Chapter 2
Knowledge-Based Leisure Time Recommendations in Social Networks

Dionisis Margaris, Costas Vassilakis and Panagiotis Georgiadis

Abstract We introduce a novel knowledge-based recommendation algorithm for leisure time information to be used in social networks, which enhances the state-of-the-art in this algorithm category by taking into account (a) qualitative aspects of the recommended places (restaurants, museums, tourist attractions etc.), such as price, service and atmosphere, (b) influencing factors between social network users, (c) the semantic and geographical distance between locations and (d) the semantic categorization of the places to be recommended. The combination of these features leads to more accurate and better user-targeted leisure time recommendations.

Keywords Knowledge-based recommender systems · Social networks · Collaborative filtering · Attribute constraints · Semantic information

2.1 Introduction

Knowledge-based recommender systems are a special type of recommender systems (RS) that use knowledge about users and products to pursue a knowledge-based approach to generating a recommendation, reasoning about what products meet the users' requirements [1]. Knowledge-based RS exploit semantic information to improve similarity matching between items or user profiles and items [2].

D. Margaris (✉) · P. Georgiadis
Department of Informatics and Telecommunications, University of Athens, Athens, Greece
e-mail: margaris@di.uoa.gr

P. Georgiadis
e-mail: p.georgiadis@di.uoa.gr

C. Vassilakis
Department of Informatics and Telecommunications, University of the Peloponnese, Tripoli, Greece
e-mail: costas@uop.gr

G. Alor-Hernández and R. Valencia-García (eds.), *Current Trends on Knowledge-Based Systems*, Intelligent Systems Reference Library 120,
DOI 10.1007/978-3-319-51905-0_2

Knowledge-based RS may employ (a) constraint-based interaction, where the user specifies constraints on the items she requests, (b) case-based interaction, where the user typically specifies specific targets and the system returns similar results and (c) may include collaborative information by identifying other users with similar profiles, and using their session information in the learning process [3].

When collaborative filtering (CF) is employed, typical RS assume that users are independent and ignore social interactions among them. Consequently, they fail to incorporate important aspects denoting interaction, such as tie strength and influence among users, which can substantially enhance recommendation quality [4, 5]. RS based on social network (SN) data tackle this issue by considering data from the user profile (e.g. location, age or gender) complemented with dynamic aspects stemming from user behavior and/or the SN state, such as user preferences, items' general acceptance and influence from social friends [4, 5]. Furthermore, tie strength between users of the SN can be exploited to further enhance the choice of recommenders, so as to consider the opinions and choices of users that have a high influence on the user for whom the recommendation is generated [6–8].

As far as SN influence is concerned, a first approach to identifying highly influential individuals within the SN is to consider those having high tie strengths, such as family members or friends with similar age; however, the influence of such individuals may be limited only to certain place categories: for instance, one may trust her friends regarding restaurants and pastry shops, but not regarding bars. Moreover, selected individuals with low tie strength, such as actors and singers, may influence a user regarding some specific categories (e.g. shops), while for some categories a user may not be influenced at all (e.g. a user may consider herself an expert in museums, hence she decides exclusively on her own, after examining the types of museums such as folk or fossil).

Recently, it has been shown that RS should consider qualitative aspects of items (typically referred to as QoS—quality of service), such as price, security, reliability, etc., the individual user behavior regarding her purchases in different item categories, and the semantic categorization of items [9, 10]. For example, if a user typically buys a glass of wine in the price range $10–$15, it would be inappropriate to recommend a restaurant charging $100 for a single glass of wine, on the grounds that some user having a high influence on the restaurant category has made a check-in in that particular restaurant. Instead, it would be more appropriate to recommend a low-cost restaurant *of the same style* with the one of the $100 glass of wine, which best fits the profile of the user to whom the recommendation is addressed.

In this chapter, we propose a novel algorithm for making accurate knowledge-based leisure time recommendations to social media users. The proposed algorithm considers qualitative attributes of the places (e.g. price, service, atmosphere), the profile and habits of the user for whom the recommendation is generated, place similarity, the physical distance of locations within which places are located, and the opinions of the user's influencers. The proposed algorithm is the first algorithm that combines the above listed features into a single and effective recommendation process, and this combination leads to increased accuracy. The proposed algorithm is evaluated both in terms of recommendation accuracy and execution performance.

In the rest of this chapter, Sect. 2.2 overviews related work, while Sect. 2.3 presents the proposed algorithm's prerequisites. Section 2.4 describes the knowledge-based recommendation algorithm for SN, while Sect. 2.5 evaluates the proposed algorithm. Finally, Sect. 2.6 concludes the chapter and outlines future work.

2.2 Related Work

Bakshy et al. [11] examine the role of SN in the RS within a field experiment that randomizes exposure to signals about friends' information and the relative role of strong and weak ties. In [12], Bakshy et al. measure social influence via social cues, demonstrate the consequences of including minimal social cues in advertising and measure the positive relationship between a consumer's response and the strength of her connection with an affiliated peer. Both these works establish that recommendation algorithms are valuable tools in SN. Oechslein et al. [7] also assert that a strong tie relationship positively influences the value of a recommendation.

In the domain of RS, numerous approaches for formulating recommendations have been proposed. Collaborative filtering (CF) formulates personalized recommendations on the basis of ratings expressed by people having similar tastes to the user for which the recommendation is generated. Taste similarity is computed by examining the resemblance of already entered ratings [13]. The CF-based recommendation approach is the most successful and widely used approach for implementing RS [14]. CF can be further distinguished in user-based and item-based approaches [15]. In user-based CF, a set of nearest neighbours of the target user is first identified, and the prediction value of items that are unknown to the target user is then computed according to this set. On the other hand, item-based CF proceeds by finding a set of similar items that are rated by different users in some similar way. Subsequently, predictions are generated for each candidate item, for example, by taking a weighted average of the active user's item ratings on these neighbour items. Item-based CF achieves prediction accuracies that are comparable to, or even better than, user-based CF algorithms [16]. To improve recommendation accuracy, knowledge-based recommender systems nowadays employ cutting-edge techniques such as data mining and segmentation [17]. Recommender systems apply to many item domains; RESYGEN [18] is a recommendation system generator that can generate multi-domain systems. For computing similarities in the recommendation process, RESYGEN provides a similarity metrics library and the RS configurator chooses the most appropriate one.

Recently, SN recommendation has received considerable research attention. Konstas et al. [19] investigate the role of SN relationships in developing a track recommendation system using CF and taking into account both the social annotations and friendships inherent in the social graph established among users, items and tags. Arazy et al. [6] outline a conceptual RS design within which the structure and dynamics of a SN contribute to the dimensions of trust propagation, source's reputation and tie strength between users, which are then taken into account to generate recommendations. Quijano-Sanchez et al. [8] enhance a content-based RS by considering the trust between individuals, users' interaction and aspects of each user's personality. In [20], a matrix factorization-based approach for recommendation in SN is explored, employing a mechanism of trust propagation. He et al. [5] analyze data from a SN and establish that friends have a tendency to select the same items and give similar ratings; they also show that using SN data within the RS improves prediction accuracy but also remedies the data sparsity and cold-start issues inherent in CF.

As far as leisure time place recommendation is concerned, Zheng et al. [21] make personalized travel recommendations from user GPS traces, by modeling multiple users' location histories and mining the top n interesting locations and the top m classical travel sequences in a region. Bao et al. [22] present a location-based and preference-aware RS, which provides a user with location recommendations around the specified geo-position based on (a) the user's personal preferences learnt from her location history and (b) social opinions mined from the *local experts* who could share similar interests. RecomMetz [23] is a context-aware knowledge-based mobile RS specialized in the domain of movie showtimes based on location, time and crowd information. RecomMetz views recommended items as composite ones, with salient aspects being the theatre, the movie and the showtime. iTravel [24] is an attraction recommendation, employing mobile peer-to-peer communications for exchanging ratings via users' mobile devices, and using these ratings for recommendation formulation. Moreno et al. [25] present SigTur/E-Destination, a RS for tourist and leisure activities situated in the region of Tarragona, Spain. SigTur/E-Destination exploits semantic information attached to leisure activities and opinions from similar users to generate recommendations. Ference et al. [26] study the issues in making location recommendations for out-of-town users, by taking into account user preference, social influence and geographical proximity and introduce UPS-CF, a recommender engine for making location recommendation for mobile users in location-based SN such as FourSquare. Table 2.1 depicts a feature comparison between the presented algorithm and other leisure time place recommenders surveyed in this section.

In Table 2.1, we can see that the proposed algorithm is the only one supporting QoS aspects, while it additionally supports all other dimensions (QoS, SN, semantic matching and proximity). Furthermore, the proposed algorithm computes influence among user pairs in SN per interest category (as contrasted to computations only at user level) and uses semantic distances between (a) places and (b) the locations within which the places are located. These features, unique to the proposed algorithm, enable the formulation of highly accurate recommendations.

Table 2.1 Leisure time place recommenders feature comparison

Algorithm-reference	Recommendation domain	QoS aspects?	SN-aware?	Semantics?	Proximity?
Zheng et al. [21]	Locations, travel sequences	No	No	No	Yes
Bao et al. [22]	Nearby locations	No	Yes	No	Yes
RecomMetz [23]	Movie showtimes	No	No	Yes	Yes
iTravel [24]	Attractions	No	Yes	No	Yes
SigTur/E-Destination [25]	Leisure activities in Taragona region	No	No	Activities	Yes
UPS-CF [26]	Locations	No	Yes	No	Yes
Proposed algorithm	Places	Yes	Yes	Yes	Yes

2.3 Social Networking, Semantics and QoS Foundations

In this section, we summarize the concepts and foundations from the areas of SN, semantic data management and quality of service (QoS), which are used in this work. QoS, in particular, relates to important attributes on which constraints are imposed within the knowledge-based recommendation process, hence it plays a central role in the presented knowledge-based recommendation algorithm.

2.3.1 *Influence in Social Networks*

Within a SN, "social friends" greatly vary regarding the nature of the relationship holding among them: they may be friends or strangers, with little or nothing in between [27]. Users have friends they consider very close, and know each other in real life and acquaintances they barely know, such as singers, actors and athletes [28]. Bakshy et al. [12] suggest that a SN user responds significantly better to recommendations (e.g. advertisements) that originate from friends of the SN to which the user has a high *tie strength*. In their work, the strength of the directed tie between users i and j is linked to the amount of communication that has taken place between the users in the recent past and is computed as:

$$W_{i,j} = \frac{C_{i,j}}{C_i} \tag{2.1}$$

where C_i is the total number of communications posted by user i in a certain time period (a period of 90 days is considered for computing the tie strength) in the SN, whereas $C_{i,j}$ is the total number of communications posted on the SN by user i during the same period and are directed towards user j or on posts by user j.

Although the tie strength metric can be used to locate the influencers of a user, it does not consider user interests, which are important in RS. In our work, we consider a more elaborate influence metric, which computes the tie strength between users *i* and *j* for each distinct interest. In more detail, the influence level $IL_{i,j}(C)$, where *C* is an interest category is defined as follows:

$$IL_{i,j}(C) = \begin{cases} W_{i,j}, & \mathit{if}\ C \in \mathit{interests}(i) \wedge C \in \mathit{interests}(j) \\ 0, & \mathit{otherwise} \end{cases} \tag{2.2}$$

Effectively, this formula assigns a zero influence level value for interests that are not shared among the considered users, whereas for common interests, the value of the tie strength is used. For the population of each user's interest set, we use the user interest lists collected by the SN [29]. Since this list is built automatically when the user interacts with the SN, it will be comprehensive and will include all categories that the user is interested in.

2.3.2 Leisure Time Places Semantic Information and Similarity

To generate successful recommendations, the algorithm must be able to find which leisure places are similar. This is achieved through recording semantic information related to the places and using this information to compute semantic similarity among them. Semantic information is stored in ontologies and Fig 2.1. illustrates an ontology concerning places: the "Building" entity is the root of the *is-a* hierarchy, and it is subclassed to generate more specific place categories such as attraction, accommodation, leisure. Each class/subclass is described through a set of properties which apply to all instances of the particular class as well as to instances of its subclasses (e.g. an attraction may have a property expressing age suitability, which is inherited to all its subclasses, namely religious monuments and museums).

In this work, we adopt a modified version of the similarity measure proposed by [30]: the semantic similarity (*SemSim*) between two places p_i and p_j is based on the ratio of the shared Resource Description Frameworks (RDF) descriptions between p_i and p_j (*count_common_desc*(p_j, p_j)) to their total descriptions (*count_total_desc* (p_i, p_j)), i.e.:

$$SemSim(p_i, p_j) = \frac{count\_common\_desc(p_i, p_j)}{count\_total\_desc(p_i, p_j)} \tag{2.3}$$

However, the RDF descriptions of two places may not be identical, yet be semantically close: e.g., a restaurant with Lebanese cuisine can be considered of high similarity to a restaurant with Tunisian cuisine, but of low similarity with a Japanese restaurant. To address this aspect, we modify the similarity metric formula to

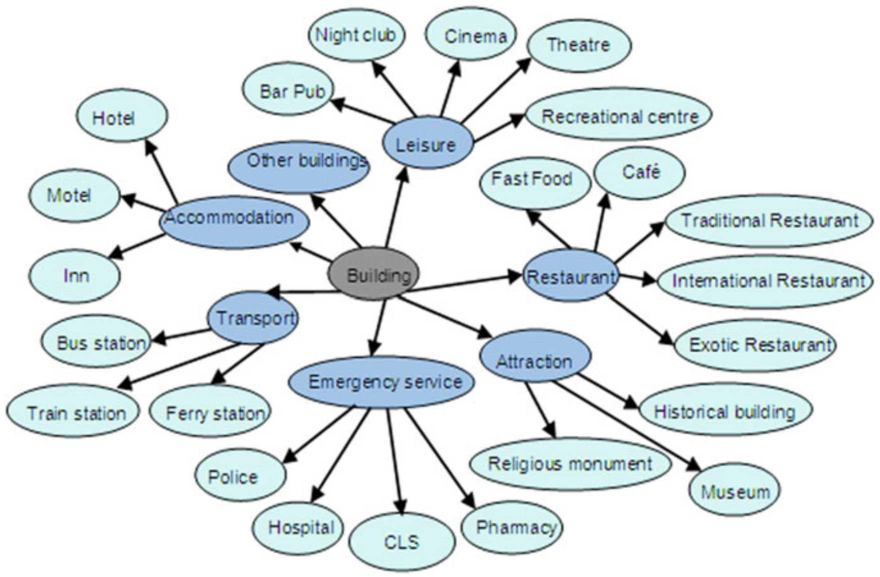

Fig. 2.1 Example of leisure time ontology tree

$$SemSim(p_i, p_j) = \frac{\sum_{p \in p_i \wedge p \in p_j} sim_p\left(V_p(p_i), V_p(p_j)\right)}{count\_total\_desc(p_i, p_j)} \tag{2.4}$$

where p is a property, $V_p(p_i)$ and $V_p(p_j)$ are the values of property p for items p_i and p_j, respectively, and sim_p is a function computing the similarity between values of property p. In the example given above, we may consider that $sim_{cuisine}(Lebanese, Tunisian)$ = 0.9 (i.e. a high value) and $sim_{cuisine}(Lebanese, Japanese)$ = 0.1 (i.e. a low value). For many properties with a numeric (integer or real) domain (e.g. prices, distances, hotel star ratings etc.) the $\mathrm{sim_p}$ function may be defined as

$$sim_{num\_prop}(\mathrm{v1}, \mathrm{v2}) = 1 - \frac{|\mathrm{v1} - \mathrm{v2}|}{\max(num\_prop) - \min(num\_prop)} \tag{2.5}$$

where $max(num\_prop)$ and $min(num\_prop)$ are the maximum and minimum values respectively of $numeric\_prop$ in the ontology extension; this is a typical value normalization formula [31]. Undoubtedly, defining a similarity function for each property within ontology is a laborious task. To mitigate this issue, automated similarity computation methods for specific domains can be used. For instance, metrics sim_g and sim_d [32] can be used for movie genres and movie directors; the metrics in [33] can be used for colors; and so forth. To further decrease the amount of similarity functions that need to be defined, place similarity computation may consider only the places' salient features (e.g. for a museum, the ticket cost and the type of its collections (such as gallery, antiquities, modern art) are salient features, but its number of floors is not). In the absence of any algorithmic or custom-provided similarity metric, the default metric

$$sim_{default}(v1, v2) = \begin{cases} 1, & \textit{if } v1 = v2 \\ 0, & \textit{otherwise} \end{cases} \tag{2.6}$$

can be used, offering performance identical to the one of the method used in [30].

Note that the *SemSim* metric is able to appropriately handle the comparison of places with overlapping features: for instance, when comparing a museum *M* which includes a gift shop to a gift shop *G*, the properties related to the gift shop features will be present in both *M* and *G* and will be appropriately compared. The non-common properties (properties related to the museum's nature, e.g. *Collection types*) will increase the value of the *count_total_desc*(*M*, *G*) quantity, leading to the computation of a lower similarity value, as would be expected.

2.3.3 Physical Distance-Based and Thematic-Based Location Similarity

Besides the semantic information considering places, the physical distance and the thematic similarity of the locations within which the places are located have been proved to play an important role when humans search for places that include a geographical dimension [34]. According to Jones et al. [34], the following criteria can be used to assess the physical distance-based similarity: (a) distance in map or geographical coordinate space between locations, (b) travel time between locations, (c) number of intervening places, (d) spatial inclusion of one location in the other, (e) overlapping between locations, and (f) boundary connectivity between locations.

For physical distance-based similarity, Jones et al. [34] compute a combined spatial closeness measure, called *Total Spatial Distance* (*TSD*) using the formula:

$$TSD(loc_1, loc_2) = w_e * ED(loc_1, loc_2) + w_h * HD(loc_1, loc_2) \tag{2.7}$$

where *ED* is the (normalized) Euclidian distance between the two locations and *HD* is the respective hierarchical distance (again, normalized), which is computed using an hierarchy of part-of relations (e.g. Paris is *part-of* Île-de-France, which is *part-of* France etc.). w_e and w_h are weights assigned to *ED* and *HD* respectively; in Jones et al. [34], w_e is set to 0.6 and w_h to 0.4. Metric value normalization is performed using a formula analogous to the sim_{num_prop} formula (c.f. Sect. 2.3.2).

Regarding the thematic-based location similarity, Jones et al. [34] introduce a *Thematic Distance* (*TD*) metric, which takes into account the semantic similarity of classification terms attached to each location; these terms are drawn from the Art and Architecture Thesaurus taxonomy. Locations having attached classification terms that are semantically close have small thematic distance, while locations having attached semantically dissimilar classification terms have a high thematic distance.

Finally, in order to combine *TSD* and *TD* into a single score, Jones et al. [34] employ the weighted average formula shown in Eq. (2.8), setting the weight w_t of *TD* to 0.4 and the weight w_s of *TSD* to 0.6. In our work, we have adopted the approach of [34], modified to allow the use of arbitrary classification terms instead of terms drawn from a specific taxonomy; to compute the semantic distance between classification terms attached to places, we used the *word2vec* library [35].

$$LocationSim(loc_1, loc_2) = 1 - (w_t * TD(loc_1, loc_2) + w_s * TSD(loc_1, loc_2)) \quad (2.8)$$

2.3.4 Leisure Time Places QoS Information

QoS may be defined in terms of attributes [36]. Attributes typically considered in the context of QoS are cost, timeliness, reliability, courtesy, etc. [37]. In this chapter we will consider only the attributes cost (c), service (s) and atmosphere (a), as used in many of the travel websites, such as Tripadvisor or Opentable. When choosing a place to visit, users typically try to minimize cost and maximize service and atmosphere. It is straightforward to extend the algorithm presented below for handling QoS information to include more attributes, hence the consideration of only three attributes does not lead to loss of generality. Regarding the cost, actual prices are used, while values for the service and atmosphere scores are taken from sites such as TripAdvisor (e.g. http://www.tripadvisor.com/Restaurants-g186338-London_England.html refers to London restaurants). These values are normalized in the scale of 1–10, with larger values denoting higher quality. An example of the London's restaurants qualitative characteristics values are shown in Table 2.2.

2.3.5 User's Profile for Enabling Recommendations

As discussed in the introduction, users are influenced regarding leisure time places they visit by other users; the set of influencers may vary between place categories, e.g. user u may trust her friends f_1 and f_2 regarding restaurants, but regarding bars

Table 2.2 Sample QoS values within the repository

Place	Cost $	Service	Atmosphere
Restaurant Gordon Ramsay	140	10	9
Italian Pizza Connection	350	9	8
London Fish & Chips	8	8	7
…			

she may trust her friend f_3 and be influenced by the choices made by actor A. Moreover, a user u may make decisions on her own for a particular place category, by personally locating candidate places and examining their characteristics. In order to accommodate these aspects in the RS, we follow the approach presented in [38], adapting it appropriately. According to this approach, in order to formulate a leisure time place recommendation, two subtasks are executed in parallel: the first task computes a QoS-based recommendation, while the second task computes a CF-based recommendation. Then, the two recommendations are combined to formulate the final recommendation, employing a metasearch algorithm [31].

In our case, the CF-based algorithm considers the opinions of the user's influencers for the particular leisure time place category. A distinct set of influencers is maintained for each place category, to increase the accuracy of the recommendations, and the sets of influencers per place category are maintained in the user profile. The QoS-based algorithm considers only the qualitative characteristics of each place. Additionally, for each user we store in her profile the average values of the QoS attributes (cost, service and atmosphere) of the places she visits for different places categories (museums, bars, etc.). This enables us to determine how close each place is to the visiting habits of the particular user.

In order to combine the QoS-based recommendation and the CF-based recommendation into a single recommendation for the user, we use the $WCombSUM_i$ formula [39]. According to this formula, the overall score for an item i within the final recommendation for user u is

$$WCombSUM_{i,u} = w_{CF,C(i),u} * score_{CF,i,u} + w_{QoS,C(i),u} * score_{QoS,i,u} \tag{2.9}$$

where $score_{CF,i,u}$ and $score_{QoS,i,u}$ are the recommendation scores for item i produced for u by the CF-based and the QoS-based algorithm respectively and $C(i)$ denotes the category of item i. $w_{CF,C(i),u}$ and $w_{QoS,C(i),u}$ are weights assigned to the CF-based and the QoS-based algorithm respectively. In order to provide highly personalized recommendations, algorithm weights are computed individually for each user and category, i.e. two distinct users may have different weights for the same category, while different weights may apply for a particular user u when considering distinct categories c_1 and c_2, (e.g. "museums" and "restaurants"). Weight values are computed using the following formulas:

$$\begin{aligned} w_{CF,C(i),u} &= \frac{|PlacesVisited_{C(i),u} \cap PlacesVisitedByInfluencers_{C(i),u}|}{|PlacesVisitedByInfluencers_{C(i),u}|} \\ w_{QoS,C(i),u} &= 1 - w_{CF,C(i),u} \end{aligned} \tag{2.10}$$

Effectively, $w_{CF,C(i),u}$ is the ratio of the places visited by u within C_i and have been recommended by influencers to the overall number of places visited by u within C_i. Clearly, the higher this ratio, the more receptive u is to suggestions made by influencers, hence the weight assigned to the CF-based algorithm increases.

Regarding the calculation of the set $PlacesVisitedByInfluencers_{C(i),u}$, a visit is considered to have been suggested by an influencer if (a) an influencer has visited the same place two years or less prior to the user's visit, (b) the user had not visited the place before the considered influencer and (c) a recommendation had been made to the user, triggered by the influencer's visit. The time frame of two years has been chosen so as to (i) include in the "influence window" two consecutive similar tourist seasons (e.g. a visit made by a user to a summer resort in August 2015 is considered to have been influenced by an influencer's visit to the same resort made in June 2014) and (ii) allow for the decaying of information that was collected long ago.

The formula computing the CF algorithm weight suffers from the cold start problem, i.e. the case that no (or very few) data are present in the system for a specific category. In more detail, if no places of a specific category have been visited by influencers, the formula is not computable; additionally, if the number of places that have been visited by influencers, within a specific category, is small, the result computed by the formula will not be indicative of how receptive a user is to her influencers' suggestions (due to lack of statistical significance). Therefore, when the cardinality of the $PlacesVisitedByInfluencers_{C(i),u}$ set is below a threshold th, we set $w_{CF,C(i),u}$ to a default value of 0.4. The value of 0.4 has been chosen based on the work presented in [38], which asserts that a value equal to 0.4 ensures that the recommendations adhere to the QoS levels desired by the user, while at the same time the opinions of the influencers have an adequately strong effect in the formulation of the final recommendation. In our work, we have used a value of th equal to 10, since this has been experimentally proven to be a sufficient number of elements to generate an acceptably accurate value for $w_{CF,C(i),u}$.

2.4 The Leisure Time Recommendation Algorithm

Having available the information listed in Sect. 2.3, the algorithm performs the three steps listed below.

Step 1—Offline Initialization. The algorithm is initially bootstrapped by executing the following actions:

- for each place category C, the minimum and maximum place cost in the category are identified, using the formulas shown in Eq. (2.12). Similarly, the $minSer(C)$ and $maxSer(C)$ *and* $minAtm(C)$ and $maxAtm(C)$ quantities are computed, corresponding to the minimum and maximum service and atmosphere of places in category C, respectively.

$$\begin{aligned} minCost(C) &= \min_{place_i \in C}(cost(place_i)) \\ maxCost(C) &= \max_{place_i \in C}(cost(place_i)) \end{aligned} \tag{2.11}$$

- for each user u and place category C, the algorithm computes the values of $w_{CF,C(i),u}$ and $w_{QoS,C(i),u}$ according to the formulas presented in Sect. 2.3.5, and the mean cost, service and atmosphere of the places within category C that user u has visited in the past.
- for each user u and place category C, the algorithm computes the influence level of her social friends (c.f. Sect. 2.3.1), and then retains the *top N* social friends with the highest influence level in C. In this work, we set $N = 6$, since we have experimentally determined that this value is adequate for producing accurate recommendations; this experiment is described in Sect. 2.5.1.
- for each pair of places (p_1, p_2), the recommendation algorithm computes the place similarity between p_1 and p_2. The place similarity metric takes into account both the semantic similarity between the places and the similarity between the locations within which the places are located and is computed as

$$PlaceSim(p_1, p_2) = SemSim(p_1, p_2) * LocSim(loc(p_1), loc(p_2)) \tag{2.12}$$

where *SemSim* (p_i, p_j) is the semantic similarity between places p_i and p_j (c.f. Sect. 2.3.2, $loc(p)$ denotes the location in which p is located, and *LocSim* (loc_i, loc_j) denotes the similarity of locations loc_i and loc_j. Recall than the *LocSim* metric encompasses both the physical and the thematic distance between the locations (c.f. Sect. 2.3.3).

Step 2—Online operation: After the initialization phase, the algorithm can be executed in an online fashion to produce recommendations. The algorithm's execution is triggered by events generated by users: in particular, the algorithm considers the generation of recommendations each time a user checks-in a leisure time place or is tagged to be in some leisure time place.

When a user *infl* checks-in or is tagged to be in a leisure time place p that belongs in category C, the algorithm considers to make a recommendation to those users that are influenced by *infl* on C. To this end, the algorithm computes the set *PRR*(C, *infl*) of Potential Recommendation Recipients as follows

$$PRR(C, infl) = \{u | infl \in influencers(u, C)\} \tag{2.13}$$

Subsequently, for each user u in *PRR* (C, *infl*) the algorithm computes which recommendation should be sent to the particular user. The rationale to formulate the recommendation to be sent is as follows:

- If the QoS parameters of p are "close" to the QoS attributes of places that user u typically visits within C, then the algorithm checks if p might be of interest to the user, considering the opinion of u's influencers in C and the QoS parameters of p. If the algorithm determines that p might be of interest to user u, then p is recommended to u.

- If the QoS parameters of p are too distant from the QoS attributes of places that user u typically visits within C, then the algorithm searches for a place p' in C that (a) is highly similar to p and (b) its QoS attributes are close to the QoS attributes of places that u typically visits within C. Then, the algorithm checks if p' could be of interest to u, considering the opinion of u's influencers within C, the similarity between p and p' and the QoS parameters of p'. If the algorithm determines that p' might be of interest to u, then p' is recommended to u.

In more detail, initially, the QoS score of place p for user u is computed as follows:

$$score_{QoS,p,u} = cost\_vicin(u,p) * ser\_vicin(u,p) * atm\_vicin(u,p) \tag{2.14}$$

where

$$\begin{aligned} cost\_vicin(u,p) &= 1 - \frac{|cost(p) - MC(u,C)|}{maxCost(C) - minCost(C)} \\ ser\_vicin(u,p) &= \begin{cases} 1 - \frac{|ser(p) - MS(u,C)|}{maxSer(C) - minSer(C)}, & if\ ser(p) \le MS(u,C) \\ 1, & if\ ser(p) > MS(u,C) \end{cases} \\ atm\_vicin(u,p) &= \begin{cases} 1 - \frac{|atm(p) - MA(u,C)|}{maxAtm(C) - minAtm(C)}, & if\ atm(p) \le MA(u,C) \\ 1, & if\ atm(p) > MA(u,C) \end{cases} \end{aligned} \tag{2.15}$$

In the equations above, *cost*(p) is the average cost (entrance ticket or cost of items that are sold there) of p, *ser*(p) and *atm*(p) are the service atmosphere ratings of p (all values are copied from sites such as www.tripadvisor.com), while *MC*(u, C), *MS*(u, C) *and MA* (u, C) are the mean cost, mean service and mean atmosphere respectively of places visited by u within C. Cost vicinity indicates how close the place price is to the user's price habits within the specific category.

When computing service vicinity or atmosphere vicinity, we consider a place close to the user's preferences if its service is either equal to or surpassing the mean service (respectively atmosphere) rating of the places that the user visits in this category. The rationale behind this calculation is that places having received high marks for these criteria would typically be of higher interest to the user. For places having service less than *MS*(u, C) [respectively, atmosphere less than *MA*(u, C)], a typical normalized distance metric [31] is employed.

If $score_{QoS,i,u}$ is greater than 0.68 (a discussion on the choice of this value is given in the conclusions section), then the QoS parameters of p are considered to be close to the QoS attributes of places that u typically visits within C. In this case, the algorithm computes the CF-based score for the recommendation of p: it first extracts the N influencers regarding C from the u's profile and, for each of these influencers IN locates the place p_{IN} within C that she has visited and has the greatest place similarity with p. Then, the CF-based score $score_{CF,p,u}$ is computed as

$$score_{CF,p,u} = \frac{\sum_{IN \in influencers(u,C)} *PlaceSim(p, p_{IN})}{\sum_{IN \in influencers(u,C)} IL_{u,IN}(C)} \tag{2.16}$$

Finally, the probability that u is interested in p is computed as follows:

$$IP_{u,p} = w_{CF,C(p),u} * score_{CF,p,u} + w_{QoS,C(p),u} * score_{QoS,p,u} \tag{2.17}$$

where $w_{CF,C(p),u}$ and are $w_{QoS,C(p),u}$ are the weights assigned to the CF and QoS dimension regarding recommendations made to u for category $C(p)$ (i.e. the category of the place appearing in the event—for details on the computation of these weights, c.f. Sect. 2.3.5). If the value of $IP_{u,p}$ meets or exceeds the interest probability (*IP*) threshold (Sect. 2.5.3 discusses the computation of the *IP* threshold's value), p is recommended to user u.

If $score_{QoS,i,u}$ is less than 0.68, the QoS levels of p are considered distant from those that u typically visits in C. In this case, the algorithm searches to locate a place p' within C that is similar to p and has QoS levels close to the habits of u. This is achieved by finding the place with the maximum value

$$PlaceSim(p, p') * score_{QoS,p',u} \tag{2.18}$$

For p', the *IP* metric is computed as described above, and if it is higher than the *IP* threshold, p' is recommended to u.

It is worth noting that in the absence of a history of visits made by a user (i.e. a new user or a user that recently visited her first place in this particular category), the quantities *MC* (u, C), *MS* (u, C) *and MA* (u, C) used for the computation of the QoS-based score are computed taking into account the relevant quantities of the user's influencers in the particular category. More specifically *MC* (u, C) is computed as

$$MC(u, C) = \frac{\sum_{IN \in influencers(u,C)} IL_{u,IN}(C) * MC(IN, C)}{\sum_{IN \in influencers(u,C)} IL_{u,IN}(C)} \tag{2.19}$$

where $IL_{u,IN}(C)$ is the influence level of user *IN* on user u regarding category C (c.f. Sect. 2.3.1). Analogous computations are performed for *MS* (u, C) and *MA* (u, C).

Step 3—Repository update. Since both the content of the social networks and the places information are dynamic, a number of information elements of our model needs to be updated, in order to maintain consistency with the current status of the social network and leisure time place information. The updates that need to be performed are as follows:

1. Each time a new place is introduced, a check needs to be made whether the minimum and maximum values of the QoS parameters within that category need to be updated.

2. After a user checks-in a place or is tagged to be there, the profile of the user is updated regarding the mean QoS attributes (cost, service and atmosphere) of the places category that the place belongs to.
3. The weights assigned to the CF dimension ($w_{CF,C(p),u}$) and the QoS-based dimension ($w_{QoS,C(p),u}$) need to be recomputed when the underlying data (places that a user has checked-in or has been tagged to be there, and visits to places suggested by influencers) change.
4. Each time a new place is introduced, place similarities between the new place and all places in the database are also computed.
5. Finally, the top influencers of each user u within each category of interest C need to be recomputed when the underlying data (categories of interest and/or number of communications) change.

Updates (1) and (2) include only basic computations hence they can be performed synchronously with the triggering event. Updates (3)–(5) are more computationally expensive and can be performed in an offline fashion, e.g. be executed periodically.

2.5 Experimental Evaluation

In this section, we report on our experiments through which:

1. We determined the values of parameters that are used in the algorithm. More specifically, an initial experiment was conducted to evaluate the optimal value for parameter N, expressing the number of influencers that must be maintained per place category so as to provide accurate recommendations. A second experiment was targeted to estimate the taxonomy level of categories that should be retained within the profile of each user (i.e. the taxonomy level of place categories for which we maintain the weights $\mathrm{Cf}_{\mathrm{weight}}$ and $\mathrm{QoS}_{\mathrm{weight}}$, as well as the top influencers for each user), in order to assess the storage requirements and the scalability of the approach.
2. We evaluated the performance of the proposed approach, both in terms of execution time (the time needed to make the recommendations) and users' satisfaction regarding the offered recommendations.

For our experiments we used two machines. The first was equipped with one 6-core Intel Xeon E5-2620@2.0GHz CPU and 16 GB of RAM, which hosted the processes corresponding to the active users (browser emulators), i.e. the users who generated the triggering events. The second machine's configuration was identical to the first, except for the memory which was 64 GB; this machine hosted (i) the algorithm's executable, (ii) a database containing the users' profiles including the influence metrics per category, the lists of top N influencers per category and the data regarding the tags and check-ins made by each user and (iii) the places database, which includes their semantic information and QoS data (cost, service and atmosphere). The machines were connected through a 1 Gbps local area network.

To assess recommendation quality, we conducted a user survey in which 60 people participated. The participants were students and staff from the University of Athens community, coming from 4 different academic departments (computer science, medicine, physics and theatre studies). 29 of the participants were women and 31 were men, and their ages range between 18 and 48 years old, with a mean of 29. All of the participants were Facebook users, and we extracted the profile data needed for the algorithm operation using the Facebook Graph API (https://developers.facebook.com/docs/graph-api). Regarding the participants' profile and behavior within Facebook, the minimum number of Facebook friends among the participants was 148 and the maximum was 629 (with a mean of 264). All participants used Facebook at least 4 days a week and one hour per day of use, and had been members for at least two years. For each person, we computed the relevant tie strengths with all of her Facebook friends in an offline fashion.

The places data used in the experiment were extracted from Tripadvisor. The data set consisted of 5000 places in 20 cities (including New York, Los Angeles, London, Rome, Paris, Dubai, Athens and Beijing) and falling in 10 places categories (museums, religious/historical monuments, bars, nightclubs, cinemas, theatres, fast food restaurants, cafés and restaurants). The cost attribute values in this repository were set according to the places' current prices, while the service and atmosphere attribute values were set according to the users' rating summary from Tripadvisor.

2.5.1 Determining the Number of Influencers

The first experiment aimed to determine the number of influencers *N* that must be maintained by the system per user and per category, in order to produce accurate recommendations; it is desirable that this number is kept to a minimum, to save space and limit the amount of data to be processed, thus reducing recommendation time. Recall that for each place category, the algorithm considers the opinions of the strongest influencers within the specific category when recommendations are generated. To find the minimum number of influencers that can be kept without limiting the quality of recommendations, we gradually increased the number of strongest influencers maintained, seeking the point at which considering more influencers does not modify the generated recommendations. The generated recommendations will converge when the number of considered influencers increases, because stronger influencers are added first to the set of considered influencers, hence increments beyond some point will only lead to incorporation of weaker influencers, and this incorporation will not result to modification of the generated recommendation.

To identify the value of *N* after which recommendations remain stable, we generated 1000 synthetic (*user, category*) pairs to formulate recommendations for.

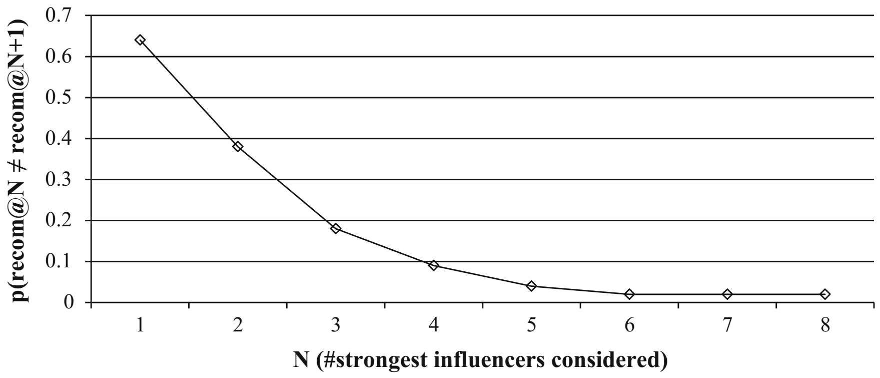

Fig. 2.2 Different recommendations made, due to the fact of considering 1 more recommender

Subsequently, we iterated on the generation of the recommendations, for different values of *N* varying from 1 to 10. Finally, we calculated the probability that a recommendation generated for a user considering her *N* strongest influencers (*recom@N*) is different than the corresponding recommendation generated considering her *N*+1 strongest influencers (*recom@N*+1); this probability is denoted as $p(recom@N \neq recom@N+1)$. The results, depicted in Fig. 2.2, show that any further increments beyond the number of 6 influencers result only in marginal modifications to the recommendations (98% of the recommendations remain unmodified). Therefore, in the following experiments we fix the number of *N* to 6.

2.5.2 Estimating the Taxonomy Level of Places Categories of Interest per User

To estimate the taxonomy level of places categories of interest per user, we conducted an experiment with our 60 participants. In this experiment, we varied the taxonomy level maintained in the profiles using the following values: level 0 (average of all the places, in general), level 1 (taxonomy level examples: leisure, restaurant, attraction, etc.), level 2 (examples: nightclubs, cinemas, fast food restaurants, cafés, museums, etc.) and level 3 (examples: Asian restaurants, operas, internet cafés, folk museums, etc.). Then, for each setting, 10 recommendations were generated for each user, and the user was asked to assess the probability that she would visit the recommended place.

The results, depicted in Fig. 2.3, show that when taxonomy is considered at level 0, visiting probability is very low (3%). Considering the taxonomy at level 1 raises visiting probability to 14%, while considering levels 2 and 3 raises further visiting probability to 30 and 35%, respectively.

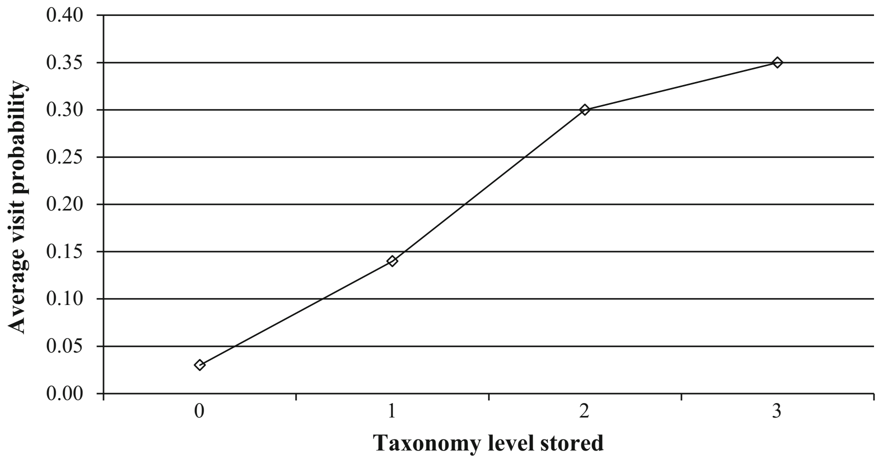

Fig. 2.3 Visit probability when information is stored at different taxonomy levels

Using the fourth level of the taxonomy (actual places) was not considered, since in this case the algorithm could produce only few useful recommendations: recall the algorithm recommends to users places in categories they are interested in, with interest category population relying on the interest lists collected by the social network. Therefore, for a place to be recommended the particular place would have to be included in such an interest list, which would typically require that the user has either visited the place's profile in the social network or has engaged in some activity or conversation about it and thus is already aware of its existence (and thus the recommendation offers little or no new information).

Regarding the data size, if we choose to maintain user preferences at taxonomy level-2, we need to store approximately 4 KB per user. However, if we choose to maintain user preferences at taxonomy level-3, storage requirements raise to approximately 100 KB per user (in the worst case scenario, where our user has an interest in all the 2000 kinds of places; the number of 2000 corresponds to the count of level-3 taxonomy branches in the SN user preference database). Hence, in the subsequent experiments we will store our data in taxonomy level-3 detail, since we can achieve better accuracy and—at the same time—the related storage requirements can be handled by the current technology.

2.5.3 Interest Probability Threshold

As discussed in Sect. 2.4, the algorithm uses an interest probability threshold to decide whether some recommendation is of interest to the user. To determine the threshold, we exploited the data gathered in the experiment described in Sect. 2.5.2. Figure 2.4 displays the relation of the interest probability metric (IP) introduced in

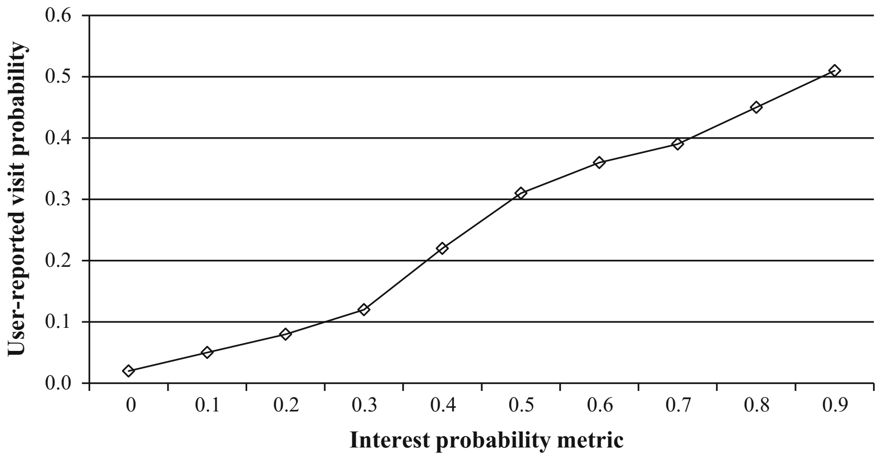

Fig. 2.4 Visit probability against interest probability metric

Sect. 2.4 to the user-reported probability for following the recommendation. From this figure, we can determine that there is positive correlation between the IP metric and the probability that the user actually follows the recommendation. Clearly, recommendations with a high IP metric are likely to be followed by users (and should thus be forwarded to them) while those with low IP metric will probably be ignored (and should thus be suppressed).

The optimal setting for the IP threshold would (a) maximize the probability that a forwarded recommendation is useful (i.e. it would be followed) and (b) maximize the probability that a recommendation that would be followed is not suppressed. In a formal notation, item (a) corresponds to maximizing the quantity

$$Prec = \frac{|\{useful\_recommendations\} \cap \{forwarded\_recommendations\}|}{|\{forwarded\_recommendations\}|} \tag{2.20}$$

while item (b) corresponds to the maximization of the quantity

$$Rec = \frac{|\{useful\_recommendations\} \cap \{forwarded\_recommendations\}|}{|\{useful\_recommendations\}|} \tag{2.21}$$

These goals are contradicting: goal (a) suggests that a high threshold is used, so that only the recommendations of high interest (and thus of a high probability to be followed) are forwarded to the user. On the contrary, goal (b) suggests that a low threshold is used, so that users are not deprived of recommendations that they would follow, even if the relevant IP metric value is small. Figure 2.5 illustrates this contradicting relation of the *Prec* and *Rec* metrics to the IP threshold.

To best serve both these contradicting goals, we seek to maximize their harmonic mean, i.e. the quantity

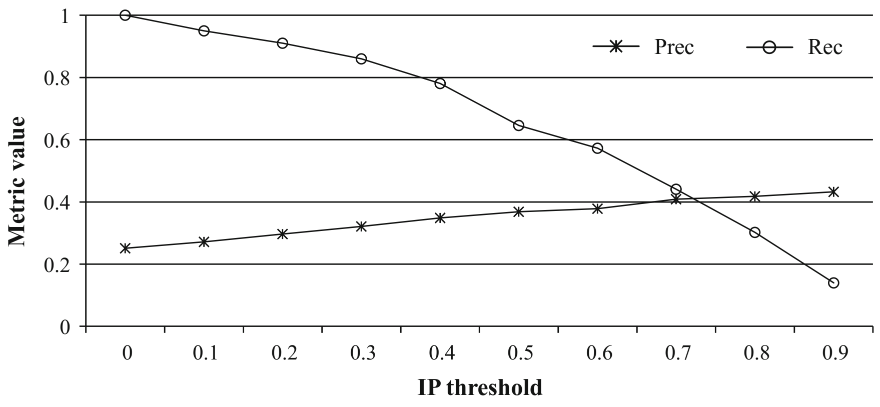

Fig. 2.5 Relation of *Prec* and *Rec* metrics to the IP threshold

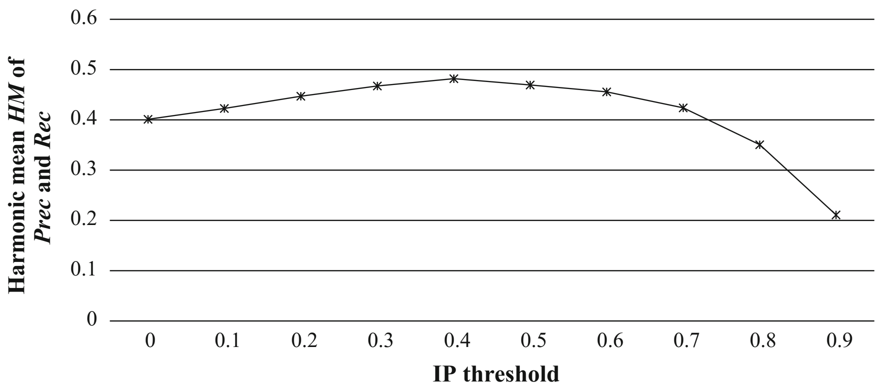

Fig. 2.6 Relation of the harmonic mean of *Prec* and *Rec* metrics to the IP threshold

$$HM = 2 * \frac{Prec * Rec}{Prec + Rec} \tag{2.22}$$

analogously to the combination of the precision and recall metrics in information retrieval, which produces the F1-measure [40]. As shown in Fig. 2.6, the value of the IP threshold that maximizes the harmonic mean is 0.4; hence, in the following experiments we will set the IP threshold to this value.

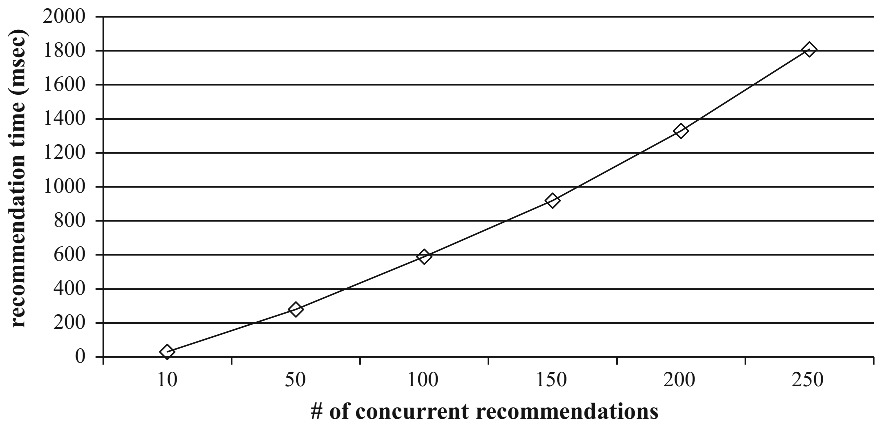

Fig. 2.7 Time needed for recommendation formulation

2.5.4 Recommendation Formulation Time

The next experiment is aimed at measuring the time needed for recommendation formulation when a triggering event occurs (users check-in or are tagged). To measure the time needed, we created a synthetic user base consisting of 50,000 users. Each user had from 100 to 1000 friends overall, with an average of 190 friends, following the mean value of friends on Facebook (https://web.facebook.com/notes/facebook-data-team/anatomy-of-facebook/10150388519243859?_rdr). Each user was set to have a history of 0–100 visits. All repositories (the places' semantic repository, the places' qualitative repository, the users' top recommenders and each user's past visits) were implemented as in-memory hash-based structures, which proved more efficient than using a separate database, such as HSQLDB (http://hsqldb.org/) (memory-based) or MySQL (http://www.mysql.com) (disk-based).

The measurements obtained from this experiment are depicted in Fig. 2.7. We can observe that the time needed remains low even for high degrees of concurrency (approximately 0.6 s for 100 concurrent recommendations) and scales linearly with the concurrency degree. Even when 250 recommendations need to be concurrently generated (250 users have simultaneously checked-in or been tagged), the average recommendation time is less than 2 s, which is deemed satisfactory for the infrastructure capacity. Note that the corresponding Facebook infrastructure was estimated to over 60,000 servers for approximately 608 million users in June 2010 [41], which gives about 10,000 users per server; in our experiment, a mid-range workstation was set to handle 50,000 users.

2.5.5 User Satisfaction

The final experiment is aimed at assessing the participants' satisfaction regarding the recommendations they received from the algorithm presented in Sect. 2.4, and at comparing this satisfaction level to that obtained from other algorithms.

In this experiment, each participant was asked to rate 40 recommendations presented to them, on a scale of 1 (totally unsatisfactory—"there is no way I would visit this place") to 10 (totally satisfactory—"I will definitely visit this place"). The recommendations offered to the users covered 90% of the taxonomy level-2 category of places (from bars, pizza places and museums, to casinos and zoos). The 40 recommendations assessed by each user were generated using five different algorithms, with each algorithm having generated 8 of the recommendations. The algorithms that were used to generate the recommendations are:

1. the proposed algorithm,
2. the proposed algorithm, modified to consider profile and place information at taxonomy level 2, instead of taxonomy level 3,
3. a plain CF algorithm (the algorithm in Sect. 2.4 taking only the cumulative influence into account and not considering the QoS dimension),
4. a plain QoS-based algorithm (the algorithm in Sect. 2.4 without the CF dimension) and
5. the proposed algorithm, but without considering per-category influencers for each user, but using a single set of influencers per user, across all categories.

Recommendations were presented to the users for assessment in randomized order.

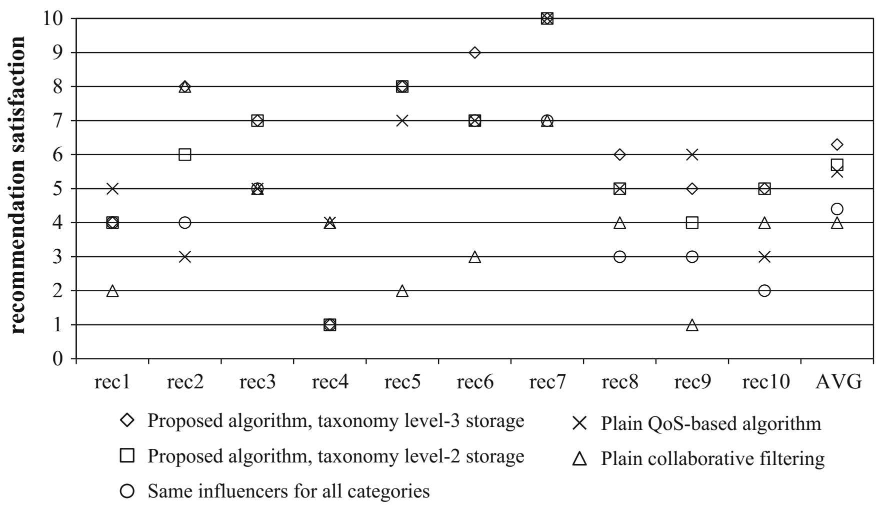

Fig. 2.8 Users' satisfaction of recommendations made by individual recommendation algorithms

Figure 2.8 depicts the participants' satisfaction regarding the recommendations they received, as measured in this experiment. On average (last column on Fig. 2.8) it is clear that the proposed algorithm using the taxonomy level-3 category storage outperforms the other algorithms, attaining an overall user satisfaction of 6.3. The proposed algorithm using the taxonomy level-2 comes in second with an overall user satisfaction of 5.7, or 90% of the satisfaction of the same algorithm with the taxonomy level-3 category storage. The plain QoS-based algorithm was ranked 3rd, the proposed algorithm modified to use a single set of influencers was ranked 4th, and the plain collaborative was ranked 5th, with their satisfaction being at the 87, 70 and 63%, respectively of the proposed algorithm with the taxonomy level-3 category storage. This experiment clearly shows that using a specialized set of recommenders for each place category provides a significant improvement in the quality of the generated recommendations.

Within Fig. 2.8 we have also included user ratings for 10 individual recommendations (rec1–rec10); these have been chosen to demonstrate that algorithm performance is not uniform across all cases. In our future work, we will further investigate cases where the proposed algorithm's recommendation received a poor rating (inferior to the ratings of other algorithms or lower than 5 out of 10).

2.6 Conclusions and Future Work

In this chapter we have presented a knowledge-based algorithm for generating leisure time place recommendation for SN, taking into account (a) qualitative characteristics, (b) the users visiting habits, (c) the influencing factors between social network users and (d) semantic information, concerning both the places to be recommended and the places' geographic locations. The recommendation algorithm follows the metasearch algorithm paradigm, using two different ranking algorithms, the first examining only the qualitative aspects of places and the related users' habits and the second being based on CF techniques. The rankings produced by these two algorithms are combined to generate the overall ranking, which is then used for generating the recommendations.

The proposed algorithm contributes to the state-of-the-art by considering qualitative aspects of the places, the influencing factors between social network users, the social network user past behavior regarding their visits in different place categories, and the semantic categorization of the places to be recommended. Furthermore, influencers in this algorithm are considered per category, to allow for formulation of more accurate recommendations and maximize the probability that recommended place are visited. The proposed algorithm has been experimentally validated regarding (i) its performance, and (ii) recommendation accuracy (users' satisfaction to the recommendations produced) and the results are encouraging.

Regarding our future work, we plan to conduct a user survey with a higher number of participants and more representative demographics. The current participants set was drawn from the University of Athens community, hence it is not a representative sample of the overall population; thus, the results drawn need further verification regarding their generalizability. A more comprehensive survey will tackle this issue and allow us to gain better insight into the satisfaction and needs of users with more diverse profiles. The planned survey will allow deeper analysis regarding the value of 0.68, which has been used in the experiments described in Sect. 2.5 as the QoS similarity threshold for places. This value has been derived by a small-scale experiment, in which participants were asked to rate whether 100 items were "close" or not to their QoS preferences, and then the QoS threshold was set to the value that maximized the QoS-predictions' F1-measure [40]. The QoS-prediction considered here is that a place p is considered "close" if its $score_{QoS,p,u}$ is greater than or equal to the QoS threshold and "not close" if its $score_{QoS,p,u}$ is less than the QoS threshold. The extended survey will allow us to obtain a more comprehensive dataset regarding the "closeness" perception and further investigate this issue. The main aspects of this investigation are (a) the value of the QoS threshold and (b) whether the QoS threshold is uniform across all categories and/or user profiles.

Furthermore, the QoS threshold mechanism leads to a behavior that the user is confined to viewing only information about places similar to those she has visited in the past, limiting thus the serendipity that may stem from recommendations, which is a desirable feature of RS [42]. To this end, means for allowing serendipity in recommendations, such as hybrid systems [23] or item-to-item mechanisms [15] will be examined. To alleviate the grey sheep problem, we will consider performing comparisons at higher taxonomy levels, when comparisons at the third level of the taxonomy lead to very small numbers of near neighbors.

We finally plan to take into account keywords of the descriptions ("hated that place", "I will never go again", "I loved that place", "best café ever", "perfect atmosphere", etc.) that follow check-ins and tags, as well as asking users to explicitly evaluate the QoS attributes of each place they visit, in order to achieve more accurate recommendations.

References

1. Burke, R.: Knowledge-based recommender systems. In: Kent, A. (ed.) The Encyclopedia of Library and Information Science. Marcel Decker Inc., U.S. (2000)
2. Blanco-Fernández, Y., Pazos-Arias, J.J., Gil-Solla, A., et al.: A flexible semantic inference methodology to reason about user preferences in knowledge-based recommender systems. Knowl.-Based Syst. **21**(4), 305–320 (2008)
3. Aggarwal, C.C.: Knowledge-based recommender systems. In: Recommender Systems. Springer, Berlin. ISBN: 978-3-319-29657-9
4. Facebook: Facebook ad targeting. https://www.facebook.com/business/products/ads/ad-targeting (2015)
5. He, J., Chu, W.W.: A social network-based recommender system (SNRS). Ann. Inform. Syst. **12**, 47–74 (2010)
6. Arazy, O., Kumar, N., Shapira, B.: Improving social recommender systems. IT professional, September (2009)

7. Oechslein, O., Hess. T.: The value of a recommendation: the role of social ties in social recommender systems. In: 47th Hawaii International Conference on System Science (2014)
8. Quijano-Sanchez, L., Recio-Garcia, J.A., Diaz-Agudo, B.: Group recommendation methods for social network environments. In: 3rd Workshop on Recommender Systems and the Social Web within the 5th ACM International Conference on Recommender Systems (RecSys'11) (2011)
9. Boulkrinat, S., Hadjali, A., Mokhtari, A.: Enhancing recommender systems prediction through qualitative preference relations. In: 11th International Symposium on Programming and Systems (ISPS), pp. 74–80 (2013)
10. Margaris, D., Georgiadis, P., Vassilakis, C.: A collaborative filtering algorithm with clustering for personalized web service selection in business processes. In: Proceedings of the IEEE 9th RCIS Conference, Athens, Greece (2015)
11. Bakshy, E., Rosenn, I., Marlow, C., Adamic L.: The role of social networks in information diffusion. In: Proceedings of the 21st International Conference on World Wide Web, pp. 519–528 (2012)
12. Bakshy, E., Eckles, D., Yan, R., Rosenn I.: Social influence in social advertising: evidence from field experiments. In: Proceedings of the 13th ACM Conference on Electronic Commerce (2012)
13. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The Adaptive Web, LNCS vol. 4321, pp. 291–324 (2007)
14. Zhang, W., Chen, T., Wang, J., Yu, Y.: Optimizing top-n collaborative filtering via dynamic negative item sampling. In: Proceedings of the 36th International ACM SIGIR (SIGIR'13), pp. 785–788 (2013)
15. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM TOIS **22**(1), 5–53 (2004)
16. Balabanovic, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Commun. ACM **40**(3), 66–72 (1997)
17. Rodríguez-González, A., Torres-Niño, J., Jimenez-Domingo, E., Gomez-Berbis, M.J., Alor-Hernandez, G.: AKNOBAS: A knowledge-based segmentation recommender system based on intelligent data mining techniques. Comput. Sci. Inform. Syst. **9**(2), (2012)
18. Monfil-Contreras, E.U., Alor-Hernández, G., Cortes-Robles, G., Rodriguez-Gonzalez, A., Gonzalez-Carrasco, I.: RESYGEN: a recommendation system generator using domain-based heuristics. Expert Syst. Appl. **40**(1), 242–256 (2013)
19. Konstas, I., Stathopoulos, V., Jose, J.M.: On social networks and collaborative recommendation. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. Boston, USA (2009)
20. Jamali M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the fourth ACM Conference on Recommender Systems, RecSys 2010. Barcelona, Spain (2010)
21. Zheng, Y., Xie, X.: Learning travel recommendations from user-generated GPS traces. ACM Trans. Intell. Syst. Technol. (TIST) 2.1 (2011)
22. Bao J., Zheng Y., Mokbel M.: Location-based and preference-aware recommendation using sparse geo-social networking data. In: Proceedings of the 20th International Conferences on Advances in Geographic Information Systems, SIGSPATIAL'12, pp. 199–208 (2012)
23. Colombo-Mendoza, L.O., Valencia-García, R., Rodríguez-González, A., Alor-Hernández, C., Samper-Zapaterd, J.J.: RecomMetz: a context-aware knowledge-based mobile recommender system for movie showtimes. Expert Syst. Appl. **42**(3), 1202–1222 (2015)
24. Yang, W.-S., Hwang, S.-Y.: iTravel: a recommender system in mobile peer-to-peer environment. J. Syst. Softw. **86**(1), 12–20 (2013)
25. Moreno, A., Valls, A., Isern, D., Marin, L., Borràs, J.: SigTur/E-destination: ontology-based personalized recommendation of tourism and leisure activities. Eng. Appl. Artif. Intell. **26**(1), 633–651 (2013)
26. Ference, G., Mao, Y., Lee, W-C.: Location recommendation for out-of-town users in location-based social networks. In: Proceedings of ACM CIKM13, pp. 721–726 (2013)

27. Gilbert, E., Karahalios, K.: Predicting tie strength with social media. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09), pp. 211–220 (2009)
28. Anagnostopoulos, A., Kumar, R., Mahdian, M.: Influence and correlation in social networks. In: Proceedings of the 14th ACM SIGKDD (KDD'08), pp. 7–15 (2008)
29. Facebook: Facebook interest targeting, https://www.facebook.com/help/188888021162119 (2015)
30. Chedrawy, Z., Abidi, S.S.R.: A web recommender system for recommending, predicting and personalizing music playlists. In: Proceedings of Web Information Systems Engineering (WISE 2009), pp. 335–342 (2009)
31. Aslam, J., Montague, M.: Models for metasearch. In: Croft, W.B., Harper, D.J., Kraft, D.H., Zobel, J. (eds.) Proceedings of the 24th Annual International ACM SIGIR 2001, pp. 276–284 (2001)
32. Pirasteh, P., Jung, J.J. Hwang, D.: Item-based collaborative filtering with attribute correlation: a case study on movie recommendation. In: 6th Asian Conference, ACIIDS 2014, Bangkok, Thailand, 7–9 April 2014, Proceedings, Part II, pp. 245–252 (2014)
33. Androutsos, D., Plataniotis, K.N., Venetsanopoulos, A.N.: Distance measures for color image retrieval. In: Proceedings of the International Conference on Image Processing, vol. 2, pp. 770–774 (1998)
34. Jones, C.B., Alani, H., Tudhope, D.: Geographical information retrieval with ontologies of place. In: Proceedings of the Conference on Spatial Information Theory, COSIT'01, pp. 322–335 (2001)
35. Word2Vec Library. https://code.google.com/archive/p/word2vec/ (2013)
36. ITU. Recommendation E.800 quality of service and dependability vocabulary (1988)
37. Mersha, T., Adlakha, V.: Attributes of service quality: the consumers' perspective. Int. J. Serv. Ind. Manage. **3**(3), 34–45 (1992)
38. Margaris, D., Vassilakis, C., Georgiadis, P.: An integrated framework for adapting WS-BPEL scenario execution using QoS and collaborative filtering techniques. Sci. Comput. Program. **98**, 707–734 (2015)
39. He, D., Wu, D.: Toward a robust data fusion for document retrieval. In: IEEE 4th International Conference on Natural Language Processing and Knowledge Engineering—NLP-KE (2008)
40. Lipton, Z.C., Elkan, C., Naryanaswamy, B.: Optimal thresholding of classifiers to maximize F1 measure. In: Proceedings of ECML PKDD 2014 (part II), pp. 225–239 (2014)
41. Data Center Knowledge: The Facebook data center FAQ. http://www.datacenterknowledge.com/the-facebook-data-center-faq/ (2013)
42. Ge, M., Delgado-Battenfeld, C., Jannach, D.: Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: Proceedings of RecSys '10, pp. 257–260 (2010)