Applied Intelligence (2022) 52:9045–9066
https://doi.org/10.1007/s10489-021-03117-4

Towards MANET-based Recommender Systems for Open Facilities

Jürgen Dunkel[1] · Ramón Hermoso[2]

Accepted: 14 December 2021 / Published online: 28 December 2021

Abstract

Nowadays, most recommender systems are based on a centralized architecture, which can cause crucial issues in terms of trust, privacy, dependability, and costs. In this paper, we propose a decentralized and distributed MANET-based (Mobile Ad-hoc NETwork) recommender system for open facilities. The system is based on mobile devices that collect sensor data about users locations to derive implicit ratings that are used for collaborative filtering recommendations. The mechanisms of deriving ratings and propagating them in a MANET network are discussed in detail. Finally, extensive experiments demonstrate the suitability of the approach in terms of different performance metrics.

Keywords Context-aware recommender systems · Location-based systems · MANET · ad-hoc networks · Open systems

1 Introduction

During the last decade, mobile devices has changed the way people interact with their environment. Built-in sensors and the high processing power can provide users with intelligent applications that exploit their current context.

Context-aware Recommender Systems (CARS) [1] form a new class of intelligent applications that infer the user's context from environmental features to provide refined recommendations that contribute to a personalized user experience. In particular, location-based CARS in open facilities such as museums, fairs, or general urban spaces, can benefit from the insights gained from sensor data about the current user's location. These might be used to derive more accurate recommendations to users in domains where location is a significant context attribute. This is the case in open facilities or urban open spaces where users are free to move around in search of items that satisfy their needs, and in which the set of users in the area changes over time (users may join or leave at will).

Problem description Common recommender systems do not exploit the manifold capabilities of *distributed smartphone-based* system architectures. Instead, most recommender systems rely on a *centralized* architecture: they make use of a server that acts as a central authority, collects the relevant data from all users, and calculates individual recommendations for all of them. We claim that such centralized recommender systems suffer from the following significant issues:

- *Trust issues*: users must trust that the recommendations received from a server are trustworthy and appropriate without knowing how they were calculated. This demands a big leap of faith, as centralized recommendation systems used in commercial areas such as online marketplaces or accommodation websites often skew recommendation results by giving more visibility to promoted items. In general, users would prefer to have more control over the recommendation process.
- *Privacy issues*: usually, users do not like to submit their data to a central authority, that might be interested and capable of exploiting their data for commercial purposes. Rather, they prefer that no single recommender server is owner of all data. Moreover, in general, users want to have complete control over their data and decide with whom they share their ratings. But centralized recommendation systems force users to provide them with their personal recommendations. They can calculate recommendations only for users whose ratings they know.

✉ Ramón Hermoso
rhermoso@unizar.es

Jürgen Dunkel
juergen.dunkel@hs-hannover.de

[1] Hannover University of Applied Sciences and Arts, Hannover, Germany

[2] University of Zaragoza, Zaragoza, Spain

- *Rating acquisition issues*: any recommender system needs to know the user's preferences to derive better personalized recommendations. Most systems attempt to learn preferences by asking the users to give ratings for items they visited/consumed. However, people usually do not want to give explicit feedback all the time, because it is overwhelming and time-consuming.
- *Dependability issues*: as in all server-centric architectures, a central recommendation server that stores all relevant data creates a bottleneck in the system and a single point of failure. The functionality of the system completely depends on the performance and availability of this server. In case of a break-down or a drop in performance, the entire recommender system can no longer continue to function.[1] In particular, for mission-critical applications and in emergency situations, server breakdowns, power outages or server overloads can cause critical problems [4, 5, 23].
- *Costs issues*: furthermore, sever-based architectures cause significant costs for setting-up an appropriate infrastructure and maintenance. Especially in temporary events such as fairs, high installation and maintenance costs are not affordable.

Contribution In this paper, we propose a *decentralized* and *distributed MANET-based* (Mobile Ad-hoc NETwork) recommender system for open spaces, which addresses the issues of traditional centralized recommender systems, namely trust, privacy, dependability and maintenance.

A MANET is a multi-hop wireless network whose nodes consist of mobile devices that can move freely and that does not require any server infrastructure [12]. In the proposed MANET-based system, each user uses an own instance of the recommender system running on her personal smartphone device. If users meet, they exchange their ratings, building in a step-by-step manner the database for the recommendation algorithm.

We claim that our distributed recommender system approach copes with the above-mentioned issues:

- *Trust*: in our approach, users can trust the recommendations because they run the recommendation algorithm themselves and also rank its results themselves. So, they can be sure that the ranking of recommendations will not be skewed by the interests of any other party.
- *Privacy*: in the MANET-based system, users do not have to disclosure their data to a central authority that might misuse their data, but only to persons they encounter by coincidence. In particular, they are required to publish only low-sensitive and anonymized data that does not cause much harm in case of loss. Especially cautious users who do not want to disclose their own ratings can still use the system.[2]
- *Rating acquisition*: instead of forcing the user to continuously provide ratings to the system, we propose an approach, where ratings are implicitly derived. Ratings are calculated automatically and without any user interaction by measuring the sojourn times in front of items.
- *Dependability*: a MANET-based system architecture increases the reliability of the system, because there is no centralized server, i.e. no single point of failure.
- *Costs*: a smartphone-based architecture provides a low-cost solution without any specific server and network infrastructure that requires extensive maintenance.

To our knowledge, there is no comparable approach that exploits the capabilities of a smartphone-based MANET to build a trustworthy, reliable, and cost-effective recommendation system with implicit ratings.

Note that the goal of the approach is not outperforming traditional centralized CF recommender systems but to overcome the problems stated above by using a distributed solution. In the following, we will show in detail how these goals can be achieved.

The paper is organized as follows. In Section 2 we give a short overview of our system architecture. Section 3 puts forward the conceptual framework the paper is based on, and discusses the key mechanisms for our MANET-based recommender system. Then, in Section 4, we present the results of extensive experiments to evaluate our approach. Section 5 refers to the related work existing in the literature. Finally, we summarize the paper and discuss some future lines of work in Section 6.

2 Architecture overview

In this section, we will give a short overview of our approach of a distributed Context-aware Recommender System (CARS). Before going into details in the subsequent sections, we outline the crucial properties of our architecture:

- *MANET*: our recommender system approach relies on a MANET (Mobile Adhoc NETwork) architecture that forms a peer-to-peer (P2P) system composed of the user smartphones. This implies that the users move in an open space so that they could encounter each other

[1] Of course, various sophisticated mechanisms such as server replication and load-balancing can avoid these problems. But in practice, often simpler architectures are deployed.

[2] Of Course, this assumes that there are enough other users that share their data.

while walking around. Each smartphone incorporates a recommender system in order to provide its owner with valuable suggestions. The MANET-approach provides all users with a high amount of autonomy, as well as complete independence from any centralized technical infrastructure.

- *Collaborative filtering*: we assume a collaborative filtering (CF) approach, which calculates recommendations by exploiting the collaborative power of ratings provided by multiple users. Collaborative recommender systems recognize commonalities between users on the basis of their ratings, and generate new recommendations based on inter-user comparisons. Collaborative filtering is the most familiar and most-established recommender technology in commercial systems[9]. In contrast to other approaches such as content-based recommendations, no elaborate domain-specific models of user profiles and item content must be developed, what makes CF rather easy to implement [2].
- *Distributed data*: because MANETs do not rely on servers, all the data required for CF is stored in a distributed manner on the MANET nodes. The user smartphones not only hold their own ratings, but also ratings given by other users. Since CF requires a large number of ratings from many other users to calculate appropriate recommendations, ratings have to be stored redundantly on numerous nodes.
- *Multi-hop data exchange*: in MANETs, ratings from other users can be received through a direct transfer between neighboring nodes when they encounter[3]. In order to get data from nodes that have not met directly but are further away, MANETs allow data to be exchanged in a multi-hop manner: data is transferred over multiple nodes to reach a user outside the range of coverage. Therefore, two nodes can be considered connected, if there is a multi-hop connection between them.
- *Implicit ratings*: a major problem of CF is deriving user ratings. Usually, people do not want to give continuously explicit feedback, so that more subtle and less invasive techniques should be used for reasoning their personal preferences. A possible solution is to infer user preferences from more abundant implicit feedback, which extracts opinions by observing user behavior [19]. Because our MANET-based recommender system is based on smart devices, it appears obvious to exploit their sensors to monitor user activities. As application scenario, we will present a recommender system for open facilities such as museums, fairs or zoos. For such surroundings, we propose to measure how long people stay in front of items for estimating their preferences. In past works, we implemented a recommender system for a museum [17] with a server-based architecture, which derives implicit ratings from visitor's movement data.[4]

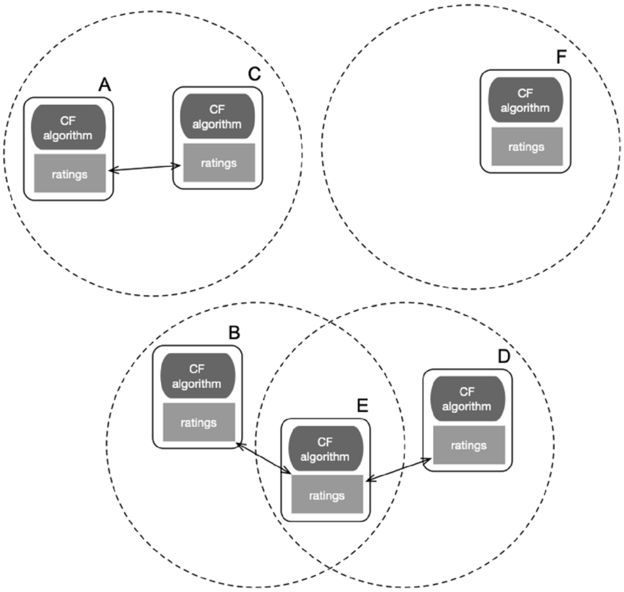


Fig. 1 Nodes in a MANET-based recommender system at a certain time instant

Figure 1 gives an example of a MANET system at a certain time instant. There are six users, all of them running their own recommender system with a CF algorithm on a smartphone. Each recommender system has its private data storage for ratings, which are own ratings derived from the smartphone's movements and ratings received from other users. If two users are passing in transmission range (for instance user *A* and user *C*) they are able to exchange their ratings. Users without any neighbor, such as user *F*, are isolated and must limit themselves to the data they already have, until they encounter other users in the near future. In the example, users *B* and *D* cannot exchange data directly, because they are out of range, but they can perform a data transfer via the MANET multi-hop mechanism using user *E* as intermediary node.

We show in Figure 2 the workflow of the system from a user's perspective using a BPMN-like notation[5]. When running the recommender system, three different events may occur. (i) If the user visits an item, her sojourn time in front of the item is measured to derive a new rating for

[3] For instance, if they are within range for establishing a Bluetooth connection.

[4] The approach presented here is not bound to this form of implicit ratings. Of course, also explicit ratings of other forms of implicit ratings can be applied.

[5] https://bpmn.org

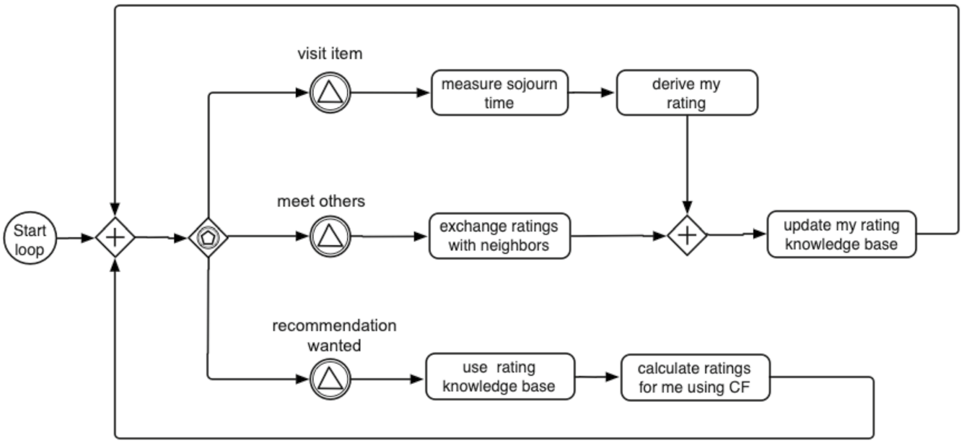

Start loop
visit item
measure sojourn time
derive my rating
meet others
exchange ratings with neighbors
update my rating knowledge base
recommendation wanted
use rating knowledge base
calculate ratings for me using CF

Fig. 2 Workflow in the MANET-based recommender from a user's perspective

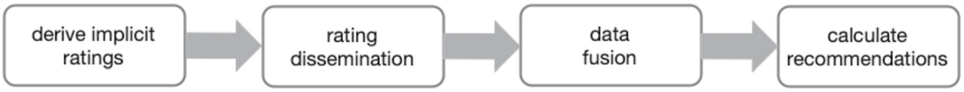


Fig. 3 Stages of the MANET recommendation mechanism from a user's perspective

her. This rating is used to complete the user's profile of her and inserted to her corresponding rating knowledge base. (ii) If the user meets some other users in her vicinity, she exchanges her data with them. The data she has received from others is then added to her rating database. (iii) If the user wants the system to recommend where to go next, she trigggers a CF recommendation algorithm that makes use of her rating knowledge base. This workflow is executed in an infinite event loop, as long as the user is in the facility.

In the following section, we will discuss in detail how to implement the aspects addressed.

3 MANET-based recommendation mechanism

In this section, we will describe the recommendation mechanisms of our proposed CARS from the perspective of a particular MANET node, i.e. a user interested in appropriate recommendations. Figure 3 shows the four stages of the recommendation process executed on each user smartphone.

1. *Derive (implicit) ratings*: as a crucial foundation of our CF-based recommender, each user must provide appropriate ratings reflecting her personal preferences. As already mentioned, we let the system to implicitly calculate a user's ratings by exploiting her movement data.
2. *Rating dissemination*: because own ratings are not sufficient for CF-based recommendations, users must get ratings from others. The rating dissemination concept describes how ratings are exchanged and propagated between MANET users to enhance their knowledge about opinions and preferences in the open space.
3. *Data fusion*: in a data fusion step, data received from other users must be merged with the information already stored in the device. In particular, the validity period of ratings must be taken into account. In public spaces, older data often has no relevance, since only recently monitored ratings describe the current situation.
4. *Calculate recommendations*: in a final step, recommendations are calculated by a CF algorithm, such as user-based collaborative filtering. The main idea is to recommend a certain user those items that other users with similar tastes liked in the past. The similarity on taste of two users is calculated based on the similarity in their rating history.

The remainder of this section is devoted to consideration of each of these stages.

3.1 Stage 1: Derive (implicit) ratings

In this subsection, we explore in more detail how implicit ratings can be determined based on a user's movements.[6] The main idea is that the longer a user stays in front of an item the more she likes this particular item; i.e. the rating on an item by a user is estimated by the sojourn time she spends close to the item. Therefore, the longer the time user spends close to the item the higher the estimated rating will be. In order to derive a meaningful rating, the average behavior of

[6] Again, note that this is only one example of a possible rating mechanism. Ratings could also be given explicitly by users, or implicitly by observing other user behavior than movements.

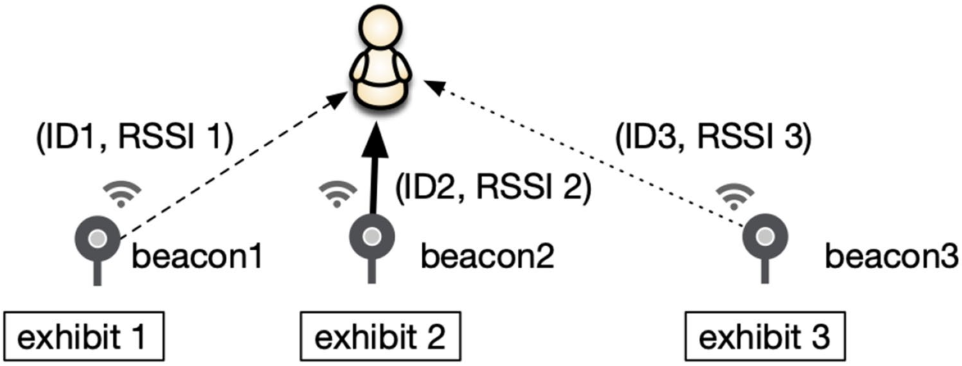


Fig. 4 Using Beacons for user localization

this user must be taken into account, e.g. whether she tends to spend a long time or a short time in front of items.

Interpreting user's sojourn times as ratings requires three steps of data preparation: (i) tracking user's movements, (ii) data cleaning of useless data for calculating sojourn times and (iii) normalizing the times to make them comparable. Note that we have already implemented a proof of concept of this approach based on a beacon infrastructure in a museum scenario [18].

3.1.1 Tracking user movements

We assume the use of iBeacon technology to monitor user localization[7]. A beacon is just a small device located at a well-known position, which sends a unique ID in a certain frequency[8] via Bluetooth LE. This beacon ID can be read by any smartphone within range[9] that scans its surroundings for beacon signals at a predefined frequency, e.g. once per second.

Figure 4 shows a situation, where a user is in the range of three different beacons, each beacon corresponding to a certain item. Now her smartphone reads the beacon signals and receives for each beacon its ID and signal strength, the so-called RSSI (received signal strength indicator). It is assumed that the beacon with the highest RSSI value is the closest one[10] that determines the user's position at the associated item. As users move through the space, the signals from the scanned beacons change and the beacon with the strongest signal defines the user's new position.

The beacon data can be used to specify the *sequence path* of user u in the space, which is the sequence of IDs from the beacons u has passed so far with the corresponding arrival times:

$$path(u) = \{(ID^u_1, t^u_1), (ID^u_2, t^u_2), ..., (ID^u_n, t^u_n)\} \tag{1}$$

where, ID^u_i denotes the ID of the i-th beacon in the trajectory of user u, and t^u_i the corresponding arrival time.

Note that for open facilities, other technologies such as GPS [24] or Wi-Fi tracking techniques might be used separately (or combined) along with a beacon infrastructure to know about users' location. For the sake of simplicity we only assume a beacon infrastructure to validate the approach.

3.1.2 Calculating sojourn times

Now, we can easily determine the *sojourn time* of user u at a certain item i by calculating the difference δ^u_i between the arrival times at item i and at the the subsequent item $i+1$ in the user's path as shown in Eq. 2.

$$\delta^u_{i,i+1} = t^u_{i+1} - t^u_i \tag{2}$$

All $\delta^u_{i,i+1}$ values below a certain threshold should not be considered, because very short times occur when users pass by items without special attention, i.e. no rating is necessary. Therefore, we clean the data by using Eq. 3,

$$s_{ui} = \begin{cases} \delta^u_{i,i+1} - \xi(u,i,i+1) & \texttt{if } delta^u_{i,i+1} - \xi(u,i,i+1) \geq \Theta \\ 0 & \texttt{otherwise} \end{cases} \tag{3}$$

where $\xi(u,i,i+1)$ is a function that approximates the walking time of user u for going from beacon i to beacon $i+1$, and Θ is a threshold time for considering the sojourn time as relevant.[11] In this way, each user determines the times she has spend at various items, which yields a list of sojourn

[7] https://developer.apple.com/ibeacon/

[8] For instance, 5 times per second.

[9] Depending on the application scenario, the beacon range could be adjusted.

[10] Normally, the smartphone repeatedly collects beacon signals over a longer time interval (between 1 and 2 seconds) to compensate for signal jitter

[11] Note that ξ function might take into account different factors, such as the distance between items, the regular walking pace of user u or the occupancy of the area.

times $\mathbf{s_u} = \{s_{ui_1}, s_{ui_2}, ..., s_{ui_m}\}$ of user u, where m denotes the number of items the user has already visited.

3.1.3 Calculate ratings

For understanding and classifying a sojourn time as a rating it must be normalized with regards to the average behavior of a user. There are users that tend to contemplate a long time at an item, while others tend to jump quickly from one item to another (which does not necessarily mean worse user satisfaction). Therefore, we have to apply a normalization step to smooth those potential different types of behavior. We use a *z-score normalization* approach calculating z_{ui} according to the following formula:

$$z_{ui} = \frac{s_{ui} - \mu_u}{\sigma_u} \tag{4}$$

with μ_u representing the mean of all non-zero sojourn times of a user u, σ_u standing for the corresponding standard deviation, and s_{ui} denoting the measured sojourn time of user u at item i. As a result of stage 1, each smartphone maintains a set of normalized sojourn times.[12] Each z_{ui} can be considered as the rating given to item i by user u. Obviously, equation 4 can be used to back-calculate normalized ratings to sojourn times.

3.2 Stage 2: Rating dissemination

As long as there are not enough ratings, traditional collaborative filtering methods cannot be applied: i.e. CF suffers from the cold-start problem [14]. Meaningful recommendations are highly dependent on the amount of ratings available to a user. In our system, users also need to collect data for some time after entering the open space before a recommendation can be made.

Therefore, a key issue of CF-based recommender systems is to provide all participants with as many ratings from other participants as possible.[13] The dissemination of ratings is not a problem in server-based systems, where all data is available at a central point without redundancy. In MANETs, however, it is more difficult to ensure that data is propagated to other nodes.

For a given node, the amount of ratings available for calculating recommendations depends on (i) how many other nodes it has met directly and (ii) how many ratings it has received via multi-hop propagation from more distant nodes. For propagating ratings, our MANET-based recommender system requires a suitable dissemination mechanism that achieves high node connectivity so that users can mutually benefit from their ratings.

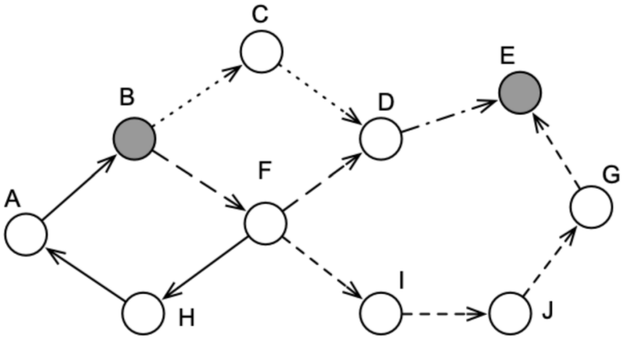


Fig. 5 Routing in MANETs

In contrast to a centralized CF approach, each user has less data available, but that data may be of more relevance. With our approach, the information exchanged tends to be more current and reflects the interests of users that are in that very moment in the facility (or are recently gone). In addition, the data usually originates from the surrounding area, because it has arrived via few MANET hops. Thus, the approach primarily considers recent and nearby ratings. For example, in the zoo scenario, exhibits should be detected that are nearby and of particular interest to the user in the current time period (e.g., a panda that just woke up from her nap). Whether the benefits of newer and more local data outweigh a larger data set depends on the application scenario, in particular how fast the state of the open facility is changing significantly.

3.2.1 MANET routing protocol

MANET routing protocols must balance connectivity against message overhead. In general, the messages are send redundantly to supply as many nodes as possible with data. But too generous forwarding of messages, e.g. via broadcast, results in a massive amount of messages with the same content. The so-called *broadcast storm* causes a large amount of redundant messages that consume scarce resources such as bandwidth and power and may cause contention [26].

Figure 5 gives an example for a MANET with 10 nodes. In the graph, nodes are connected with an arrow, if they are in range for data exchange.[14] Assuming that we want to transfer the ratings of node B to node E, there are two different routes with 3 hops: $r_1 = (B \rightarrow C \rightarrow D \rightarrow E)$ and $r_2 = (B \rightarrow F \rightarrow D \rightarrow E)$; and one route requiring 5 hops:

[12] Note that this type of normalization ensures z_{ui} is derived from a normal distribution $N(\mu'_u = 0, \sigma'_u = 1)$.

[13] Of course, CF recommendation requires also enough ratings from the own user to understand her preferences.

[14] To make data propagation in the picture more clear, we used unidirectional arrows, but of course data exchange is bidirectional.

$r_3 = (B \rightarrow F \rightarrow I \rightarrow J \rightarrow G \rightarrow E)$. In total, the ratings of node B are propagated over 9 hops[15], but with perfect knowledge of the network topology, only 3 hops would be necessary.

TTL controlled flooding: A simple approach to avoid the unmanageable flooding of MANETs with messages is the well-known *Time-to-live (TTL)* mechanism [8]. TTL defines the maximum number of hops that a message may be propagated in the network. For implementing TTL mechanism in MANETs, each message contains a TTL field, which serves as a counter. When a message is created, the TTL counter is initialized with the maximum allowed number of hops. Every time the message is sent, the counter is decremented. Incoming messages with $TTL = 0$ are not propagated further.

The TTL value controls the connectivity between nodes in the MANET. An optimal TTL value is difficult to choose and strongly depends on the application scenario. In the example of Fig. 5, the route r_3 is too long and not available, if $TTL = 3$ has been chosen. With a smaller $TTL = 2$ there is no route available to connect nodes B and E. In general, in densely populated open spaces, users will have no problem obtaining sufficient ratings with small TTLs, even through direct encounters. However, in sparsely populated areas, a larger TTL is required to enable connection between distant nodes over longer multi-hop routes. As a general strategy, the TTL value must be increased if a user does not have enough ratings available.

Similarity-based flooding: Another approach to avoid message overload is similarity-based flooding. Because users are mainly interested in the preferences of users who are similar to themselves, they maintain an overlay network connecting them only with users with the same preferences. Then, messages are only exchanged between similar users, so less messages are spread across the network and need to be processed at each node.

The similarity $\rho(u, v)$ between two users u and v can be estimated according to the Pearson correlation index as shown in eq. 5.

$$\rho(u, v) = \frac{\sum_{i \in I_u \cap I_v} z_{ui} \cdot z_{vi}}{\sqrt{\sum_{i \in I_u \cap I_v} z_{ui}^2} \sqrt{\sum_{i \in I_u \cap I_v} z_{vi}^2}} \tag{5}$$

where z_{ui}, z_{vi} are the ratings of item i given by user u and user v, respectively (according to equation 4). Note that equation 5 considers only items that have been rated by both users, i.e. item $i \in I_u \cap I_v$, where I_u is the set of exhibits visited by user u so far. The Pearson index calculates the degree of likeness between two users by comparing the ratings of those items that the two users have in common.

To establish a similarity-based overlay network, the node of a user u forwards messages from a user v only if similarity $\rho(u, v) > \psi$, i.e. from users whose similarity is above a certain threshold ψ. The threshold ψ controls the connectivity of the overlay network. If ψ has a value near to 1, only very similar users may exchange data. Again, an optimal value depends on the application scenario. If there are enough similar users in the facility, a high value can be selected. Otherwise, a smaller ψ provides users with enough ratings, but which may be less relevant, because they come from more diverse users.

3.2.2 Message format

For enabling collaborative filtering, a MANET node associated to user u has to send messages σ_u containing the following information:

$$\sigma_u = (id, t, TTL, uid, \mathbf{Z}^u) \tag{6}$$

First, some meta data is necessary to enable MANET routing including a message identifier *id* to avoid message duplicity, a timestamp t to allow discarding of old messages and a *TTL* field used as described above. The payload itself includes a user identifier *uid* indicating the user who triggered the message, and the rating matrix $\mathbf{Z}^u = [z_{vi}]$, which contains all ratings known to user u. A matrix entry z_{vi} is the rating given to item i by user v, which is calculated according to equation 4.

3.2.3 Triggering rating dissemination

There are different policies to decide when a should propagate her ratings. Generally, the dissemination should be triggered when a new situation has occurred. The following trigger events can be distinguished:

- Elapsed Time Window: the device of user u periodically sends a rating message σ_u with a certain frequency ω. Technically, this approach checks at fixed time intervals, whether the user is within range of yet unknown users. This way, a user detects other passing-by users.
- New Rating Available: a dissemination of the rating messages σ_u takes place, if the user has received new ratings from other users, or if she has finished her visit at an item and can calculate a new rating of her own. Here, new rating information is propagated without any latency.
- New Area Detected: as soon as a user has entered a new area of the facility, she searches for users within range who are still unknown to her in order to exchange ratings.

[15] If we assume that the same message is not exchanged several times between the same nodes, e.g. the exchange $B \rightarrow F$ is not done for each of the three routes.

Note that each device decides when sending information to others. Therefore, there is no synchronicity among users to exchange their rating data.

3.3 Stage 3: Data fusion

In a data fusion step, ratings received from other users are combined with the information already stored in the device. This means, that the rating matrix $\mathbf{Z}^u = [z_{vi}]$, which contains all the ratings the device owner u knows so far, must be merged with matrices $\mathbf{Z}^v$ contained in the data messages received from other users v.

The dimension of matrix $\mathbf{Z}^u$ is $|U| \times |I|$ with the number of items $|I|$ and the number of users $|U|$. If the total numbers of users and items are unknown in advance, the matrix dimension can grow over time. In most recommendation scenarios $|U| \gg |I|$ holds, because the number of users grows continuously while the number of items remains more or less the same. Furthermore, the matrix $\mathbf{Z}^u$ is usually *sparse*, i.e. it contains many zero entries. This corresponds to the fact that users rate only a small fraction of the available items.[16]

If a device owned by user u receives a new rating matrix $\mathbf{Z}^v$ from another user v just encountered, the following steps are executed:

1. For implementing similarity-based flooding, matrix $\mathbf{Z}^v$ is used to calculate similarity $\rho(u, v)$ due to Pearson correlation as described by eq. 5. Only ratings from users with a similarity beyond a certain threshold ψ are accepted and integrated in $\mathbf{Z}^u$.[17]
2. If user v is similar to user u, each entry z_{wi} of the matrix $\mathbf{Z}^v$ is inserted into matrix $\mathbf{Z}^u$. It can happen that there is already an entry at position (w, i) in $\mathbf{Z}^u$, namely when the rating z_{wi} has already arrived over another MANET route. But because we assume that each user gives a rating only once, inconsistencies cannot occur.

In public spaces, older data often has no relevance, since only recently submitted ratings describe the current situation. In a zoo, for example, visitors are interested only in the current situation at an enclosure and not what it looked like a few hours ago. Implementing such a mechanism requires knowing at what time a rating was issued. Then in a cleaning step, all ratings older than an application-specific threshold can be removed from the matrix $\mathbf{Z}^u$.

3.4 Stage 4: Calculating recommendations

There are various methods for calculating CF-based recommendations. The most prominent ones are user-base collaborative filtering (UBCF) and item-based collaborative filtering (IBCF). Because IBCF has some advantage in recommender scenarios with much more users than items [22], here we will consider IBCF.[18] The calculation of IBCF recommendations works as follows [32]:

1. First, *similarities between items* are calculated by using z_{vi}-ratings in $\mathbf{Z}^u$. Among several possibilities, the similarity $w_{i,i'}$ between to items i and i' can be computed by the Adjusted Cosine function:

$$w_{i,i'} = \frac{\sum_{v \in U_i \cap U_{i'}} z_{vi} \cdot z_{vi'}}{\sqrt{\sum_{v \in U_i \cap U_{i'}} z_{vi}^2} \sqrt{\sum_{v \in U_i \cap U_{i'}} z_{vi'}^2}} \tag{7}$$

in which $v \in U_i \cap U_{i'}$ represents a user v that visited both items, where U_i is the set of users that have already visited item i.
2. To calculate the prediction $\hat{z}_{u,i'}$ of the rating a user u gives to a yet unvisited item i Eq. 8 is used.

$$\hat{z}_{u,i'} = \frac{\sum_{i \in I_u} w(i, i') \cdot z_{ui}}{\sum_{i \in I_u} w(i, i')} \tag{8}$$

where I_u denotes the set of exhibits the user u has already visited. Finally, we would calculate the predicted sojourn time $\hat{s}_{ui'}$ by rearranging eq. 4.

$$\hat{s}_{ui'} = \hat{z}_{ui'} \cdot \sigma_u + \mu_u \tag{9}$$

One of the main common problems when dealing with collaborative filtering approaches is the so-called cold-start problem; that is, users need some time to receive accurate recommendations, due to the lack of own personal experiences. Since the recommender mechanism in our approach works with the user's partial view of the environment (built from information exchanged), it also needs some more time to use collected information to be combined with own experience. This means that in our MANET-based approach, the dissemination of rating data might be slower than in a centralized architecture.

Depending on how many people are in the open space, the cold start phase may therefore take longer until enough

[16] For the sake of brevity, we do not consider optimized data structures for the sparse matrices $\mathbf{Z}^u$.

[17] Or at least the data of user v should not be forwarded to other users.

[18] IBCF solves some issues related to: (i) the scalability of the system (systems containing more users than items) and (ii) efficiency (only calculates similarities between items instead of between users such as in user-based approaches). Furthermore, it tends to be more stable, since changes in user profiles are supposed to occur more frequently than in item profiles.

ratings have arrived at a user. Although beyond the scope of this paper, there is a large body of work in the literature that attempts to mitigate this problem in CF approaches[9, 14, 15], any of which could be used for our distributed approach.

3.5 Privacy issues

Recommender systems based on collaborative filtering always face privacy issues, as they require users to share their personal ratings with others. Although the aspects to take into account depend to a great extent on the domain (e.g. there is much work done in the health domain [36]), in order to evaluate privacy, we have to consider (i) the sensitivity of data that a user is required to disclose and (ii) who is granted access to the data. Data sensitivity In our system, the data disclosed by users is of low sensitivity. First, the information exchanged is always anonymous, since the messages contain only an artificial user id that could be created by a UUID generator. This ensures anonymity, since user ids do not link to any information about the real users.

Furthermore as depicted in eq. 6, a message contains a Z^u user-item matrix with the normalized ratings that are known to the sender so far, accompanied by some meta data such as message id, timestamp, and TTL value. It is important to remark that location-based services may also be sensitive to tracking issues for users [11, 39]. Note that in our system, there are no timestamps for rating informations, so users' routes cannot be tracked. Overall, the disclosed data cannot be used to derive any unique characteristics of the user such as name, address, or personal preferences. It can be assumed that it does not cause serious harm if the data is lost. Considering the evaluation scheme in [33], we claim that data sensitivity of our system belongs to category III 'Low Sensitivity', i.e. loss of such data would not affect the person. Data access In our MANET-based system, users do not have to submit their data to a central authority that might be interested and capable to exploit the data for commercial purposes. This also means that there is no instance in the system that owns all rating data. Instead, users share their data only with people they encounter by coincidence and who are unlikely to have the capability to access and misuse the data.

Furthermore, users have the control over the information they exchange. They can decide what information they share with others, e.g. the could reveal only their *k* favorite items to others. In particular, the system could allow users to act only as intermediaries, meaning that they share only other users' ratings, but not their own. Nevertheless even such a user can calculate recommendations because the CF algorithms run on her private smartphone, which is the only one using her own ratings. This is an advantage over a centralized architecture where users are forced to reveal their personal ratings in order to receive a recommendation[19]. Because no private, in particular no rating data is shown to the user, we claim our system provides data visibility of category 'Low' according to [33].

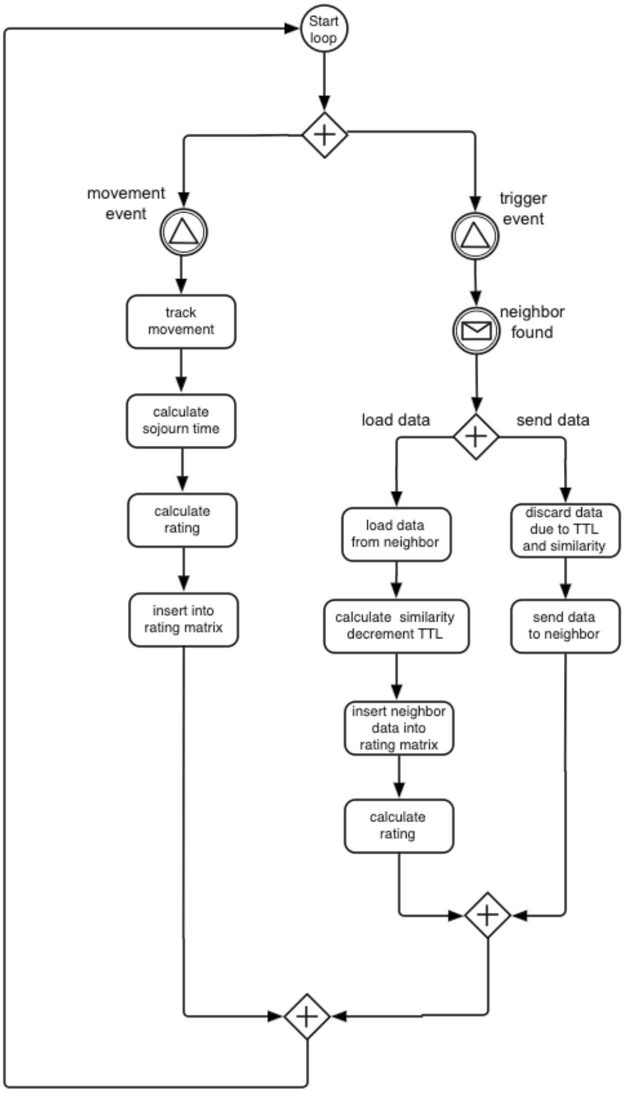


Fig. 6 Overview of the routing process

In summary, our MANET-based recommender addresses privacy by using anonymous data of low sensitivity. Furthermore, there is no central authority that has access to all data; instead a user can control for herself what data she wants to reveal.

[19] Of course, this only works if enough users provide their rating data generously.

3.6 Summary – Routing protocol

Figure 6 gives an overview over the entire MANET recommendation process (using the BPMN notation[20]) and shows how the different parts fit together.

Each smartphone is executing three different subprocesses concurrently in an endless loop:

- *Calculating implicit ratings:* a movement event, such as a GPS position change or a new beacon reading, triggers the calculation of ratings. According to subsection 3.1 this requires tracking the movements (e.g. assigning positions to items), calculating the sojourn time in front of an item and normalizing it to derive the rating. Finally, the new derived rating is inserted into the rating matrix $\mathbf{Z}^u$.

The other two subprocesses are started if a trigger event (according to subsection 3.2.3) has occurred and afterwards a neighbor in range has been found. After connecting with an encountered user v, the message containing her rating data matrix $\mathbf{Z}^v$ is loaded. Afterwards, the similarity with user v is calculated, as well as the the TTL counter is decremented (according to subsection 3.2.1). Now, concurrently, the next two steps make use of the new rating data:

- *Calculate recommendations:* The data in $\mathbf{Z}^v$ is inserted into the own rating matrix $\mathbf{Z}^u$ (according to section 3.3). In the subsequent step, $\mathbf{Z}^u$ is used to calculate ratings based on IBCF as discussed in section 3.4). These ratings will be used to hopefully make appropriate recommendations to the owner of the device.
- *Sending valid data to encountered neighbors:* the user's data stored in $\mathbf{Z}^u$ can be sent to neighbor users in range. Before sending it is checked that the data still has a $TTL > 0$ and eventually if the neighbor user is similar. Otherwise, the data may be discarded.

4 Experiments

In order to exhaustively test our approach we conducted an extensive set of simulation experiments. The resulting understanding is the basis for the practical operation of the system.

4.1 Application scenario

We have embraced a public zoo as an application scenario. We assume zoo visitors are equipped with any sort of mobile devices (e.g. an smartphone or a tablet) running the recommender system we have presented above, and are strolling around and linger in front of open-air enclosures with different animals.

The recommendation app monitors the precise trajectory of its user by using a beacon infrastructure as already explained in section 3 in more detail[21]. The user trajectory, in particular the sojourn time spent by the visitor around the enclosures, is used to derive implicit ratings in the owner device.

The zoo scenario also illustrates an advantages of a MANET approach: because multi-hop communication in MANETs is restricted by the TTL threshold, the ratings a user receives tend to come from surrounding areas, because they arrived in limited number of hops. Thus it is more likely that the recommended enclosures are within reach.

4.2 Simulation

The simulator was implemented using the Repast Simphony framework[22]. To evaluate our approach, we conducted various simulation experiments based on the following set-up.

4.2.1 System Model

Repast allows modeling the ground plan of the zoo facility as a 2-D grid of dimensions $n \times m$. Each grid cell may be occupied by any number of visitors at the same time. Besides the spatial constraints, the simulation setup allows to tune some other important parameters:

- *Enclosures*: some of the grid cells may contain enclosures. We assume a number of n_e enclosures, randomly distributed in the grid. For categorizing enclosures, we assume that each enclosure k corresponds with one preference category $p_k \in P$.
- *Preference Categories* are defined by a set $P = \{p_1, p_2, ..., p_n\}$. In the zoo scenario, preferences correspond with generic animal categories. For instance this might be $P = \{birds, fishes, reptiles, predators, ...\}$.
- *Preference Profiles* P_u of a user u is an ordered set representing preference categories in which u is interested. For instance, the preference profile $P_u = \{predators, reptiles, birds, ...\}$ determines the order $predators \succ_i reptiles \succ_i birds \succ_i ...$ where $\succ_i$ stands for an preference operator, i.e. the user is more interested in predators than in reptiles.

[20] https://www.bpmn.org

[21] As this scenarios refers to an open-air facility , we suggest beacon technology might be combined with GPS, so yielding more precise positions. Nevertheless, comparing different location technologies is out of the scope of this paper.

[22] https://repast.github.io

4.2.2 Visitor behavior

In our model, zoo visitors behave as follows:

- *User Life Cycle*: when the simulation starts, a number of visitors is created, entering the facility through the main gate. We assume there is only one entrance gate. For understanding the impact of different visitor occupancy on the quality of our approach, we define a capacity C of visitors, i.e. we assume a constant number of visitors in the zoo. After the capacity is reached in a transient starting phase, a new visitor can only enter when another one has left. Visitor behavior consists of moving around and staying in front of some enclosures observing animals, before she leaves the facility (allowing another visitor to enter).
- *User Movement*: users move around according to two different strategies:
    - *Stochastic path*: each visitor visits enclosures following a random path. When leaving an enclosure a new random suggestion is derived for the visitor.
    - *Recommended path*: visitors follow an IBCF approach to obtain the next suggested enclosure to visit. In the cold start phase, when visitors have just entered the facility and not much information is yet available to make accurate recommendations, a random recommendation is made.
- *Rating exchange*: after each simulator time tick, visitors exchange their rating data according to the rules already described in section 3, in particular considering the TTLs. Data is exchanged between those visitors that are in range, i.e. they are in the same cell or in neighbor cells.

4.2.3 Time behavior

For calculating the sojourn time of a user in a certain grid cell we have to distinguish two different situations: i) whether a cell is empty or if it contains an enclosure the user is not interested in (not suggested by any of both strategies for user movement above), she is staying only a short time in the cell representing the time for passing to the next cell; ii) however, if the cell contains an enclosure the visitor is interested in, then a specific sojourn time is drawn using the following mechanism:

- A *behavior profile B* is assigned to each user to distinguish visitors with different behaviors. This reflects the fact that some visitors tend to stay longer in front of enclosures than others (even if they all liked them). A behavior profile specifies the mean sojourn time a visitor spends at an enclosure belonging to a certain category of her preference profile. It is formalized as an ordered set $B_p = \{b_1, b_2, b_n\}$ with b_j representing the average sojourn time the visitor spends at an enclose belonging to the category of preference p_j. For instance, $B_{slow} = \{16, 8, 4, 2, ...\}$ means that on average, a user with this profile spends 16 minutes at enclosures of her favorite category of preference, 8 minutes at her next-favorite category, and so on. A behavior profile for a fast user B_{fast} just defines smaller average sojourn times such as $B_{fast} = \{8, 4, 2, 1, ...\}$ meaning that compared with slow users, visitors stay only half the time at the enclosures. This could also allow configure simulations with visitors with different capabilities, e.g. visitors with physical disabilities, different age-ranged visitors, etc. For the evaluation three different behavior profiles for visitors have been defined: a fast, a slow, and a medium behavior (in between the average paces of the other two).
- *Sojourn times*: the sojourn time s_{ue} of user u at enclosure e is drawn using a Gaussian distribution. If an enclosure has the preference category that corresponds with position i in user's u preference profile, then the sojourn time is drawn by using i-th entry in the user's behavior profile as mean value. For instance, user u with the P_u preference profile described above and the B_{slow} behavior profile waits on average 16 minutes in front of predators, 8 minutes in front of reptiles, and so on. We introduce some variability with the standard deviation that defines the Gaussian distribution. As we will explain below, this is an input parameter for the simulation setup.

4.3 Performance metrics

We use two different metrics to measure performance in each simulation scenario. On the one hand, we calculate the *predicted time error*, i.e. the difference between the predicted sojourn time the visitor stands in front of an enclosure and the actual sojourn time she actually spent (see eq. 10).

$$\xi_{ue} = |\hat{s}_{ue} - s_{ue}| \tag{10}$$

On the other hand, we use the *Root Mean Squared Error* (RMSE) as a metric calculated as follows:

$$RMSE = \frac{\sqrt{\frac{1}{|S_{test}|} \cdot \sum_{s_{ue} \in S_{test}} (\hat{s}_{ue} - s_{ue})^2}}{(s_{max} - s_{min})} \tag{11}$$

considering the square root of the mean square difference between measured sojourn times $s_{u,e}$ and the predicted $\hat{s}_{u,e}$ sojourn times. Note that $|S_{test}|$ is the number of values in the test data set and that the RMSE value is normalized by dividing it by the difference between the maximum and minimum sojourn time in the matrix ($s_{max} - s_{min}$).

Table 1 Experimental setup

Maximum number of visitors	50, 100, 500, 1000, 10000
Number of enclosures	50, 100, 200, 500
Visitor capacity	25, 50, 100
Visits before exit	30%, 50%, 70%
Std. deviation	30%, 50%, 70%
TTL	1, 3, 5, 7, 9, 15

4.4 Experimental setup and results

We conducted a large set of experiments to evaluate our approach. As different features may influence the behavior of users in the system, we have studied how each of them affects the visitors' performance over time. Following we present the different input parameters for the simulations:

Facility dimensions It defines the grid dimensions for the simulation facility ground. Each cell may accept more than one visitor. For the sake of simplicity we only consider squared grids set to 50x50 in every scenario. We have carried out experiments changing this input parameter. However, we have noticed this does not entail significant performance variability according to the obtained outcomes.

Maximum number of visitors It represents the number of different visitors that enter the facility during the simulations. If this parameter is set to n, then when the n-th visitor enters the facility the simulation ends.

Number of enclosures It represents the number of enclosures exhibited in the zoo. Enclosures are randomly placed in the grid at the beginning of each experiment. Two different enclosures cannot occupy the same grid cell. Again, for the sake of simplicity we only consider scenarios with a fixed number of 50 enclosures.

Visitor capacity It defines the constant number of visitors in the facility at the same time. When a visitor leaves a new one enters the facility.

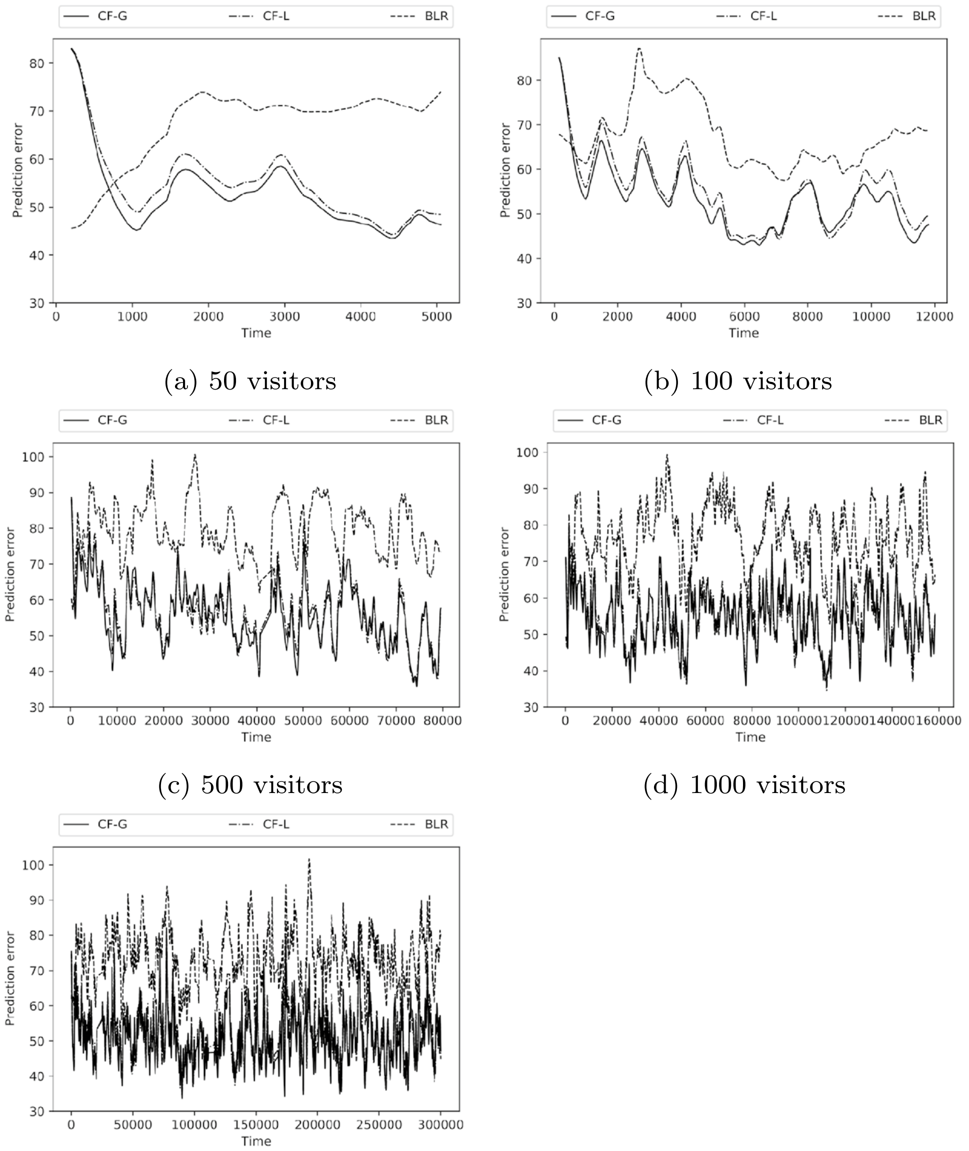
Visits before exit This parameter controls the visitor life cycle in the facility, since it defines the percentage of enclosures (out of the total number of existing enclosures in the zoo) that must be visited before leaving the facility. This parameter allows tuning the time visitors spend in the zoo. Once this value is reached by a visitor, she is automatically forced to leave the simulation.

Standard deviation This allows representing stochastic variations in the visitor behavior in order to draw sojourn times (from their behavior profile according to her preferences using the Gaussian distribution). This value is

Table 2 Avg. prediction time error and avg. RMSE

		Prediction error			RMSE		
		CF-G	CF-L	BLR	CF-G	CF-L	BLR
# visitors	**50**	52.951	55.135	66.943	0.645	0.673	0.913
	100	53.378	55.246	67.665	0.658	0.680	0.915
	500	55.761	56.508	79.488	0.695	0.702	1.107
	1000	54.246	54.875	77.001	0.675	0.681	1.081
	10000	50.705	51.999	73.852	0.635	0.650	1.039
Facility capacity	**25**	46.185	48.594	68.737	0.777	0.817	1.261
	50	52.396	54.524	70.467	0.770	0.859	1.214
	100	47.769	52.223	56.861	0.582	0.634	0.710
Visits before exit	**30%**	71.260	78.829	81.496	0.720	0.805	0.908
	50%	57.408	58.761	77.087	0.711	0.728	1.070
	70%	50.567	51.571	72.721	0.714	0.728	1.151
Std. deviation	**30%**	60.255	60.501	77.649	0.821	0.895	1.403
	50%	55.709	56.978	76.886	0.910	0.930	1.391
	70%	51.574	53.410	74.596	0.875	0.903	1.366
TTL	**1**	53.553	54.411	71.708	0.806	0.873	1.271
	3	52.730	54.286	77.923	0.876	0.900	1.409
	5	52.761	53.008	76.949	0.873	0.875	1.384
	7	50.246	51.028	70.357	0.840	0.851	1.288
	9	49.605	50.040	73.029	0.817	0.823	1.323
	15	50.852	51.792	73.894	0.838	0.852	1.337

Fig. 7 Results for different total number of visitors (Prediction error)



(a) 50 visitors

(b) 100 visitors

(c) 500 visitors

(d) 1000 visitors

(e) 10000 visitors

represented as a percentage with regards to the mean value.

TTL It represents the number of hops a message takes among the visitors. This limitation prevents from message flooding.

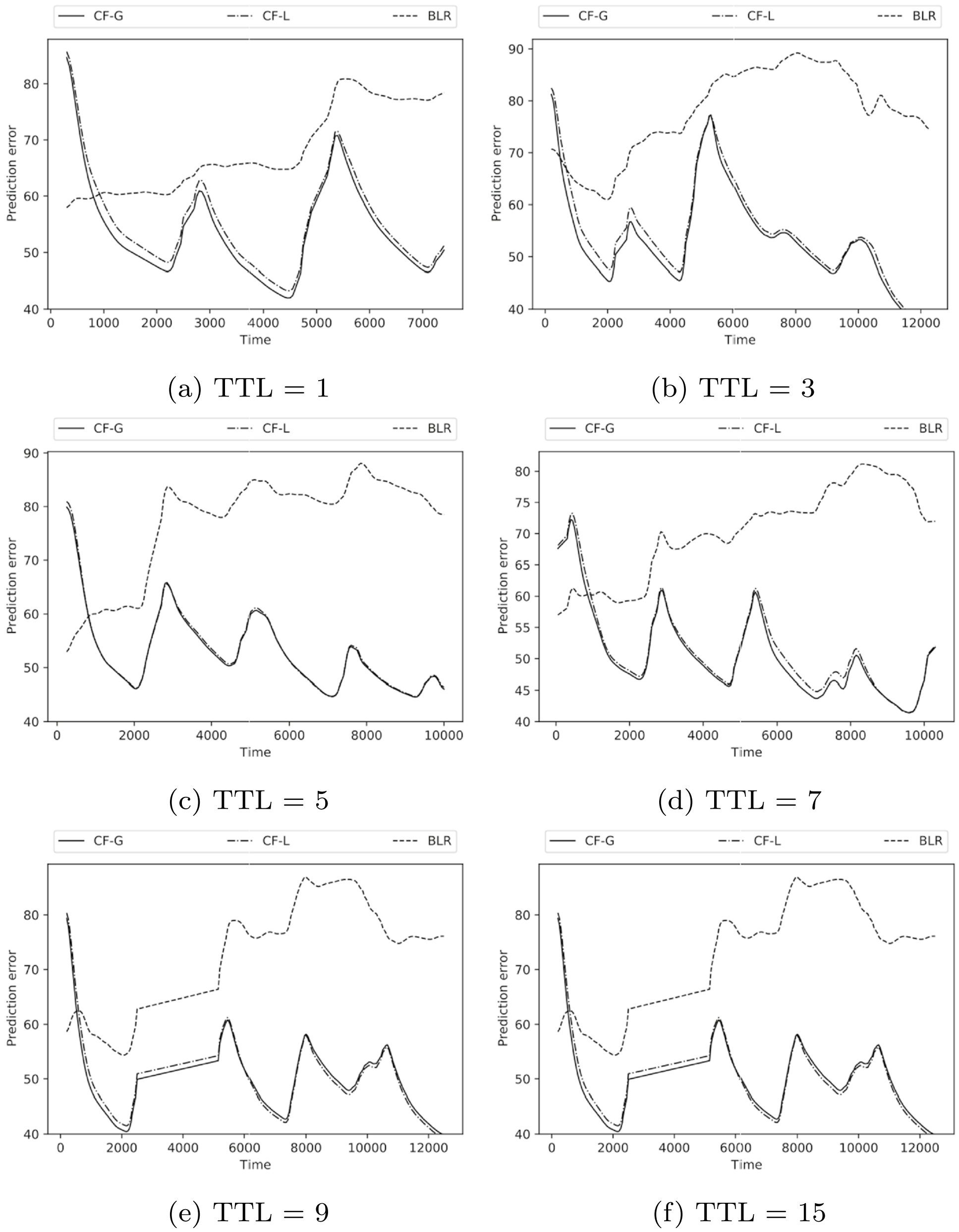
The effect of these parameters is exhaustively tested with a large set of different experimental setups, trying different input parameter values as shown in Table 1.

In these experiments we compare three different approaches to suggest visitors the next enclosure to visit:

- CF-G: a traditional IBCF approach with kNN selection, using a global user-item matrix containing all sojourn times of the current visitors in the facility. This approach represents a server-based architecture.
- CF-L: each visitor uses its own user-item normalized sojourn time matrix (**Z**), combining her own visit experiences along with information exchanged with other visitors during their stay. The recommendation technique remains the same for every visitor (IBCF with kNN selection). This approach represents our MANET-based architecture.
- BLR: a baseline method, merely a random recommendation among the set of enclosures for any visitor.

Note that in any of the above recommendation methods once a visitor is recommended an enclosure, this cannot be suggested ever again.

As this experimental section is focused on the recommendation performance for visitors with a distributed approach



(a) TTL = 1

(b) TTL = 3

(c) TTL = 5

(d) TTL = 7

(e) TTL = 9

(f) TTL = 15

Fig. 8 Results for TTL parameter (Prediction error)

in the proposed scenario, we only use one triggering rating dissemination policy. Visitor devices adhere to a ’elapsed time window’ policy set to 1 to broadcast their local information, i.e. each visitor is able to receive and broadcast local information at every time tick. In the same line, messages are broadcast and received by any visitor in range. Since in this scenario we have not identified a message overload, testing other policies to reduce message flooding, such as a similarity-based flooding (as explained in Section 3.2.1) are not necessary.

Table 2 summarizes results for the different experimental setups as well as different recommendation methods. Average prediction errors and RMSE are presented. As expected, CF-G and CF-L outperform BRL method. The CF-G approach can be seen as an optimal method, since the collaborative filtering is calculated with the information of all current visitors in the system, CF-L achieves close-to-optimal results, which proves a MANET approach is effective in this type of scenarios.

As it can be observed, the number of visitors does not entail significant differences on the metrics observed. Regarding TTL, results confirm that an increase in the number of hops a message can be forwarded allows to reach more visitors, so enhancing the recommendation accuracy. However, we identify that there is an upper limit in which a rise of TTL does not entail a better performance in terms of prediction time error.

For a more visual understanding of the results, Figure 7 shows how the variation in the total number of visitors affects the visitors' performance over time. As can be seen, CF-L curve follows the same trend as the CF-G curve and in turn clearly outperforms the BLR curve. In fact, CF-L and CF-G overlap most of the time, so assuring that information exchange of local information may offer an equivalent solution to recommendation problems, overcoming some important problems of centralised systems though, as stated in the Introduction.

Regarding how TTL parameter affects visitor performance over time, Figure 8 shows the curves for the different experiments using the three proposed recommendation methods, in terms of predicted time error. Note that peaks in the curves are due to the openness of the scenario, in which visitors leave the facility and new ones occupy their places. Newcomers need a period to get accurate recommendations (classic cold-start problems). Although not all visitors stay the same time in the system, all of them are constrained to visit the same percentage of enclosures before leaving, that is the reason why some periods are more likely to contain a significant number of leavings and enterings.

For a deeper look into the rest of experiments, all of them are displayed in Appendix A.

5 Related work

There is not much work in the literature about using MANET as a way of implementing distributed recommender systems. Instead, there is some work that uses recommender systems to control information exchange in MANETs, trying to prevent nodes from exchanging messages with misbehaving counterparts [34, 35].

Nevertheless, in recent times some efforts have been made to use MANET techniques to enable information exchange in distributed recommender systems. For instance, in [7] authors use a MANET approach to exchange ratings among visitors in a museum. In that, users exchange the individual rating right after visiting an artwork and suggestions provided by the recommender system contain a trajectory, taking into account some other contextual factors. In our case, the complete sojourn times matrix is exchanged between visitors, what helps visitor's mobile device to have a more complete information to issue recommendations in less time. This is particularly important in open facilities in which large distances between visitors make it more unlikely to ran into each other to exchange data.

Although the idea might be considered quite similar, in that approach visitors exchange explicit ratings and calculate recommended trajectories, instead of working with implicit recommendations and single suggestions.

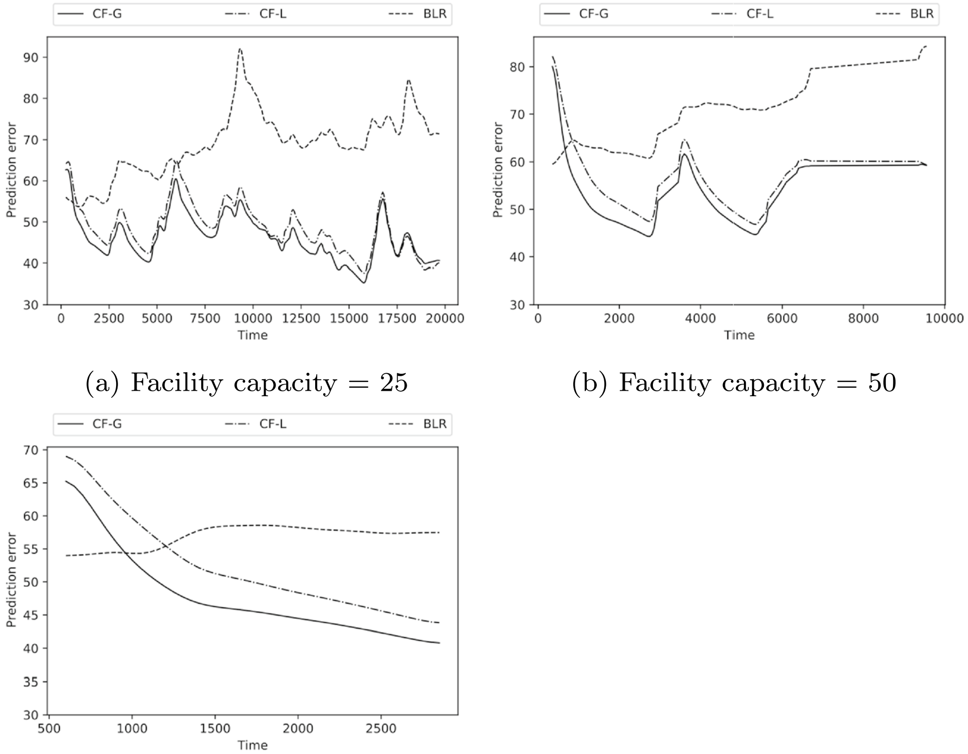
Other works focus on peer-to-peer approaches for recommendation environments. As an example, in [37] authors put forwards various data exchange methods among users for the tourism domain. Again, information exchanged in messages is related to explicit ratings users store. In the same line [13] presents a P2P approach based on gossip mechanisms to share the information in a pervasive environment. However, in an open environment, in which the network topology frequently changes, we believe a MANET approach is more suitable.

Techniques for extracting implicit feedback in recommendation environments have been researched during last decades [20]. As we do, most of these works attempt to infer user feedback to incorporate the knowledge in collaborative filtering models [18]. For instance, in [27] a system for recommending electronic books is presented. Authors use the user's interaction with the system to infer an implicit rating that is then fed into the recommendation process. In our case, we do not work on an online environment but on a physical one, so an infrastructure (e.g. a beacon infrastructure) is needed to calculate the sojourn times. The work presented in [10] goes a step beyond since it attempts to combine both implicit and explicit feedback in order to provide more accurate recommendations by using matrix factorization techniques. We consider this type of approaches unfeasible in some domains: in scenarios in which the number of items/exhibits to be visited is high it seems unrealistic to bother the users to rate every thing they visited. Other approaches also try to address the use of implicit feedback by using Bayesian techniques in order to approximate the user behavior [16, 29]. In our case, we focus on a pervasive environment where visitors have their own mobile device that can be used to monitor the times visitors spend close to an item.

Movement patterns in open spaces are also studied in [11]. The authors use location data from cell phones to derive a periodic and social mobility model. Looking at movements of individuals over a larger period of time, typical patterns can be detected, such as a particular person going to her workplace every weekday morning at 8:00, or having a network of close friends that she visits regularly. Instead in our work, we consider open facilities that may be visited only once. Therefore, it is not possible to find regular repeating patterns of movements: people's movements depend on their preferences and are strongly influenced by current situation in the facility.

In [39], a recommender system for Points of Interests (POI) based on sentimental and spatial characteristics of locations is presented. Using sentiment analysis of microblog posts, sentimental features are inferred for each POI. In combination with spatial features such as geographical location, individual recommendations can be provided. Unlike our system, this approach relies on much more input from users, here in form of microblog posts to derive their sentiments about a POI. Our system has no further information

Fig. 9 Results for different facility capacities (Prediction error)



(a) Facility capacity = 25

(b) Facility capacity = 50

(c) Facility capacity = 100

about users other than their sojourn times. Therefore, this approach cannot be easily applied to our system.

An important issue to be considered in MANET is privacy. We assume in this paper that users may share their local information with other users in the facility. However, this assumption could become more complex, since information disclosure is user dependent and law-related issues may emerge. An example of work in this line can be found in [28]. More specific work in the literature uses randomized dissemination protocols [6] to achieve privacy or consider privacy in large-scale social networks [38]. In [25], the authors propose a trusted service for handling users' information, where users can configure the data sharing policy.

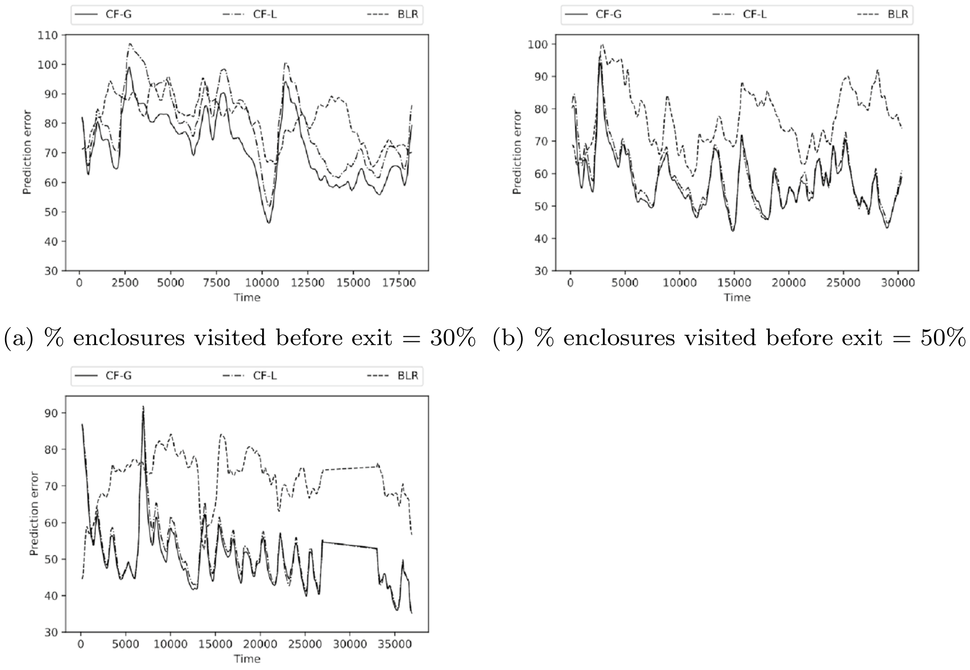
Fake ratings are another important concern when dealing with CF-based recommender systems. The effects and mechanisms to keep users from tampering with ratings to make users miscalculate predictions on unknown items has been studied in the literature [3, 30]. One of the key problems of introducing malicious ratings into a CF system is how fast (and how far) those ratings spread among the users. However, unlike a centralized approach, our MANET-based architecture prevents spoofed ratings from spreading quickly through the system. First, introducing malicious or fake ratings into our approach is not straightforward. User ratings are based on sojourn times and inferred by the recommender system running in the user's mobile device. They cannot not simply be entered directly by the user.

Nevertheless, even if malicious rating injection attacks are still possible, they would have a less harmful effect than in a centralized approach. In centralized systems, there is a unique rating data pool, which is used for the calculation of all subsequent ratings predictions. From the exact moment a user injects a malicious rating into the system, this is used for the collaborative rating calculations for all users of the recommender system, since they all use the same user-item rating matrix.

Since users in our approach only share the information they have so far (a partial view of the user-rating item matrix), the propagation of a malicious rating should take longer – on average – than in a centralized approach. Moreover, some malicious ratings might never reach some users, depending on the route users take in the facility (due to the propagation control used by means of the TTL field in messages).

Most of the work on distributed recommender systems has been focused on the domain of tourism due to several factors: the amount of available data sets, high economic impact, and an easily understood domain, just to mention a few. In [21] authors explore the future avenues on the use of information technology in the field of tourism. One of the main applications is on the availability of user-oriented applications with embedded recommender systems to provide with accurate suggestions to users. For a complete review of the use of recommender systems for tourism, please see [31].

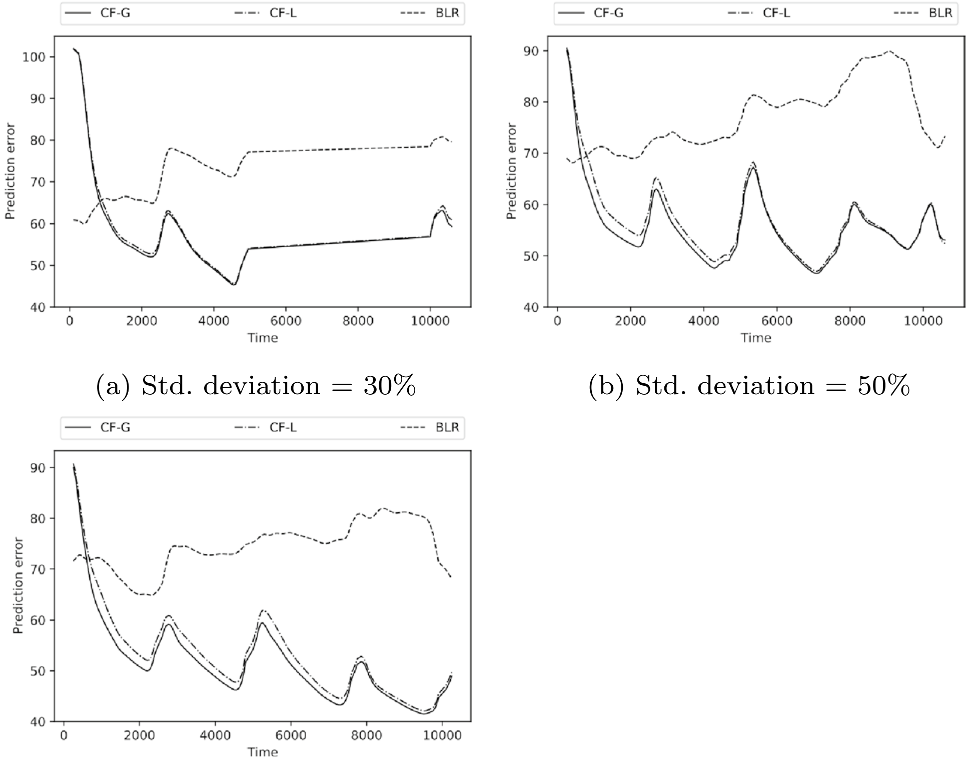
Fig. 10 Results for visits before exit parameter (Prediction error)



(a) % enclosures visited before exit = 30% (b) % enclosures visited before exit = 50%

(c) % enclosures visited before exit = 70%

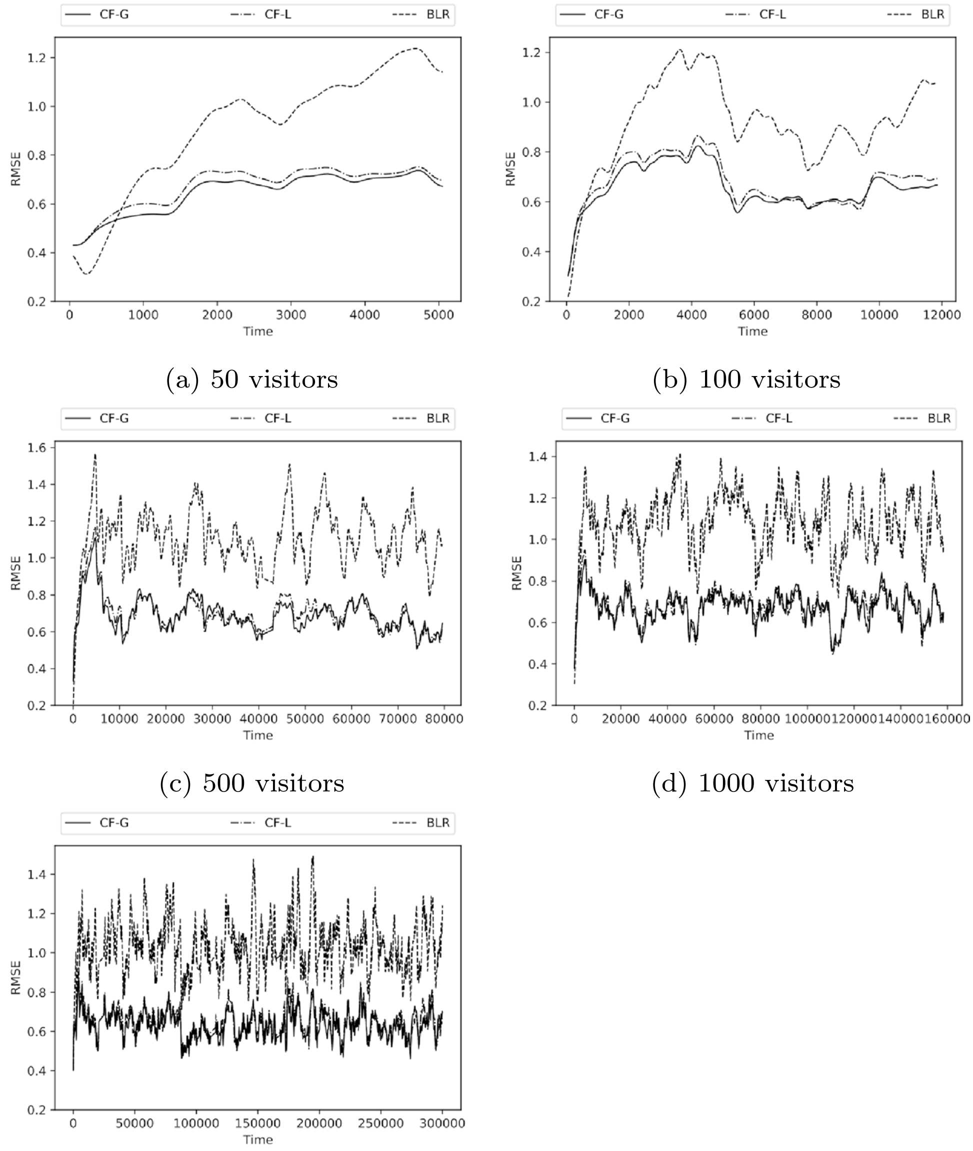
Fig. 11 Results for std. deviation parameter (Prediction error)



(a) Std. deviation = 30% (b) Std. deviation = 50%

(c) Std. deviation = 70%

Fig. 12 Results for different total number of visitors (RMSE)



(a) 50 visitors

(b) 100 visitors

(c) 500 visitors

(d) 1000 visitors

(e) 10000 visitors

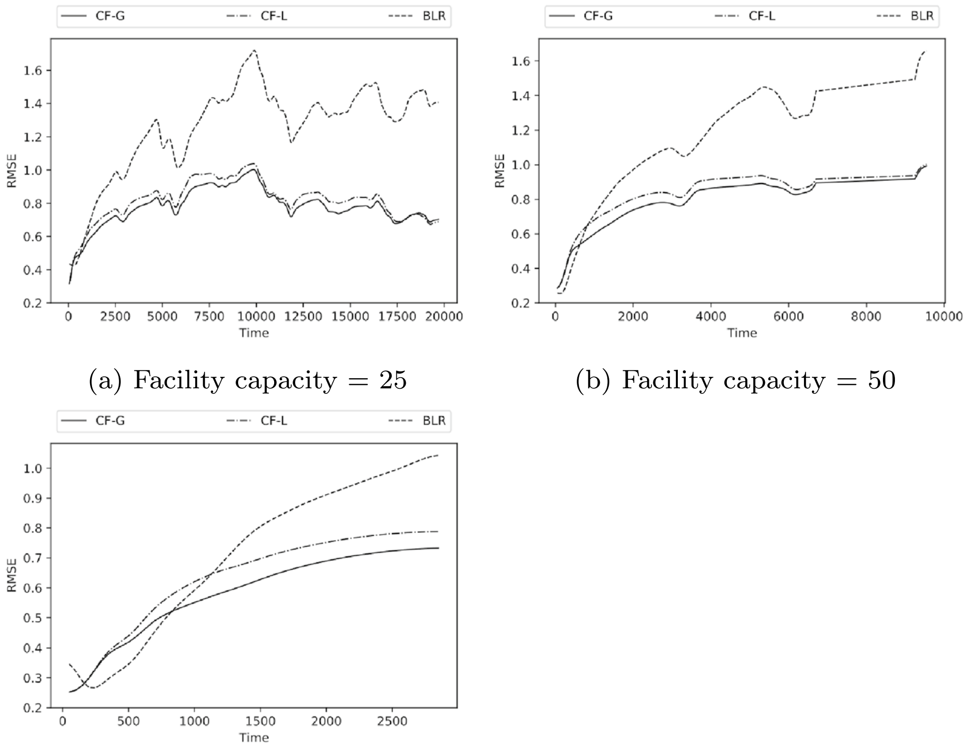
6 Conclusions

In this paper, we present a novel recommendation approach based on a decentralized and distributed MANET model for rating dissemination in open facilities. Visitors of open facilities can use their mobile devices to calculate recommendations through collaborative filtering in combination with information from other visitors they have encountered. An exhaustive set of experiments have been carried out, confirming that this type of techniques, besides alleviating classic problems of centralized systems, such as dependability, trust concerns, privacy issues or solution costs, also reaches close-to-optimal performance, in terms of predicted time error and RMSE metrics.

As future avenues of research we plan to test the MANET-based approach using other techniques for message-forwarding, e.g. similarity-based flooding to control information exchange among similar visitors. Moreover, the presented approach could be used to calculate recommended trajectories instead of single suggestions. This might be of particular interest when there exist time constraints for users. Another interesting step would be to look at other recommendation techniques, such as matrix factorization, to examine the extent to which MANET-based recommender systems depend on the recommender method used underneath.

Furthermore, other domains are also planned to be explored, particularly indoor facilities where user location must be more accurate.

Fig. 13 Results for different facility capacities (RMSE)

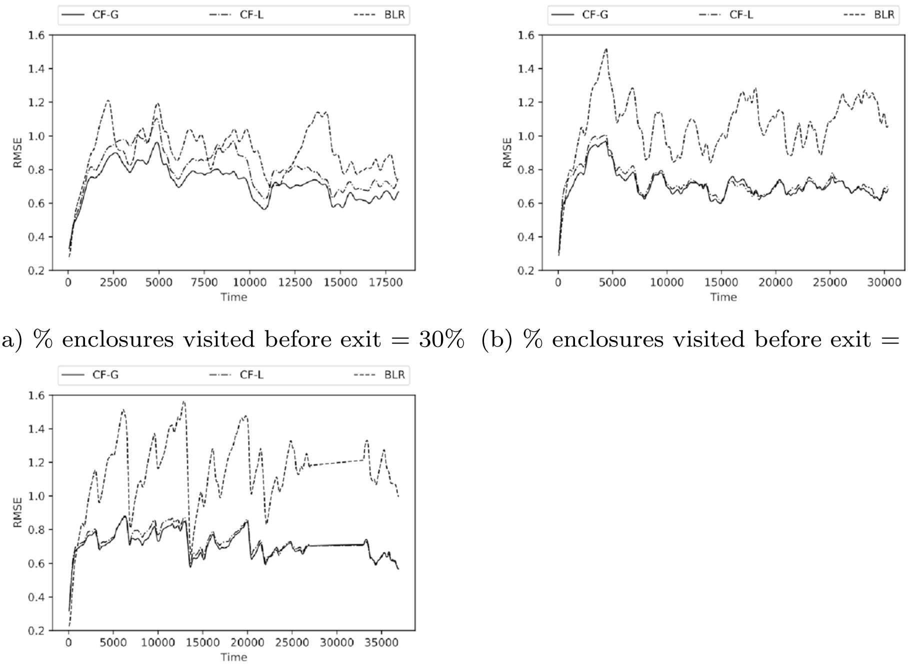


(a) Facility capacity = 25

(b) Facility capacity = 50

(c) Facility capacity = 100

Fig. 14 Results for visits before exit parameter (RMSE)



(a) % enclosures visited before exit = 30%

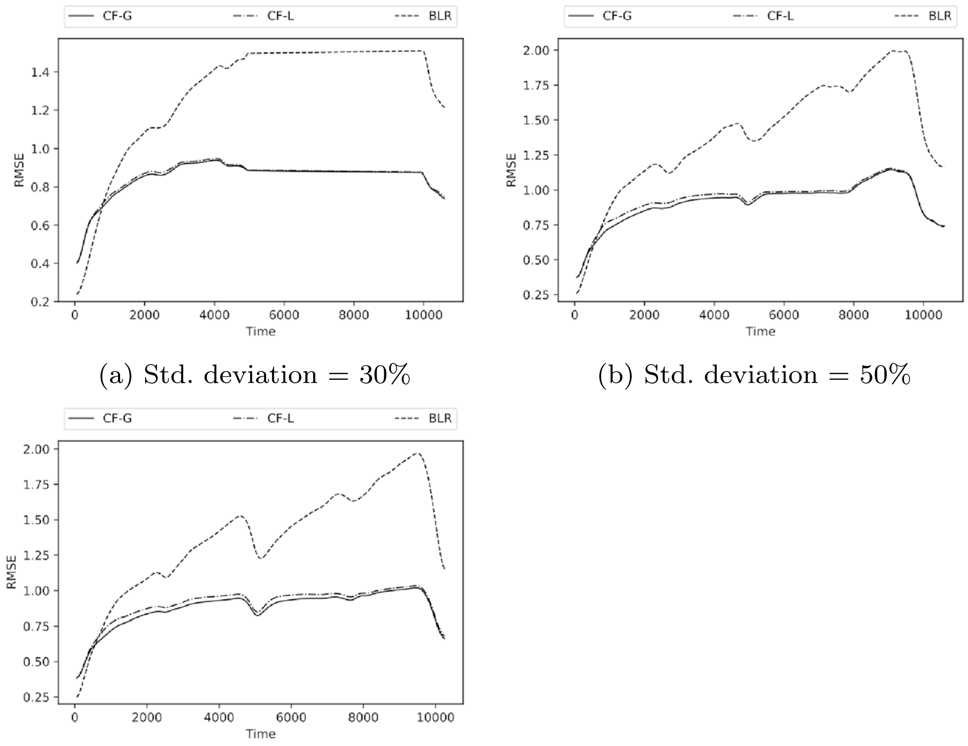
(b) % enclosures visited before exit = 50%

(c) % enclosures visited before exit = 70%

Appendix A: Other results

This appendix contains graphical plots for every experiment carried out for the empirical evaluation of the approach, except those already presented in Section 4 (Figure 9, 10, 11, 12, 13, 14, 15 and 16).

Fig. 15 Results for std. deviation parameter (RMSE)



(a) Std. deviation = 30%

(b) Std. deviation = 50%

(c) Std. deviation = 70%

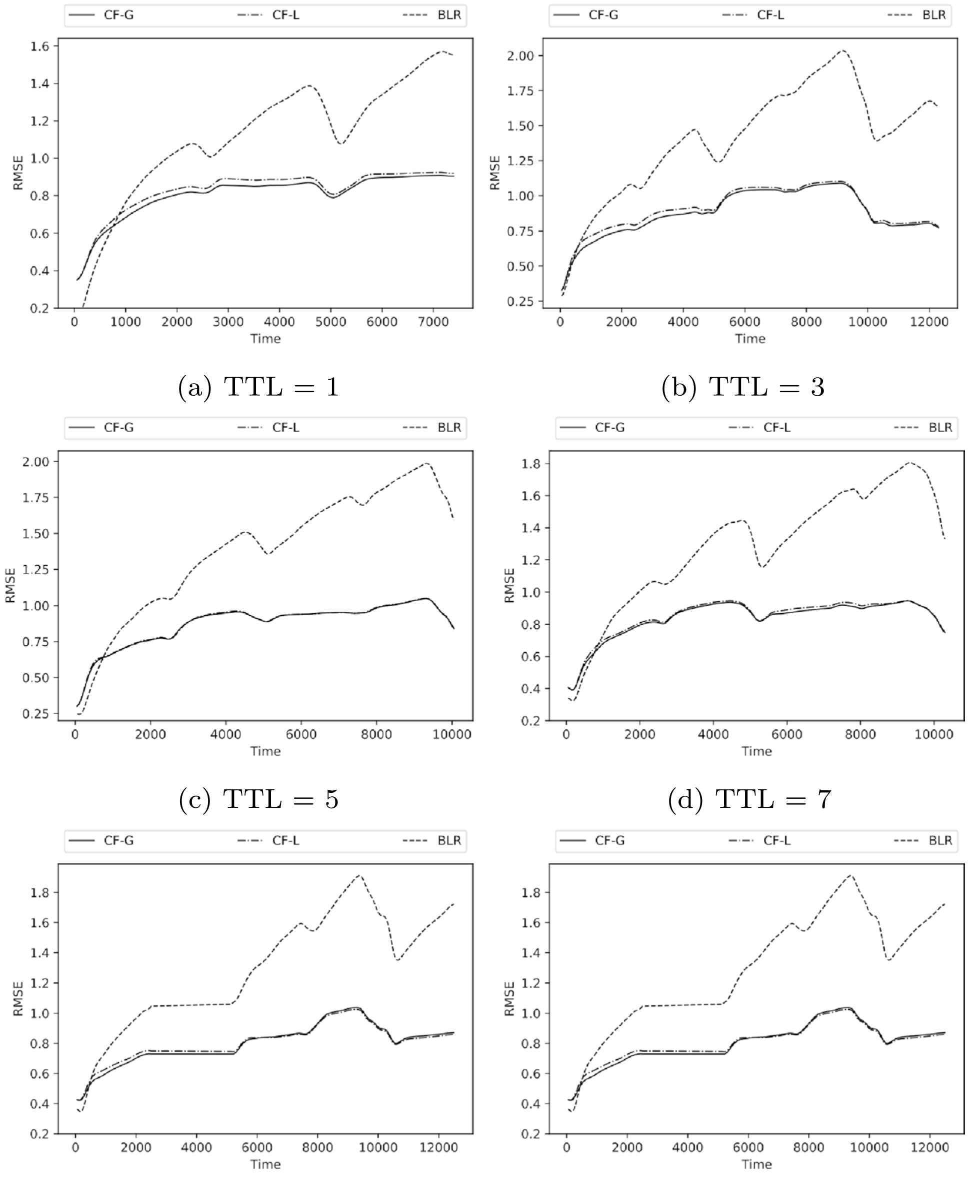
Acknowledgements This research has been supported by the Government of Aragon'Aquitaine-Aragon project PASEO 2.0 (AQ-8), project TIN2016-78011-C4-3-R (AEI/FEDER, UE), project PID2020-113037RB-I00/AEI/10.13039/501100011033 and the Government of Aragon: Group Reference T64_20R, COSMOS research group).

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

References

1. Adomavicius, G, Tuzhilin, A: Context-aware recommender systems. In: F. Ricci, L. Rokach, B. Shapira, P.B. Kantor (eds.) Recommender Systems Handbook, pp. 217–253. Springer (2011)
2. Aggarwal CC (2016) Recommender Systems: The Textbook, 1st edn. Springer Publishing Company, Incorporated
3. Alonso S, Bobadilla J, Ortega F, Moya R (2019) Robust model-based reliability approach to tackle shilling attacks in collaborative filtering recommender systems. IEEE Access 7:41782–41798
4. Anjum SS, Noor RM, Anisi MH (2017) Review on manet based communication for search and rescue operations. Wireless personal communications 94(1):31–52
5. Arsalaan AS, Nguyen H, Coyle A, Fida M (2021) Quality of information with minimum requirements for emergency communications. Ad Hoc Networks 111:102331
6. Boutet A, Frey D, Guerraoui R, Jégou A, Kermarrec AM (2016) Privacy-preserving distributed collaborative filtering. Computing 98(8):827–846
7. del Carmen Rodríguez-Hernández, M, Ilarri, S, Trillo, R, Hermoso, R: Context-aware recommendations using mobile p2p. In: Proceedings of the 15th International Conference on Advances in Mobile Computing & Multimedia, MoMM2017, p. 82–91. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3151848.3151856
8. Chang, N, Liu, M: Revisiting the TTL-based controlled flooding search: Optimality and randomization. In: Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, MobiCom '04, pp. 85–99. Association for Computing Machinery, New York, NY, USA (2004)
9. Chen R, Hua Q, Chang YS, Wang B, Zhang L, Kong X (2018) A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. IEEE Access 6:64301–64320
10. Chen S, Peng Y (2018) Matrix factorization for recommendation with explicit and implicit feedback. Knowledge-Based Systems 158:109–117
11. Cho, E, Myers, SA, Leskovec, J: Friendship and mobility: user movement in location-based social networks. In: Proceedings of

Fig. 16 Results for TTL parameter (RMSE)



(a) TTL = 1

(b) TTL = 3

(c) TTL = 5

(d) TTL = 7

(e) TTL = 9

(f) TTL = 15

the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1082–1090 (2011)

12. Conti M, Giordano S (2014) Mobile ad hoc networking: milestones, challenges, and new research directions. IEEE Communications Magazine 52(1):85–96
13. Eichinger, T, Beierle, F, Papke, R, Rebscher, L, Tran, HC, Trzeciak, M: On gossip-based information dissemination in pervasive recommender systems. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp. 442–446 (2019)
14. Gondaliya SN, Amin K (2019) Solutions to the cold start problem in recommender system: a survey. Journal of Software Engineering Tools & Technology Trends 5(3):14–21
15. Gope, J, Jain, SK: A survey on solving cold start problem in recommender systems. In: 2017 International Conference on Computing, Communication and Automation (ICCCA), pp. 133–138. IEEE (2017)
16. He, R, McAuley, J: Vbpr: visual bayesian personalized ranking from implicit feedback. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)
17. Hermoso, R, Dunkel, J, Krause, J: Situation awareness for push-based recommendations in mobile devices. In: 19th Intern. Conf. on Business Information Systems (BIS), 2016, pp. 117–129. Leipzig (2016)
18. Hermoso, R, Dunkel, J, Rückauf, F: Should i stay or should i go? exploiting visitor movements to derive individualized recommendations in museums. In: Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWIM '18, pp. 343–346. Association for Computing Machinery (2018)
19. Hu, Y, Koren, Y, Volinsky, C: Collaborative filtering for implicit feedback datasets. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 263–272 (2008)

20. Li, G, Chen, Q: Exploiting explicit and implicit feedback for personalized ranking. Mathematical Problems in Engineering **2016** (2016)
21. Liang S, Schuckert M, Law R, Masiero L (2017) The relevance of mobile tourism and information technology: an analysis of recent trends and future research directions. Journal of Travel & Tourism Marketing 34(6):732–748
22. Linden, G, Smith, B, York, J: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing **7**(1), 76–80 (2003). https://doi.org/10.1109/MIC.2003.1167344
23. Lujak M, Billhardt H, Dunkel J, Fernández A, Hermoso R, Ossowski S (2017) A distributed architecture for real-time evacuation guidance in large smart buildings. Computer Science and Information Systems 14(1):257–282
24. Lujak M, Billhardt H, Dunkel J, Fernández A, Hermoso R, Ossowski S (2017) A distributed architecture for real-time evacuation guidance in large smart buildings. Comput. Sci. Inf. Syst. 14(1):257–282
25. Mazeh I, Shmueli E (2020) A personal data store approach for recommender systems: enhancing privacy without sacrificing accuracy. Expert Systems with Applications 139:112858. https://doi.org/10.1016/j.eswa.2019.112858, https://www.sciencedirect.com/science/article/pii/S0957417419305688
26. Ni, SY, Tseng, YC, Chen, YS, Sheu, JP: The broadcast storm problem in a mobile ad hoc network. In: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom ’99, pp. 151–162. Association for Computing Machinery (1999)
27. Núñez-Valdez ER, Quintana D, Crespo RG, Isasi P, Herrera-Viedma E (2018) A recommender system based on implicit feedback for selective dissemination of ebooks. Information Sciences 467:87–98
28. Qi L, Wang R, Hu C, Li S, He Q, Xu X (2019) Time-aware distributed service recommendation with privacy-preservation. Information Sciences 480:354–364
29. Qiu H, Liu Y, Guo G, Sun Z, Zhang J, Nguyen HT (2018) Bprh: Bayesian personalized ranking for heterogeneous implicit feedback. Information Sciences 453:80–98
30. Rezaimehr F, Dadkhah C (2021) A survey of attack detection approaches in collaborative filtering recommender systems. Artificial Intelligence Review 54(3):2011–2066
31. Ricci, F: Recommender systems in tourism. Handbook of e-Tourism pp. 1–18 (2020)
32. Sarwar, B, Karypis, G, Konstan, J, Riedl, J: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, WWW ’01, pp. 285–295 (2001)
33. Senerath, A, Grobler, M, Arachchilage, N: A model for system developers to measure the privacy risk of data. In: Proceedings of the 52nd Hawaii International Conference on System Sciences, HICSS, pp. 1–10 (2019)
34. Shabut AM, Dahal KP, Bista SK, Awan IU (2015) Recommendation based trust model with an effective defence scheme for manets. IEEE Transactions on Mobile Computing 14(10):2101–2115. https://doi.org/10.1109/TMC.2014.2374154
35. Shabut AM, Kaiser MS, Dahal KP, Chen W (2018) A multidimensional trust evaluation model for manets. Journal of Network and Computer Applications 123:32–41
36. Tran TNT, Felfernig A, Trattner C, Holzinger A (2021) Recommender systems in the healthcare domain: state-of-the-art and research issues. Journal of Intelligent Information Systems 57(1):171–201
37. Yang WS, Hwang SY (2013) itravel: A recommender system in mobile peer-to-peer environment. Journal of Systems and Software 86(1):12–20. https://doi.org/10.1016/j.jss.2012.06.041, https://www.sciencedirect.com/science/article/pii/S0164121212001768
38. Zhang F, Lee VE, Jin R, Garg S, Choo KKR, Maasberg M, Dong L, Cheng C (2019) Privacy-aware smart city: A case study in collaborative filtering recommender systems. Journal of Parallel and Distributed Computing 127:145–159
39. Zhao G, Lou P, Qian X, Hou X (2020) Personalized location recommendation by fusing sentimental and spatial context. Knowledge-Based Systems 196:105849