price, services, location, etc., for flights price, services, etc. and for restaurants price, kind of cooking, dining options, etc.) and trip constraints (e.g., vacation period and destinations); *item features* (i.e., travel-related content metadata) and their similarity (e.g., a semantic relatedness based on one or more available taxonomies); *user behavior* in terms of the sequence of items – of the same kind – that in the past the community of users have observed and positively rated; *user feedbacks* in terms of the user community ratings; *user opinions* in terms of the average sentiment that items have aroused on the user community; *context information* in terms of item features that satisfy additional criteria (e.g., cheap flight departing from airports near user, but conformable hotels near the sea) or that have a certain "similarity' with respect to the items user has selected and is watching.

In particular, a tourist typically starts the interaction with a system selecting the trip period and, of course, the desired destination. If the user prefers hotels in front of the ocean and desires to eat pizza in a famous restaurant, the system can initially suggest a set of travel packages having as destination Naples and containing three kinds of items: an accommodation in a hotel with ocean view (in the chosen period), a pizza degustation in a famous restaurant and a flight departing from the most famous European cities. After the pre-filtering stage, the candidate items matching user preferences are initially ranked on the base of the related social *popularity* (e.g., number of accesses across past users' paths, average rating and sentiment from users' reviews, etc.) and the travel packages are scored using the average ratings of component items. Successively if the user chooses to read more information about one or more items in a package and rate them positively, a new list of filtered items is then provided with the most popular travel-related contents that are similar in terms of metadata with the selected items. Subsequently, the list of suggested packages is updated and will consider only those containing an accommodation, a pizza degustation and a flight all belonging to the filtered items list. Eventually if the user chooses to add new constraint as low cost flights, avoid early morning flights and comfortable hotels (additional constraint on metadata values) and select his/her position as context information, all the packages matching user preferences with an accommodation in 4 or 5 stars hotels, a pizza degustation and an afternoon flight from airports near to the actual user position will be finally proposed. Alternatively, our system can be exploited to create personalized packages separately recommending the different kinds of contents and finally grouping them on the base of further constraints.

3. Related works

Recommender Systems represent a meaningful response to the problem of information overload since the mid-1990s (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994) when early works on this topic have been proposed. The aim of such systems is to predict user's preferences and make meaningful suggestions about items that could be of interest (Ricci et al., 2011).

In *content-based* approach, the system recommends an item to a user relying on the ratings made by the user himself for *similar* items in the past (Pazzani & Billsus, 2007). In recent times, some improvements, such as a deeper user profile analysis (Su & Khoshgoftaar, 2009) and the use of probabilistic methods (Yildirim & Krishnamoorthy, 2008), have been introduced together with some attempts to apply the content based approach to multimedia data (Hijikata, Iwahama, & Nishida, 2006; Maidel, Shoval, Shapira, & Taieb-Maimon, 2008; Musial, Juszczyszyn, & Kazienko, 2008). However, a critical drawback of this approach is *overspecialization*, since the systems only recommend items similar to those already rated by the user. In *collaborative filtering* (Adomavicius & Tuzhilin, 2005), the recommendation is performed by filtering and evaluating items with respect to ratings from other users. Typically, users are asked to rate items and a similarity between their profiles is also computed to be used as a weight when making recommendations for highly rated items (Su & Khoshgoftaar, 2009). Ratings can be attributed in different ways and collected by explicitly asking users or implicitly tracking their actions (Albanese, d'Acierno, Moscato, Persia, & Picariello, 2013). Two basic methods – *passive* and *active* filtering – are exploited for filtering and recommending items together with *nearest neighbor* techniques (Koren, 2008; Ramakrishnan, Keller, Mirza, Grama, & Karypis, 2001). An important limitation of collaborative filtering systems is the *cold start problem*, that describes situations in which a recommender is unable to make meaningful recommendations due to an initial lack of ratings. A particular kind of collaborative approach is the *collaborative competitive filtering* that aims at learning user preferences by modeling the choice process in recommender systems (Yang, Long, Smola, Zha, & Zheng, 2011).

Content-based filtering and collaborative filtering may be then combined in the so called *hybrid* approach that helps to overcome limitations of each method (Schein, Popescul, Ungar, & Pennock, 2002).

Eventually, a recommendation strategy should be able to provide users with relevant information depending on the *context* (Dourish, 2004; Kabassi, 2013; Karatzoglou, Amatriain, Baltrunas, & Oliver, 2010) (i.e. user location, observed items, weather and environmental conditions, etc.) as in *Context Aware Recommendation Systems*. In the *Contextual Pre-filtering* techniques context information is used to initially select the set of relevant items, while a classic recommender is used to predict ratings. In *Contextual Post-filtering* approaches context is used in the last step of the recommending process to contextualize the output of a traditional recommender.

Finally, a recent category of recommenders, named *Large Scale Recommender Systems* (LSRS) (Yu, Hsieh, Si, & Dhillon, 2013), calls for new capabilities of such applications to deal with very large amount of data with respect to scalability and efficiency issues.

Performance of classical recommender systems is strictly related to the *availability* and *quality* of user profiles and ratings (Sarwar, Karypis, Konstan, & Riedl, 2001): the density of the available ratings in commercial systems is often less than 1% and the proliferation of *fake* users can arise *malicious ratings* (Sinha & Swearingen, 2001).

An important improvement for traditional recommender systems to overcome such problems lies in the possibility to embed *social* elements into a recommendation strategy (Zhou, Xu, Li, Josang, & Cox, 2012). In fact, the great increase of user-generated content in social networks, such as product reviews, tags, forum discussions and blogs, has been followed by a bunch of valuable user opinions, perspectives or tastes towards items or other users, that are useful to build enhanced user profiles. In such a context, customer *opinion summarization* and *sentiment analysis* (Ding, Liu, & Yu, 2008; Zhuang, Jing, & Zhu, 2006) techniques represent effective augmentations to traditional recommendation strategy, for example by not recommending items that receive a lot of negative feedbacks (Zhou et al., 2012). Indeed, a lot of attention is nowadays being payed from vendors to consumer's voices because of the great influence they may have on the opinions and decisions of others (Kurilovas, Juskeviciene, Kubilinskiene, & Serikoviene, 2014; Stan, Muhlenbach, & Largeron, 2014) and some companies already provide several *opinion mining* services (e.g., Amazon, Epinions, etc.).

In recent times, some works have been proposed to extend traditional collaborative filtering with the use of sentiment analysis techniques, thus providing effective improvement to system performances (Drigas, Ioannidou, Kokkalia, & Lytras, 2014; Leung, Chan, & Chung, 2006): most of them make use of *Part Of Speech* (POS) *tagging* techniques and aim at refining standard collaborative filtering ranking outcomes in terms of numerical scales to take into account user community opinions.

The work in Singh, Mukherjee, and Mehta (2011) proposes a recommender system for movies that combines collaborative filtering with sentiment: here sentiment classification is performed through both *Naïve Bayes classifier* and unsupervised *semantic orientation* approach. Ganu, Kakodkar, and Marian (2013) proposes a method to derive a text-based rating from the body of restaurant reviews: users are grouped together using soft clustering techniques based on the topics and sentiments that appear in the reviews to obtain better review score predictions than those derived by a coarse numerical star rating. Pappas and Popescu-Belis (2013) focuses on user comments – that are not accompanied by explicit ratings – and their utility for a one-class collaborative filtering task such as bookmarking.

Finally, in Dong, O'Mahony, Schaal, McCarthy, and Smyth (2013) an approach based on opinionated product descriptions that are automatically mined from user-generated product reviews is presented: the recommendation ranking strategy combines similarity and sentiment to suggest products similar to a query product according to the opinion of reviewers.

Similarly to the described approaches, our idea is to exploit sentiment classification techniques – based a *Latent Dirichlet Allocation* (LDA) analysis – of comments and feedbacks from online social networks to refine and enrich a collaborative recommendation strategy that some of the authors have proposed for recommendation in multimedia browsing systems (Albanese, d'Acierno, Moscato, Persia, & Picariello, 2011, 2013; Bartolini et al., 2014). We thus obtained a novel user-centered approach in which several aspect of a user (preferences, opinions, feedbacks, behaviors) are simultaneously considered together with item features and context information within a unique and general framework.

4. A framework supporting social recommendation

4.1. System overview

Fig. 1 describes at a glance an overview of the proposed system. In terms of its main components, that we are describing in the following.

Data to be recommended are retrieved by a *Wrapper* component that is composed by several modules. The *Crawling Interface* is responsible of: (i) periodically accessing to the items' repositories (e.g., *Imdb*, *Flickr*, *tripadvisor*, *Google Images*, *YouTube*, *Facebook*, etc.) and (ii) extracting for each item all the *features* (e.g., metadata, ratings, reviews, etc.) and other information of interest (e.g. user preferences, time-stamped items' observations, etc.). A part of such information will be then exploited by the *User Logs and Preference Analyzer* and *Opinion Analyzer* modules to determine the user behavior graphs with the related profile and the users' mood on the different items.

After the wrapping phase, all the information will be stored in the *Knowledge Base* of the system. In particular, it is composed by: (i) the *User Behaviour Graphs* containing a set of graphs related to the users' browsing sessions, (ii) the *Items DB* containing items with all the related features, (iii) *User Profiles* containing user preferences in terms of items' features and (iv) *Contextual Data* containing some additional context information (e.g. actual user location, weather conditions, etc.).

The *Recommender Engine* provides a set of recommendation facilities for multi-dimensional and interactive browsing of items. Exploiting user preferences, the *Pre-filtering* module selects a set of *candidate* items for recommendation; successively, the *Objects Ranking* module assigns a ranking of such candidates exploiting a proper social strategy (that uses the *Users and Similarity Matrices Computation* module and information on users' opinions).

Finally, the *Constraints Based Post-filtering* module dynamically selects a subset of candidates, on the base of the item that a user is currently watching and context information.

4.2. Recommendation process

The basic idea behind our proposal is that when a user is browsing a particular items' collection, the recommender system: (i) determines a set of useful *candidate* items for the recommendation, on the base of user actual needs and preferences (*pre-filtering stage*); (ii) opportunely assigns to these items a rank, previously computed exploiting items' intrinsic features and users' past behaviors, and using as refinement, social information in the shape of users' opinions and feedbacks (*ranking stage*); and (iii) dynamically, when a user "selects" as interesting one or more of the candidate items, determines the list of most suitable items (*post-filtering stage*), also considering other context information expressed by users in the shape of constraints on items' features.

Eventually, final recommended item can be arranged in specific groups of items considering further constraints.

In the following, we are detailing all the described stages.

4.2.1. Pre-filtering stage using user preferences

In the *pre-filtering* stage, our aim is to select for a given user u_h a subset $O_h^c \subset O$ containing items that are good "candidates" to be recommended: such items usually have to match some (static) user preferences and (dynamic) actual needs.

Each item subjected to recommendation may be represented in different and heterogeneous feature spaces. For instance, a movie may be described by a set of metadata as title, genre, stars, by the list of theaters (characterized by name and location) in which they are coming, by the users' comments and feedbacks and so on. Each of these sets of features contributes to the characterization of the items to different extents.

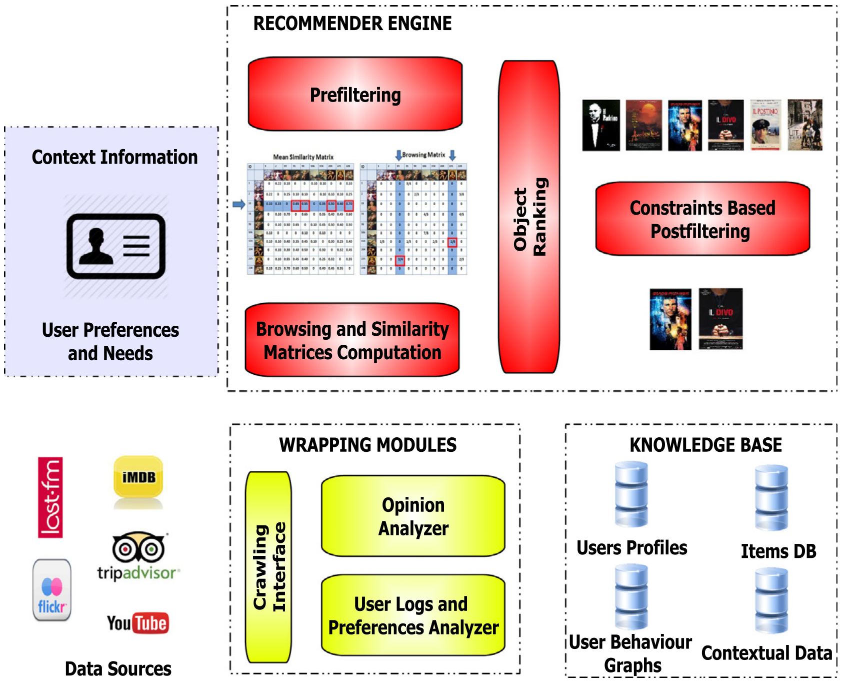


Fig. 1. System overview.

The first step consists in clustering together "similar" items, where the similarity should consider all (or subsets of) the different spaces of features.

To this purpose, we employ *high-order star-structured co-clustering* techniques – that some of the authors have adopted in previous work (Bartolini et al., 2014; Ienco, Robardet, Pensa, & Meo, 2013) – to address the problem of heterogeneous data pre-filtering.

In this context, the same set of items is represented in different feature spaces. Such data represent items of a certain type, connected to other types of data, the features, so that the overall data schema forms a star structure of inter-relationships. The co-clustering task consists in clustering simultaneously the set of items and the set of values in the different feature spaces. In this way we obtain a partition of the items influenced by each of the feature spaces and at the same time a partition of each feature space. The pre-filtering stage leverages the clustering results to select a set of items by using the user's profile, which is modeled as sets of descriptors in the same spaces as the items' descriptors.

Let $O = \{o_1, \ldots, o_n\}$ be the set of items and $\mathcal{F} = \{F^1, \ldots, F^l\}$ a set of l feature spaces. In our recommendation problem, a user u_h is represented as a set of vectors in the same l feature spaces describing the items.

To provide a first candidate list of items to be recommended, we measure the *cosine distance* of the user vectors associated to the k-th space, with the centroids of each item clusters in the k-th space. For each space, the most similar item cluster is chosen leading to l clusters $\{X_1^c, \ldots, X_l^c\}$ of candidate items.

Then, two different strategies can be adopted to provide the pre-filtered list of candidate items O_h^c: (i) *set-union strategy* – the items belonging to the union of all clusters are retained, i.e., $O_h^c = \bigcup_k X_k^c$ and (ii) *threshold strategy* – the items that appears in at least *ths* clusters ($ths \in \{1 \ldots l\}$) are retained.

The first strategy is suitable when user's vectors are associated to very small clusters. In any other situation, the second strategy is the most appropriate.

As a final step, items already visited/liked/browsed by the user are filtered out.

Notice that, thanks to this approach, users are not described by set of items, but by sets of features that characterize the objects they visit, like or browse.

4.2.2. Ranking stage using user behavior and items similarity

The main goal of this stage is to automatically rank the set of items O embedding in a collaborative learning context (user preferences are represented modeling the choice process in recommender system and learnt by users' *browsing behaviors*) their *intrinsic features* (those on the top of which it is possible to introduce a *similarity* notion).

In particular, we use a novel technique that some of the authors have proposed in previous works – combining in a novel manner low and high level features of items, possible past behavior of individual users and overall behavior of the whole user "community" (Albanese et al., 2011; Albanese et al., 2013) – to provide useful recommendations during the browsing of multimedia collections.

Our basic idea is to assume that when an item o_i is chosen after an item o_j in the same user *browsing session* (and both the explored items have been positively rated or have captured attention of users for an adequate time), this event means that o_i "is voting" for o_j.

Similarly, the fact that an item o_i is "very similar" in terms of some intrinsic features to o_j can also be interpreted as o_j "recommending" o_i (and viceversa).

Thus, we are able to model a browsing system for the set of items O as a labeled graph (G, l), where: (i) $G = (O, E)$ is a *directed graph*; (ii) $l : E \rightarrow \{pattern, sim\} \times R^+$ is a *labeling function* that associates each edge in $E \subseteq O \times O$ with a pair (t, w), where t is the type of the edge which can assume two enumerative values (*pattern* and *similarity*) and w is the weight of the edge.

We list two different cases:

1. a *pattern label* for an edge (o_j, o_i) denotes the fact that an item o_i was chosen immediately after an item o_j and, in this case, the weight w_j^i is the number of times o_i was chosen immediately after o_j;
2. a *similarity label* for an edge (o_j, o_i) denotes the fact that an item o_i is similar to o_j and, in this case, the weight w_j^i is the "similarity" between the two items. Thus, a link from o_j to o_i indicates that part of the importance of o_j is transferred to o_i.

Given an item $o_i \in O$, its *recommendation grade* $\rho(o_i)$ is defined as follows:

$$\rho(o_i) = \sum_{o_j \in P_G(o_i)} \hat{w}_{ij} \cdot \rho(o_j) \tag{1}$$

where $P_G(o_i) = \{o_j \in O \mid (o_j, o_i) \in E\}$ is the set of predecessors of o_i in G, and $\hat{w}_{ij}$ is the normalized weight of the edge from o_j to o_i.

We note that for each $o_j \in O$ $\sum_{o_i \in S_G(o_j)} \hat{w}_{ij} = 1$ must hold, where $S_G(o_j) = \{o_i \in O | (o_j, o_i) \in E\}$ is the set of successors of o_j in G.

In Albanese et al. (2013), it has been shown that the ranking vector $R = [\rho(o_1) \ldots \rho(o_n)]^T$ of all the items can be computed as the solution to the equation $R = C \cdot R$, where $C = \{\hat{w}_{ij}\}$ is an ad hoc matrix that defines how the importance of each item is transferred to other items.

The matrix can be seen as a linear combination of:

- a *local browsing matrix* $A_h = \{a_{ij}^h\}$ for each user u_h, where its generic element a_{ij}^l is defined as the ratio of the number of times item o_i has been chosen by user u_h immediately after o_j to the number of times any item in O has been chosen by u_h immediately after o_j;
- a *global browsing matrix* $A = \{a_{ij}\}$, where its generic element a_{ij} is defined as the ratio of the number of times item o_i has been chosen by any user immediately after o_j to the number of times any item in O has been chosen immediately after o_j;
- a *similarity matrix* $B = \{b_{ij}\}$ such that b_{ij} denotes the similarity between two items o_i and o_j (e.g., a semantic relatedness based on a given taxonomy using some high-level features values, eventually combined with a low-level features comparison in the case of multimedia data).

The successive step is to compute *customized* rankings for each individual user.

In this case, we can rewrite previous equation considering the ranking for each user as $R_h = C \cdot R_h$, where R_h is the vector of preference grades, customized for a user u_h considering only items in the related O_h^c.

We note that solving the discussed equation corresponds to finding the stationary vector of C, i.e., the eigenvector with eigenvalue equal to 1. In Albanese et al. (2013), it has been demonstrated that C, under certain assumptions and transformations, is a real square matrix having positive elements, with a unique largest real eigenvalue and the corresponding eigenvector has strictly positive components. In such conditions, the equation can be solved using the *Power Method* algorithm.

4.2.3. Refining items ranks using user sentiments and feedbacks

This subsection describes the proposed methodology for the sentiments' extraction from user comments/reviews and its integration in the proposed recommendation strategy.

In particular, the used sentiment extraction technique is an improvement of the approach presented by some of the authors in a previous work (Colace, De Santo, & Greco, 2013), where the

Latent Dirichlet Allocation (LDA) has been adopted for mining the sentiment inside documents.

In our view, the knowledge within a set of documents can be represented in a compact fashion by the use of a complex structure: the *Mixed Graph of Terms* (mGT).

This graph contains the most discriminative words and the probabilistic links between them. More in details, we define a structure made of *weighted word pairs*, which has proven to be effective for sentiment classification problems as well as text categorization and query expansion problems (Colace, De Santo, & Greco, 2014a; Colace, De Santo, Greco, & Napoletano, 2014, 2015). The main reason of such discriminative power is that LDA-based topic modeling is essentially an effective conceptual clustering process and it helps discover semantically rich concepts describing the respective affective relationships. Using these semantically rich concepts, that contain more useful relationship indicators to identify the sentiment from messages, it is possible to accurately discover more latent relationships and make fewer errors in the predictions.

The mGT is built starting from a set of comments belonging to a well-defined knowledge domain and manually labeled according to the sentiment expressed within them.

In this way the mGT contains words (and their probabilistic relationships) which are representative of a certain sentiment for that knowledge domain.

The LDA approach allows to obtain an effective graph by using only few documents. A mGT graph includes two kinds of nodes: the *aggregate roots* nodes, defined as the words whose occurrence is most implied by the occurrence of all other words in the training corpus, and the *aggregate* nodes, defined as the words most related to aggregate roots nodes from a probabilistic point of view.

In Colace et al. (2013) the LDA approach and the mGT formalism have been used for the detection of sentiment in tweets. The approach aims at using the mGT, obtained by LDA based analysis of tweets, as a filter for the classification of the sentiment in a tweet.

Here, the sentiment mining algorithm has been improved by the introduction of new features and its description is reported in the following.

- *Input of the algorithm*: (i) A subset $\widehat{F}^k$ of the feature space F^k corresponding to the set of comments about a given item in a specific knowledge domain. For each comment, additional information about *trustiness* and *ratings* of the commenters has been collected; (ii) the sentiment oriented mixed graphs of terms mGT$^+$ and mGT$^-$ obtained analyzing the (positively and negatively) opinionated items related to a knowledge domain; and (iii) an annotated lexicon L.
- *Output of the algorithm*: (i) The average probabilities P^+ and P^- which express the probability that a sentiment, extracted from the set of comments related to a given item, is "positive" or "negative" (the probabilities P^+ and P^- also take into account the overall rating and trustiness of commenters).
- *Description of the main steps*:
  1. For each word in the mGT$^+$ and the mGT$^-$ their synonymous are retrieved through the annotated lexicon L. In this case, an enriched version of *Wordnet* (Esuli & Sebastiani, 2006; Neviarouskaya, Prendinger, & Ishizuka, 2011) has been selected as lexicon.
  2. For each comment $f_i \in \widehat{F}^k$ if $\frac{T^h_{f_i} - \widetilde{T}_{f_i}}{T_{f_i}} < \tau$ the system discards the comment, $T^h_{f_i}$ being the trustiness of reviewer h, τ a fixed threshold, and $\widetilde{T}_{f_i}$ the average trustiness for all the comments about the selected item.
  3. For each comment the probabilities $P^+_{f_i}$ and $P^-_{f_i}$ are determined as:

     $$\frac{P^{+/-}_{f_i} = (A + B + C + D)}{4}$$

     A being the ratio between the sum of occurrences in the comment of words that are Aggregate Root Nodes and the total number of the Aggregate Root Nodes in the (positive/negative) mGT; B the ratio between the sum of occurrences in the document of words that are Aggregates Nodes and the total number of the Aggregates Nodes in the (positive/negative) mGT; C the ratio between the sum of the co-occurence probabilities of Aggregate Root Nodes pairs that are in the document and the sum of all the co-occurence probabilities of Aggregate Root Nodes pairs in the (positive/negative) mGT; D the ratio between the sum of the co-occurence probabilities of Aggregate Nodes pairs that are in the document and the sum of all the co-occurence probabilities of Aggregate Nodes pairs that are in the (positive/negative) mGT;
  4. For each comment the probabilities $P^+_{f_i}$ and $P^-_{f_i}$ are opportunely weighted by the use of a correction factor which takes into account reviewer's trustiness and rating. In particular, the correction factor is inversely proportional to the difference between the sentiment probability and the normalized rating.
  5. For each item the probabilities P^+ and P^- are determined as:

     $$P^+ = \sum_i \frac{P^+_{f_i}}{num\_of\_comments}$$

     $$P^- = \sum_i \frac{P^-_{f_i}}{num\_of\_comments}$$

Finally, we need a ranking refining function that combines in a proper way the recommendation grade of an item obtained in the previous phase and the values of P^+ and P^- obtained by the described sentiment analysis procedure. The output of this function is for each item $o_i \in O$ the final ranking value $\rho^*(o_i)$, computed as in the following:

$$\rho^*(o_i) = \begin{cases} \rho^{(1-\alpha)} & \text{if } P^+ \geqslant P^- \\ 1 - \rho^{(1-\alpha)} & \text{otherwise} \end{cases} \qquad (2)$$

where $\alpha = |P_+ - P_-|$.

This function increases the recommendation grade value if the sentiment within item's comments is positive, in the opposite decreases it in the case of negative mood.

4.2.4. Post-filtering stage using context information

In this stage, we have introduced a *post-filtering* method for generating the final set of "real" candidates for recommendation using *context* information.

The context is represented by means of the well-known *key-value* model (Adomavicius, Sankaranarayanan, Sen, & Tuzhilin, 2005) using as dimensions some of the different feature spaces related to items. In our system, context features can be expressed either directly using some *target items* (e.g. objects that have positively captured user attention) or specifying the related values in the shape of *constraints* that recommended items have to satisfy.

Assume that a user u_h is currently interested in a target item o_j. We can define the set of candidate recommendations as follows:

$$O^f_{h,j} = \bigcup_{k=1}^{M} \left\{ o_i \in O^c_h \,|\, a^k_{ij} > 0 \right\} \cup \left\{ o_i \in NNQ(o_j, O^c_h) \right\} \qquad (3)$$

The set of candidates includes the items that have been accessed by at least one user within k steps from o_j, with k between 1 and M, and the items that are most similar to o_j according to the results of a *Nearest Neighbor Query* ($NNQ(o_j, O_h^c)$) functionality. Note that a positive element a_{ij}^k of A^k indicates that o_i was accessed exactly k steps after o_j at least once.

The ranked list of recommendations is then generated by ranking the items in $O_{h,j}^f$, for each item o_j selected as interesting by user u_h, using the ranking vector R_h thus obtaining the final set O_h^f.

Finally, for each user all the items that do not respect possible context constraints are removed from the final list.

5. Case studies

We now present two different customizations for our system based on the examples described in Section 2.

5.1. Using the system for recommending movies

We have opportunely customized our system in order to provide recommendation services for users that are interested in coming soon movies.

The design choices are briefly reported in the following.

- We consider as data source the *IMDB* web site, collecting about 10,000 items.
- As items' metadata, we consider for each movie information related on *title*, *genre*, *stars*, *description*, *year*, *director* and *list of theaters* (characterized by name and location) in which they are coming.
- For each movie, available users' preferences, comments and feedbacks have been captured, also exploiting correlated public information from Social Networks (i.e. *Facebook*).
- Users' behaviors has been reconstructed considering the available log time-stamped information of users that have positively rated or watching for a certain time some items in the same browsing session.

For what implementation details concern, the Wrapping modules leverage several *JAVA* technologies with Opinion Mining libraries and exploit the *IMDB API* and *JAXB* libraries to collect the different information of interest.

The Knowledge Base, realized using different technologies, allows to manage all the different kind of information: Contextual Data instances (messages containing information about users' position) are managed by the *Cassandra* DBMS, Items' descriptions are stored in the *Turtle* format and managed by the *Sesame* Repository and *JENA* libraries (semantics of data can be specified by linking values of some attributes to some available ontological schema[1]), user profiles and behaviors are respectively managed by *MongoDB* and *Neo4* DBMSs.

On the other hand, the Recommender Engine exploits proper *JAVA* libraries (developed for the system presented in Albanese et al. (2013) and integrated with co-clustering libraries (Bartolini et al., 2014) and the rank refining procedure) to accomplish its tasks.

5.2. Using the system for recommending travel packages

The system has been also opportunely customized to provide recommendation services for users that are interested in booking travel packages.

The design choices are briefly reported in the following.

- We consider as data source the *tripadvisor* web site, collecting about 30,000 items of different kinds (*hotels*, *fligths*, *restaurants*).
- As items' metadata we consider for each hotel information on *name*, *class*, *price for night*, *services* and *location*; for each restaurant information on *name*, *kind of cooking*, *average prices*, *dining options* and *location*; for each flight information on *price*, *services*, *list of possible stops*, *flight company*, *booking provider*, *Airport of Departure* and *Airport of Arrival*.
- For each item, available users' preferences, comments and feedbacks have been captured, also exploiting correlated public information from Social Networks (i.e., *Facebook*).
- Users' behaviors has been reconstructed considering the available log time-stamped information of users that have positively rated or watching for a certain time some items in the same browsing session.

For what implementation details concern, we use the same technologies of the previous case study except for the data gathering that exploits a combination of the *tripadvisor API* and some wrapping facilities (Canfora, Fasolino, Frattolillo, & Tramontana, 2008) and for the adopted ontologies that leverage some ad hoc and manually created taxonomies for each kinds of items.

6. Experimental results

Recommender Systems are very complex applications that are based on a combination of several models, algorithms and heuristics. This complexity makes evaluation efforts very difficult and thus results are hardly generalizable, as reported in the literature (Adomavicius & Zhang, 2012). Moreover, characterizing and evaluating the quality of a user's experience and subjective attitude toward the acceptance of recommender technology is an important issue which we will consider in the following.

The majority of research efforts on recommender system evaluation have mainly focused on prediction *accuracy* and *stability* (e.g., Adomavicius & Zhang, 2012).

More recently, researchers began examining issues related to users' subjective opinions and developing additional criteria to evaluate recommender systems. In particular, they suggest that user satisfaction does not always (or, at least, not only) correlate with the overall recommender's accuracy.

Starting from these considerations and based on current trends in the literature, we decided to perform both a *user-centric* evaluation and a more traditional evaluation based on well-established accuracy metrics. In particular, the proposed evaluation strategy aims at measuring: (i) *user satisfaction* with respect to assigned browsing tasks for the package travels scenario and (ii) effectiveness of the system in terms of *accuracy* for the movie recommendation problem.

In particular, we evaluated, from one hand, how our recommendations can effectively support browsing tasks of different complexity when the complexity of desired travel packages increases, and from the other hand, how our ranking strategy is accurate for movies recommendation with respect to other recommendation strategies.

6.1. User satisfaction for travel packages recommendation

In a first set of experiments, we evaluated, as in our previous work (Albanese et al., 2011, 2013; Bartolini et al., 2014), the impact of the proposed system on users engaged in several *search tasks* of travel packages and compared its performances with the well-known *tripadvisor* system that, in turn, provides basic search mechanisms as well as sentiment classification facilities.

[1] We use the *Movie Ontology* (http://www.movieontology.org) augmented with names of the most important directors and actors.

In particular, our goal was to establish how helpful our system is in assisting the search of specific travel packages and guiding the users towards information which satisfy their interests. The dataset used in these experiments is a subset of about 5000 items (hotels, restaurants and flights) related to the travel-related content from tripadvisor. In the training phase, a certain amount of browsing sessions have been captured using timestamped data and final browsing matrices have been built.

In order to evaluate the impact of the system on the users, we have conducted the following experiments. We asked a group of about 50 people to browse the collection of travel items and complete several search tasks (20 tasks per user) of different complexity (five tasks for each complexity level), using tripadvisor facilities. After this test, we asked them to browse the same collection with the assistance of our recommender system and complete other 20 tasks of similar complexity. We have subdivided browsing tasks in the following four broad categories:

1. ***Low complexity*** search tasks (T_1): e.g. find at least 15 travel packages – containing a generic hotel, a flight from European airports and a dinner in a generic restaurant – related to some given destination and period.
2. ***Medium complexity*** search tasks (T_2): e.g. find at least 25 travel packages – containing a generic hotel, an economic flight from European airports and a pizza dinner in a generic restaurant – related to some given destination and period.
3. ***High complexity*** search tasks (T_3): e.g. find at least 35 travel packages – containing a four or five stars hotel, an economic flight from European airports and a pizza dinner in a generic restaurant – related to some given destination and period.
4. ***Very high complexity*** search tasks (T_4): e.g. find at least 50 travel packages – containing a four or five stars hotel, an economic flight from an airport near user location and a pizza dinner in a restaurant near the sea- related to some given destination and period.

Note that the complexity of a task depends on several factors: the number of items to explore, the type of desired features and the number of additional constraints. Two strategies were used to evaluate the results of this experiment: (i) empirical measurements of access complexity in terms of *mouse clicks* and *time* and (ii) TLX (*NASA Task Load Index factor*). With respect to the first strategy, we measured the following parameters: (i) *access time* (t_a) – the average time spent by the users to request and access all the packages for a given class of tasks and (ii) *number of clicks* (n_c) – the average number of clicks necessary to collect all the requested packages for a given class of tasks.

Table 1 reports the average values of t_a and n_c for both tripadvisor and our system (User-Centered – UC – Recommender), for each of the four task complexity levels defined. Especially for the most complex tasks, our system shows better performances than tripadvisor.

We then asked the same group of users to express their opinion about the capability of tripadvisor and our system respectively to provide an effective user experience in completing the assigned search tasks, based on the TLX evaluation protocol (Hart & Staveland, 1988). Specifically, TLX is a multi-dimensional rating procedure that provides an overall workload score based on a weighted average of ratings on six sub-scales: mental demand, physical demand, temporal demand, own performance, effort and frustration. Lower TLX scores are better. The average scores are reported in Table 2.

Our system outperforms in a significative way tripadvisor in every sub-scale except for *mental demand* and *performance*: this happens because sometimes an expert user considers the automatic suggestions not useful, just because they know what they are looking for. In summary, our system provides a better (less frustrating) user experience during the search tasks. In addition, the fact that search tasks can be completed faster using our system is an indication that recommendations are effective, as they allow a user to explore interesting and related items one after another, without the interference of undesired items that would otherwise slow down the process.

Table 1
Comparison between our system and tripadvisor in terms of t_a and n_c average values.

Task class	System	t_a (s)	n_c
Low complexity	UC recommender	157	33
Low complexity	Tripadvisor	166	35
Medium complexity	UC recommender	245	70
Medium complexity	Tripadvisor	354	91
High complexity	UC recommender	416	121
High complexity	Tripadvisor	502	162
Very high complexity	UC recommender	645	248
Very high complexity	Tripadvisor	842	300

Table 2
Comparison between our system and Picasa in terms of TLX factors for each category of users.

TLX factor	UC recommender	Tripadvisor
Mental demand	32	39
Physical demand	34.3	50
Temporal demand	33	38
Effort	32	55.2
Performances	63.4	79.8
Frustation	23.4	38.5

6.2. Accuracy for movies recommendation

For this kind of experiments, we used the dataset provided by the *GroupLens*[2] website, which makes available data collected by the *MovieLens*[3] recommender system. Through its website, MovieLens collects the preferences expressed by a community of registered users on a huge set of movie titles.

The adopted dataset contains (i) explicit ratings about 1682 movies made by 943 users (only users who have rated at least 20 movies are considered), (ii) demographic information about users (age, gender, occupation, zip code), and (iii) a brief description of the movies (title, release year, genres). The dataset is characterized by a very high density.

The experiments have been conducted on a collection of about 1000 movies (for which a consistent number of comments are available), rated by a subset of 100 users: each of them had rated at least 150 movies and at most 300, assigning each movie a score between 1 (*"Awful"*) and 5 (*"Must see"*). Additionally, using the *timestamp* information, we were able to reconstruct usage patterns for each user and consequently the browsing matrices.

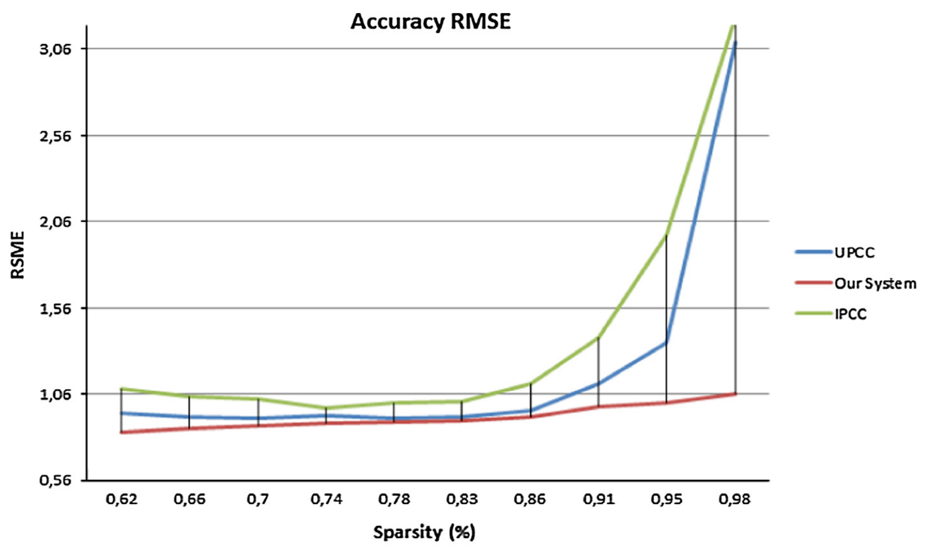
We used the *Mean Absolute Error* (MAE) and the *Root Mean Square Error* (RMSE) as metrics in our experiments. In our case, MAE and RMSE are defined as:

$$\mathrm{MAE} = \frac{1}{N}\sum_{u,i,j}|r^j_{ui} - -\hat{r}^j_{ui}|$$

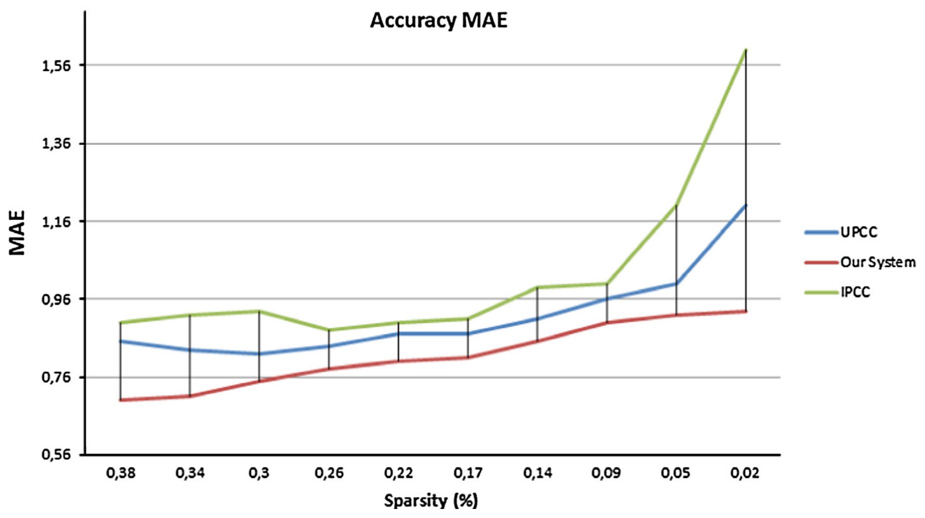
$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum\nolimits_{u,i,j}(r^j_{ui} - -\hat{r}^j_{ui})^2}$$

where r^j_{ui} is the actual rating that the user u has given to item i for the item j, $\hat{r}^j_{ui}$ is the system predicted rating (the recommendation grades were also normalized on a scale from 1 to 5), and N is the

[2] http://www.grouplens.org.
[3] http://movielens.umn.edu.



(a) RMSE comparison



(b) MAE comparison

Fig. 2. Comparison in terms of MAE and RMSE between our approach and UPCC and IPCC respectively.

total number of test ratings. Both MAE and RMSE thus attempt to measure the prediction error (accuracy of the recommendation): RMSE is considered as a stronger measure than MAE as larger prediction errors are penalized more. For both the metrics, smaller values indicate better performances.

We compared the accuracy in terms of *Root Mean Square Error* of the predictions computed by our recommender system with the UPCC and IPCC (Su & Khoshgoftaar, 2009) approaches (which reliable implementation can be obtained leveraging machine learning libraries provided by the *Apache Mahout* framework). In particular, we selected 50 test users (with a low value of trustiness) and computed the average accuracy for 50 predictions on a subset of the most recently observed items, increasing data sparsity for the same users. In the case of our system, we used the last observed item in the test users' browsing sessions as the query item for post-filtering stage, and normalized the recommendation grade on a scale ranging from 1 to 5.

Fig. 2 shows the trend of RMSE for our system as well as for the UPCC and IPCC algorithms, as the sparsity of the rating matrix increases. Our approach outperforms UPPC and IPPC ones for each value of items' sparsity – and especially for higher values – showing as social information can improve recommendations.

However, our approach is more accurate when the sparsity of the rating matrix is high. This is due to the use of the *similarity matrix*, which provides useful information to the algorithm, in order to compute meaningful predictions even if a user's browsing session data is not available. Thus, our approach does not suffer from the *cold start problem*.

6.3. Considerations on efficiency

In order to evaluate the efficiency of our recommender system, we have measured execution times w.r.t. the execution times of other state-of-the-art methods. Because the recommendation grades computation can be performed in off-line manner and the related updates are correlated to the insert of a new item (or an update of its features) or new user, the running time are essentially dependent on the size of candidate items' set obtained in the pre-filtering stage.

In general, we observed that the average computation times for all methods are comparable and it takes at most few seconds to obtain useful recommendations also for large sets of candidates.

Fig. 3 shows the trend of average ranking computation times for our approach as the size of the candidate set increases.

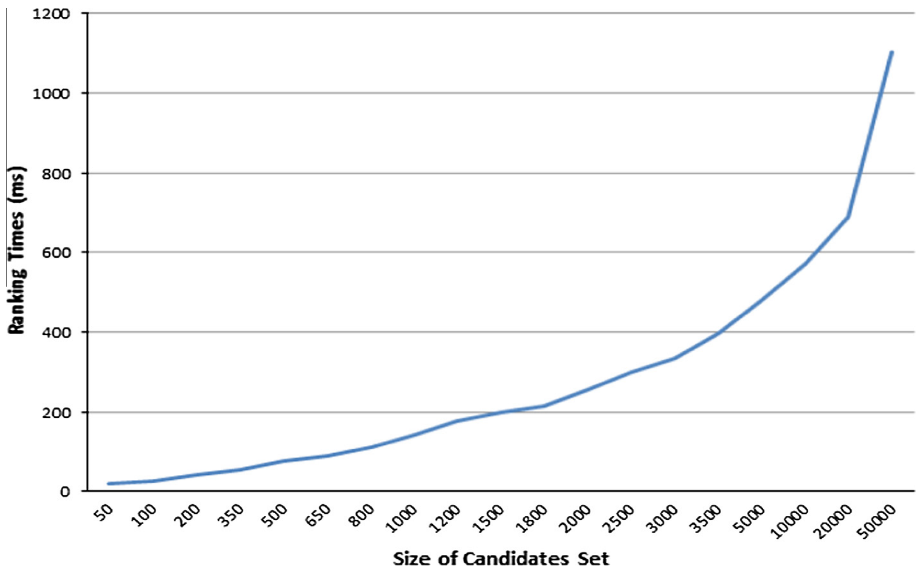


Fig. 3. Computation times for ranking vs. size of the candidate set.

7. Conclusions and future work

In this paper a novel collaborative and user-centered recommendation approach for online social networks was proposed.

Several aspects related to users – i.e. *preferences* (usually in the shape of items' metadata), *opinions* (textual comments to which it is possible to associate a sentiment), *behavior* (in the majority of cases logs of past items' observations made by users), *feedbacks* (usually expressed in the form of ratings) – were considered and integrated together with *items' features* and *context information* within a general framework that can support different applications using proper customizations.

Social elements and item features were considered and embedded in our strategy to improve effectiveness of recommendations, overcoming limits of collaborative learning approaches due to the availability and quality of user profiles and ratings.

In particular, modern social networking applications can take the more important advantages of our framework that can also be adopted as the recommendation engine for a variety of items (news to read, movie to watch, music to listen, product to buy, friends to invite, travel to do, etc.) for existing applications in social networks (e.g. the *Movies App* on Facebook).

We focused on two case studies and implemented two recommender systems based on our innovative approach.

The first system helps users to choose coming soon movies having IMDB as main data source, whereas the second system helps users to find travel packages with certain characteristics having tripadvisor as main repository. Then we investigated the effectiveness of the proposed approach in the considered scenarios, through the evaluation of *accuracy* and *user satisfaction*.

As shown through the two case studies, the proposed approach can be easily adapted to several kinds of applications. The primary difference, when applying our approach to diverse domains, is in the choice of items' featues (for multimedia data also low level features should be considered) and in the metric used to assess the similarity between any two data items with respect to such descriptors.

Therefore, the specific nature of the items in the repository only affects the computation of the *B* matrix.

In addition, the system can be easily used to provide recommendations of more than one category of items. In this case, mechanisms for grouping different items in a unique object are necessary.

Experimental results have shown that our approach is promising and encourage further research in this direction. The current major limitation of this work is represented by the relatively small size of the data sets used for the experiments, both in terms of number of items and in terms of number of users involved in the evaluation.

Testing our approach on larger volumes of data will certainly strengthen our work: however we note that enrolling a large population of users for the type of experiments described in this paper is not trivial and extremely time consuming. Nonetheless, we are working on expanding our experimental setup, and expect to present additional results in the near future.

Summing up, future efforts will be devoted to (i) extending the experimental evaluation to a larger image data set, also considering the *stability* of recommendations and (ii) applying our approach to other kinds of data from heterogeneous collections and compare it with other more recent approaches of the literature.

References

Adomavicius, G., Sankaranarayanan, R., Sen, S., & Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems (TOIS), 23*(1), 103–145.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17*, 734–749.

Adomavicius, G., & Zhang, J. (2012). Stability of recommendation algorithms. *ACM Transactions on Information Systems (TOIS), 30*(4), 23.

Albanese, M., d'Acierno, A., Moscato, V., Persia, F., & Picariello, A. (2011). A multimedia semantic recommender system for cultural heritage applications. In *2011 Fifth IEEE international conference on semantic computing (ICSC)* (pp. 403–410). IEEE.

Albanese, M., d'Acierno, A., Moscato, V., Persia, F., & Picariello, A. (2013). A multimedia recommender system. *ACM Transactions on Internet Technology, 13*(1), 3:1–3:32.

Bartolini, I., Moscato, V., Pensa, R. G., Penta, A., Picariello, A., Sansone, C., et al. (2014). Recommending multimedia visiting paths in cultural heritage applications. *Multimedia Tools and Applications*, 1–30.

Canfora, G., Fasolino, A. R., Frattolillo, G., & Tramontana, P. (2008). A wrapping approach for migrating legacy system interactive functionalities to service oriented architectures. *Journal of Systems and Software, 81*(4), 463–480.

Colace, F., De Santo, M., & Greco, L. (2014a). An adaptive product configurator based on slow intelligence approach. *International Journal of Metadata, Semantics and Ontologies, 9*(2), 128–137.

Colace, F., De Santo, M., & Greco, L. (2014b). E-learning and personalized learning path: A proposal based on the adaptive educational hypermedia system. *International Journal of Emerging Technologies in Learning (iJET), 9*(2), 9.

Colace, F., De Santo, M., Greco, L., & Napoletano, P. (2014). Text classification using a few labeled examples. *Computers in Human Behavior, 30*, 689–697.

Colace, F., De Santo, M., Greco, L., & Napoletano, P. (2015). Weighted word pairs for query expansion. *Information Processing & Management, 51*(1), 179–193.

Colace, F., De Santo, M., & Greco, L. (2013). A probabilistic approach to tweets' sentiment classification. In *Affective computing and intelligent interaction (ACII), 2013 humaine association conference on* (pp. 37–42).

Ding, X., Liu, B., & Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining, WSDM '08* (pp. 231–240). New York, NY, USA: ACM.

Dong, R., O'Mahony, M. P., Schaal, M., McCarthy, K., & Smyth, B. (2013). Sentimental product recommendation. In *Proceedings of the 7th ACM conference on recommender systems, RecSys '13* (pp. 411–414). New York, NY, USA: ACM.

Dourish, P. (2004). What we talk about when we talk about context. *Personal and Ubiquitous Computing, 8*(1), 19–30.

Drigas, A. S., Ioannidou, R.-E., Kokkalia, G., & Lytras, M. (2014). ICTs, mobile learning and social media to enhance learning for attention difficulties. *Journal of Universal Computer Science, 20*(10), 1499–1510.

Esuli, A., & Sebastiani, F. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th conference on language resources and evaluation (LREC 2006)* (pp. 417–422).

Ganu, G., Kakodkar, Y., & Marian, A. (2013). Improving the quality of predictions using textual information in online user reviews. *Information Systems, 38*(1), 1–15.

Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in Psychology, 52*, 139–183.

Hijikata, Y., Iwahama, K., & Nishida, S. (2006). Content-based music filtering system with editable user profile. In *Proceedings of the 2006 ACM symposium on applied computing, SAC '06* (pp. 1050–1057). New York, NY, USA: ACM.

Ienco, D., Robardet, C., Pensa, R. G., & Meo, R. (2013). Parameter-less co-clustering for star-structured heterogeneous data. *Data Mining and Knowledge Discovery, 26*(2), 217–254.

Kabassi, K. (2013). Personalisation systems for cultural tourism. In *Multimedia services in intelligent environments* (pp. 101–111). Springer.

Karatzoglou, A., Amatriain, X., Baltrunas, L., & Oliver, N. (2010). Multiverse recommendation: *n*-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on recommender systems* (pp. 79–86). ACM.

Koren, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '08* (pp. 426–434). New York, NY, USA: ACM.

Kurilovas, E., Juskeviciene, A., Kubilinskiene, S., & Serikoviene, S. (2014). Several Semantic web approaches to improving the adaptation quality of virtual learning environments. *Journal of Universal Computer Science, 20*(10), 1418–1432.

Leung, C. W. K., Chan, S. C. F., & Chung, F.-l. (2006). Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *Proceedings of The ECAI 2006 workshop on recommender systems* (pp. 62–66). Citeseer.

Maidel, V., Shoval, P., Shapira, B., & Taieb-Maimon, M. (2008). Evaluation of an ontology-content based filtering method for a personalized newspaper. In *Proceedings of the 2008 ACM conference on recommender systems, RecSys '08* (pp. 91–98). New York, NY, USA: ACM.

Musial, K., Juszczyszyn, K., & Kazienko, P. (2008). Ontology-based recommendation in multimedia sharing systems. *System Science, 34*, 97–106.

Neviarouskaya, A., Prendinger, H., & Ishizuka, M. (2011). Sentiful: A lexicon for sentiment analysis. *IEEE Transactions on Affective Computing, 2*(1), 22–36.

Pappas, N., & Popescu-Belis, A. (2013). Sentiment analysis of user comments for one-class collaborative filtering over ted talks. In *Proceedings of the 36th international ACM SIGIR conference on research and development in information retrieval, SIGIR '13* (pp. 773–776). New York, NY, USA: ACM.

Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web lecture notes in computer science* (Vol. 4321, pp. 325–341).

Ramakrishnan, N., Keller, B. J., Mirza, B. J., Grama, A. Y., & Karypis, G. (2001). Privacy risks in recommender systems. *IEEE Internet Computing, 5*, 54–62.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). *Grouplens: An open architecture for collaborative filtering of netnews*. ACM Press.

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (Eds.). (2011). *Recommender systems handbook*. Springer.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web, WWW '01* (pp. 285–295). New York, NY, USA: ACM.

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '02* (pp. 253–260). New York, NY, USA: ACM.

Singh, V. K., Mukherjee, M., & Mehta, G. K. (2011). Combining collaborative filtering and sentiment classification for improved movie recommendations. In C. Sombattheera, A. Agarwal, S. K. Udgata, & K. Lavangnananda (Eds.), *MIWAI. Lecture notes in computer science* (Vol. 7080, pp. 38–50). Springer.

Sinha, R. R., & Swearingen, K. (2001). Comparing recommendations made by online systems and friends. In *DELOS workshop: Personalisation and recommender systems in digital libraries*.

Stan, J., Muhlenbach, F., Largeron, C., et al. (2014). Recommender systems using social network analysis: Challenges and future trends. *Encyclopedia of Social Network Analysis and Mining*, 1–22.

Su, X., & Khoshgoftaar, T. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*.

Yang, S.-H., Long, B., Smola, A. J., Zha, H., & Zheng, Z. (2011). Collaborative competitive filtering: Learning recommender using context of user choice. In *Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval* (pp. 295–304). ACM.

Yildirim, H., & Krishnamoorthy, M. S. (2008). A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the 2008 ACM conference on recommender systems, RecSys '08* (pp. 131–138). New York, NY, USA: ACM.

Yu, H.-F., Hsieh, C.-J., Si, S., & Dhillon, I. S. (2013). Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, 1–27.

Zhou, X., Xu, Y., Li, Y., Josang, A., & Cox, C. (2012). The state-of-the-art in personalized recommender systems for social networking. *Artificial Intelligence Review, 37*(2), 119–132.

Zhuang, L., Jing, F., & Zhu, X.-Y. (2006). Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on information and knowledge management, CIKM '06* (pp. 43–50). New York, NY, USA: ACM.