# Multi-Agent Architecture for Point of Interest Detection and Recommendation

Claudia Cavallaro, Gabriella Verga, Emiliano Tramontana, Orazio Muscato

Department of Mathematics and Computer Science

University of Catania, Italy

E-mail: {claudia.cavallaro@unict.it, gabriella.verga@unict.it, tramontana@dmi.unict.it, muscato@dmi.unict.it}

Abstract—Geographical positions are widely employed in many applications, such as recommendation systems. The wide-spread use of mobile devices and location-based Internet services (e.g., Google Maps) gives the opportunity to collect user locations. Taking advantage of a multi-agent system, this work proposes an approach providing users with personalised recommendations of places of interests, such as libraries, museum, restaurants, etc. The approach offers a better experience by giving additional dynamic data (e.g. popularity, as number of users) to a list of Points Of Interest (POIs), and by exploring their temporal relations. Indeed, for POIs, which we determine using a DBSCAN algorithm, we take into account the time slots when the users visited them, to offer a more advanced service. Finally, the approach was designed to preserve the privacy of users, i.e. it does not reveal the position of users.

Index Terms—GPS data, Points Of Interest, Stay Points, Data analysis, DBSCAN, Privacy

## I. INTRODUCTION

Given the extraordinary use of mobile devices and various technologies tracing one's geographical position, it becomes increasingly easier to acquire information relating to users' GPS in real time. This availability has triggered several studies based on user positioning, such as the analysis of the flows of people in the cities [27], or the prediction of people movements [11]. This has also led to the improvement of services that identify the points of interest for a city to offer benefits to users who want to reach a place but they do not have enough knowledge for an immediate choice. Points of Interest, commonly abbreviated POIs, are a well-known concept in literature [2], [15], [31]. A POI is defined as an object associated with a latitude and a longitude which at least one person would reasonably be expected to have an interest or an utility. POI recommendation is one of the services available, suggesting places for users to visit [17].

This paper proposes an approach for POI recommendation using collaborating agents and a centralised server. The server dynamically acquires information coming from agents, which are held on the users' mobile device, creating new suggestions about the next place to visit. Common knowledge is important because, by definition, each agent can independently infer information and share it with the group. In our context, an agent is an application that improves user navigation in a city. An agent communicates with a centralised server to learn new information about POIs. Moreover, an agent gives to the server information about GPS locations for the most visited POIs, and their most frequent time slots.

POIs are taken from an automatic analysis of a real dataset offered by the Geolife experiment [33]. The points found by our analysis were verified by matching the results with Google maps data. It was confirmed that they correspond to real POIs, i.e. parks, restaurants, etc., hence validating our approach. GeoLife GPS trajectories were collected in the framework of (Microsoft Research Asia) Geolife project by 182 users in a period of over three years (from April 2007 to August 2012). A GPS trajectory in such a dataset is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude and altitude [27], [32]–[34].

Thanks to the collaborating agents in our architecture, the proposed approach provides users with: (i) a list of POIs, and for each point, (ii) further information based on real time data gathered from other users, which helps her choose the next destination with greater awareness. A key objective is to have this information as close as possible to real time data, and rank places according to feedback from other users, the most frequent time slots and the time spent visiting a place by other users.

The remainder of the paper is as follows. Next section discusses the related work. Section III describes the proposed multi-agent system. Section IV details the methodology used to find POIs and gives the corresponding results. Section V draws our conclusions.

## II. RELATED WORK

This study lies at an intersection of multiple disciplines, including multi-agent systems, POI recommendation systems, collaborative filtering and privacy preserving systems.

**Multi-agent**: several studies have proposed the use of multi-agent systems (MAS) [8] to a wide range of different domains. In 1998, a study described a supporting system for suggesting possible purchases during shopping based on the GPS position with the use of agents [10]. In general, there are two main approaches to MAS developments: centralised policies (CMAS) and decentralised policies (DMAS) [29].

• A centralised approach consists of taking all of the decisions in one place. In a typical CMAS, a central server collects all the relevant data that come from the different actors (that is, agents) and identifies the decisions for each agent according to the global system

state. The centralised view of the system can be described by a multi-agent Markov decision process model, a good example is presented in [3].

 A decentralised approach consists of making each entity responsible for its own decision. In a typical DMAS, an agent cannot see other agents local states and local actions, and has to decide the next local action on its own. Thus, each agent has only a partial view of the systems global state, and different agents have different partial views. A good example is in [30] whose authors propose a decentralised multi-agent decision process framework that provides the basis for a decision-theoretic study of decentralised policies.

The decentralized architecture has advantages in synchronisation, reusability, scalability, and modularity [12], [14]. However, the complexity of decentralised systems is greater than that of centralised ones. Although decentralisation shows obvious advantages, decentralisation also has its own drawbacks, including that agents cannot predict the group behavior based only on the available local information, possible instability, and sub-optimal decisions.

Due to the importance of total knowledge, our choice fell into the first category. Moreover, the centralised server is able to filter information offering advices to users without sending their sensitive data; this preserves the user's privacy.

**POI**: Researchers have ventured into several studies concerning the analysis of user trajectories. This interest is driven by the possibilities it offers for marketing and the many services that can be offered to users. Since 2005 researchers have faced the problem of analysing trajectories according to space-time. The first studies on the analysis of trajectories offer an overview on how it is possible to analyse the trajectories starting from a set of POIs [15]. Over the years, these analyses have fed other different studies on trajectories, such as calculating the probability of moving from one POI to another, using, for example, the Markov chains [11], [18] and then creating methods that predict the next movements of individual users from the analysis of their POIs.

Collaborative Filtering: Collaborative filtering (CF) techniques are widely adopted for recommendation systems and many CF recommendation methods have been proposed, as e.g. in [19]. The CF approach is one of the approaches for creating recommendation systems. It creates suggestions using a similarity metric among users. The assumption is that similar users probably have similar tastes. The concept of CF was introduced in 1992 by the Xerox research staff within the Tapestry project, a system that allowed users to trace documents based on comments left by other users [13]. Later, several ratings-based automated recommendation systems were developed, e.g. the GroupLens research system [24] provides a pseudonymous CF solution for Usenet news and movies. Other technologies have also been applied to recommendation systems as Bayesian networks [4], [20], [25] and clustering [7], [28].

**Privacy Preserving**: CF techniques have been very successful in e-commerce and in direct recommendation applications.



Fig. 1. Schematic of Centralised Server. Each device identifies an agent.

They are widely used and very useful but they often fail to protect users privacy, hence they have some disadvantages. In [5], [6] the privacy breaches are tackled with cryptographic systems, which can reduce the risk for the user. In other research works, e.g. in [22], each user first disguises her private data, and then sends it to the data collector. Therefore, a Randomised Perturbation (RP) technique is used to disguise private data [1]. Moreover, anonymisation techniques can be used, however these introduce some attack problem, making datasets not very useful [23], [26].

Unlike other approaches, our proposal includes a solution to identify POIs through the use of the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm. Then, collaboration filtering is used with the dynamic calculation of ratings based on user experiences. For such a rating we use spatio-temporal variables offering a dynamic and realistic outcomes. This is done by safeguarding the privacy of users because the centralised server only tracks the movements near the POIs. Furthermore, it is important to offer a service that makes the user and her privacy more secure. To do this we have users sharing their position only if they are close enough to a POI and this information is manipulated to ensure user safety. E.g. the position of a user within the radius of a POI will be saved in our central server with an error rate of about 300 meters in order to preserve the user's privacy. This does not corrupt our system data and better protects users.

#### III. PROPOSED MULTI-AGENT SYSTEM

In our software architecture, a centralised server gathers and analyses the data coming from several agents, with the aim to offer users suggestions and some real time data on POIs to visit. Moreover, each user (with reference to an application) has been modeled as an independent agent that communicates with the centralised server. Therefore, POIs recommendation is based on a multi-agent system performing the following steps (see Figure 1).

- Step 1: the server sends to agents the list of known POIs for a city. Each POI has been previously determined by the DBSCAN algorithm [9] discussed in section IV. Each POI has the following information:
  - the most visited time slots;
  - the number of agents present on the site (POI) in real time;
  - a set of site feedbacks, created by a sentiment analysis algorithm that analyses the comments released by users;
  - a **rating** estimated from previous information that recommends (or dismiss) the POI to the user.
- Step 2: thanks to the rating, the agent chooses a place of interest which the user can visit. As soon as the GPS coordinates are in a range of less than one kilometer from the coordinates of a POI, then the position will be sent to the centralised server, which can determine the number of users present. The GPS coordinates are only sent when in controlled areas in order to preserve user privacy.
- Step 3: once the visit is over, the user can use the agent to issue comments on the place visited. This information will be sent to the server, and there it will be analysed using sentiment analysis algorithms [16], which in turn lets the server determine a score that identifies whether the POI was satisfactory for the user.
- Step 4: the agent will again receive the list of POIs by adding information for a specific POI related to the site already visited based on the experiences of previous users.

Such a cooperation can generate a benefit for groups of people who share the same interests (tourists, students, etc.). Thanks to the exchange of information between agents and servers, it is possible to define the rules for our recommendation system based on POIs (see Figure 2).

In short, the approach presented in this article finds POIs (starting from a set of user trajectories) and uses information exchange with a centralised server to improve city services and user knowledge by creating a content filtering software that creates customised recommendations specific to the user to help him in his choices.

Figure 3 shows a list of POIs nearby to the user. Then, for each POI the user can access the ratings gathered from to other people comments, as well as give her comments. As mentioned in Step 1, POIs are found by the implemented algorithm discussed in the next section. We can see, in Figure 3, four nearby points labeled as POIs:

- Chaofan Weiye Kejiao Bookstore with coordinates: 39.98405510061326, 116.3204636235443;
- *Haidian Stadium* with coordinates: 39.987213527969644, 116.30248430595732;
- Beijing Rural Commercial Bank Zhongguancun Branchcon with coordinates: 39.980016801082485, 116.30856309688643;



Fig. 2. Schematic of POI recommendation. Each device sends its own information to a centralised server which processes them and suggests a new POI for agents. Using the experiences of users based on time, duration, their personal feedback and the next goal, the rating is calculated that suggests a next goal to the leading agent.



Fig. 3. User is presented with a list of POIs and associated dynamic data.

• *Beihang University* with coordinates: 39.98011363182701, 116.34218061609567.

Another nearby POI has been associated with a parking area (however, it is not listed in Figure 3):

• *Satellite Building Parking Lot* with coordinates: 39.97673497237701, 116.33137904408086.

For validating the results of our algorithm finding POIs, each discovered site was checked against Google Maps. Hence, the above list consists of actual sites, being POIs according to Google Maps, which are within a radius of 100 meters from the POIs found by our algorithm.

### IV. METHODOLOGY FOR DETERMINING POIS

We have acquired the data from the database Geolife of Microsoft Research Asia, that contains the routes of 182 users and it reports, at regular intervals of time, each point in GPS coordinates: longitude, latitude, altitude with corresponding



Fig. 4. Plot of clean trajectories in the selected Beijing metropolitan area. Axis: X=longitude, Y=latitude.

date and time. This GPS trajectory dataset contains about 18 thousand trajectories with a total distance of 1,292,951 kilometers and a total duration of 50,176 hours. Most of the trajectories were logged in a dense representation, e.g. every  $1 \sim 5$  seconds or every  $5 \sim 10$  meters per point.

The distance  $d(p_i, p_j)$  between point  $p_i$  and point  $p_j$ , when their coordinates are given by latitude (lt) and longitude (lg), is defined as follows by the Haversine formula. Such a formula provides the distance between two points laying on a sphere surface given their latitude and longitude.

$$d(p_i, p_j) = 2R \arcsin \sqrt{\sin^2 \frac{lt_i - lt_j}{2} + \cos lt_i \cos lt_j \sin^2 \frac{lg_i - lg_j}{2}}$$
(1)

The first step in our analysis was the data cleaning of the trajectories, based on the speed of their GPS points, with the goal to remove inconsistent data. Considering a trajectory, a sequence of GPS points ordered by the time of recording, we computed the velocity of a point as the ratio of the distance from it and its consecutive (applying the Haversine formula, Equation 1) and the difference of time recording them. If this velocity exceeds 100 m/s, the second point was deleted. Another case is when the velocity appeared in the form 0/0, this noise was caused by the GPS device that did not run properly. For our research we have chosen the range of longitude and latitude of  $[116.1, 39.7] \times [116.7, 40.13]$  (Beijing metropolitan area of 51 kilometers per 48 kilometers, see Figure 4).

By plotting the trajectories we can see a second problem: some paths appear broken (not continuous) probably due to the presence of buildings or tunnels that disturb the GPS signal (in some areas the recording is lost). The filtered data, formed by 18,021,911 GPS points, were then grouped into 6 time slots of 4 hours each: Slot1 [00:00:00, 03:59:59], Slot2 [04:00:00, 07:59:59] and so on, in order to analyse the traffic in Beijing during different time slots. Information about these slots are

TABLE I INFORMATION ABOUT DIFFERENT TIME SLOTS

Time	Total number of						
Slot	GPS Points	Trajectories	Users				
1	3978234	5878	156				
2	3729429	4302	150				
3	4976744	6613	166				
4	3107232	4702	168				
5	889076	1505	114				
6	1341196	2537	129				



Fig. 5. Map with the trajectories of Slot 3.

shown in Table I. For time Slot 3's trajectory data we have drawn the GPS data on the map to get a rough idea of the users' activity in this period of time in this area (see Figure 5).

For every slot of time, after grouping trajectories by users, the second step of our work was the StayPoints (SPs) detection [21]. When we find a region in which a user has spent a considerable time on its surroundings, the centroid (the mean of coordinates of the points belong to it) of this cluster represents an SP. The algorithm that we implemented for the SP detection needs as input a TimeThreshold and a DistanceThreshold. If an individual stays over 20 minutes (TimeThreshold) within a distance of 200 meters (DistanceThreshold), a SP is detected. The execution time of the SPs detection algorithm (see 1 for its psuedo-code) for 100 trajectories is about 16 minutes.

We obtained many SPs for every time slot, as shown in Table II. For the Slot 3's trajectories the plot of their SPs is in Figure 6. Then, we focused on POIs that cluster together SPs of different users (at least 10), and checked if, in different time slots, the users that previously had a common POI move together to another one. We applied DBSCAN to the SPs obtained as it works well with large geographical dataset and likewise can be adapted for any distance functions. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a popular unsupervised learning method, proposed in 1996 [9], has been used in model building and machine learning algorithms.

The advantages of DBSCAN are as follows.

• It is very good for separating clusters of high density

**Data:** A trajectory  $T = \{p_i, p_i+1, ...\}$ , a distance threshold (DistThr) and time span threshold (TimeThr) **Result:** A set of stay points SP = {} i=0, cardinality\_traj= |T|while  $i < cardinality_traj$  do j = i + 1;while  $j < cardinality_traj$  do dist=distance ( p i, p j ); if *dist* > *DistThr* then *δ*time=time\_p\_j-time\_p\_i; if  $\delta time > TimeThr$  then S.coords=MeanCoords( $\{p_k | i < k < j\}$ ); S.arrive\_time=time\_p\_i; S.left\_time=time\_p\_j; SP.append(S); break; end end j = j + 1;end i = j;end return SP

Algorithm 1: Pseudocode for SPs detection



Fig. 6. StayPoints of the trajectories on Slot3.

versus clusters of low density within a given dataset;

- unlike K-means, DBSCAN does not require the user to specify the number of clusters to be generated;
- DBSCAN can find any shape of clusters, i.e. the cluster doesnt have to be circular;
- DBSCAN can identify outliers.

The goal of this algorithm is to identify dense regions, which can be measured by the number of objects close to a given point. Two important parameters are required for DBSCAN: Epsilon ("Eps") and minimum points ("MinPts"). The parameter Eps defines the radius of neighbourhood around



Fig. 7. POIs obtained for the trajectories on time Slot 3.

a point x. It is called the EPS-neighbourhood of x. The parameter MinPts is the minimum number of neighbours within "Eps" radius.

For our data, the clustering algorithm DBSCAN has determined clusters for all SPs. We set MinPts equal to 10 or 15 and Eps from a minimum of 200 meters to a maximum of 400 meters. We obtained on average 20 clusters for every time slot (see Table II) that represent significant places for users, i.e. the centroids of these clusters are POIs. E.g. when we considered time Slot 3 and we set that the minimum number of SPs necessary to make a cluster as 15 and the Eps equal to 200 meters, we obtained 29 clusters, hence 29 POIs (see Figure 7).

The last step of our work was to filter the POIs detected according to popularity. We considered only POIs with a number of users greater than 10 (called Popular POIs), in order to understand users interaction and similarity. E.g., for time Slot 3 we obtained 9 POIs shared by a minimum of 11 individuals to a maximum of 80 individuals (see Figure 8). Our experiments have shown that in different time slots a set of different individuals move together to the same POIs, like parks, departments of Universities, shopping centres, hostels, parking spaces, libraries, stadiums, banks, Metro and bus stops. This suggests us a similarity between users. For detected POIs we can further say that our experiments show a correlation of people moving from one POI to another: users remain in these areas in certain common time slots.

In our experiments the execution time of DBSCAN on the 6 time slots ranges from a minimum of 240 ms to a maximum of 1.44 s. Our implementation uses Python 3, and the experiments were run in a host having an Intel Xeon CPU E5-2620 v3 2.40GHz, with RAM 32,0 GB.

## V. DISCUSSION

Thanks to the discussed algorithm, it is possible to find a set of POIs based on gathered trajectories. In our experiments, such data were gathered by Geolife experiments, and the set



Fig. 8. POIs with a number of users greater than 10 for the trajectories of time Slot 3.

TABLE II Results about SPs and POIs obtained

Time		Tota	DBSCAN			
Slot	SPs	Users	POIs	Popular POIs	Eps(km)	MinPts
1	2966	122	18	8	0.3	15
2	3772	124	27	8	0.25	15
3	4146	145	29	9	0.2	15
4	1899	130	24	6	0.2	15
5	751	84	13	4	0.4	10
6	545	84	9	1	0.4	10

of POIs were used as a knowledge base for our recommendation system. The large amount of data was instrumental for validating our approach.

For the above proposed multi-agent system, a setting in our agent application on the smartphone allows collecting GPS coordinates. This is useful for a geographical location were there are no previously gathered trajectories. Then, we can continually extract the trajectories of users and updating both suggestions for users and POIs recommendation.

Using such a setting, each agent releases to the server the GPS coordinates and its identity periodically, however to take into account privacy concerns, the user identity is masked, and the GPS location is randomly moved by at maximum 100 meters. Then, we obtain a trajectory which is not very precise, however still useful.

As trajectories are dynamically gathered, also POIs are determined dynamically, as data arrive to the server. This is possible since our DBSCAN implementation performs well.

# VI. CONCLUSIONS

Nowadays, thousands of users use their mobile device to gain access to new information in relation to their geographical location. This innovation has given rise to new services, such as reading GPS coordinates in order to receive information on nearby Points Of Interest (POIs). In this paper we used a multiagent system for creating recommendations for POIs. The POIs were created from a dataset supplied to us by the Geolife project. From our results we can conclude the following.

- It is possible to find the POIs from a set of trajectories. We have identified about 36 POIs by analysing 182 trajectories. In addition, the Popular POIs, related to more than 10 users, correspond to 30% of the POIs previously found;
- it has been verified that the POIs correspond to wellknown places, e.g. restaurants, parks, etc;
- the visiting hours of the POIs are uniform, therefore each POI has a time slot preferred by users. In general, the most frequent time slot is between 08:00 and 12:00;
- it has been shown that there is a correlation of people moving from one POI to another in the city.

Thanks to the points listed above, the proposed multi-agent system, exchanging information with the centralised server, has been used for the creation of a recommendation tool for POIs.

#### REFERENCES

- R. Agrawal and R. Srikant. Privacy-preserving data mining. ACM Sigmod Record, 29(2):439–450, 2000.
- [2] M. Aliannejadi and F. Crestani. Personalized context-aware point of interest recommendation. ACM Transactions on Information Systems (TOIS), 36(4):45, 2018.
- [3] C. Boutilier. Sequential optimality and coordination in multiagent systems. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, volume 99, pages 478–485, 1999.
- [4] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.
- [5] J. Canny. Collaborative filtering with privacy. In Proceedings IEEE Symposium on Security and Privacy, pages 45–57, 2002.
- [6] J. Canny. Collaborative filtering with privacy via factor analysis. In Proceedings of international ACM SIGIR conference on Research and development in information retrieval, pages 238–245, 2002.
- [7] S. H. S. Chee, J. Han, and K. Wang. Rectree: An efficient collaborative filtering method. In *Proceedings of International Conference on Data Warehousing and Knowledge Discovery*, pages 141–151. Springer, 2001.
- [8] J. Cruz, E. Silva, R. J. Rossetti, D. C. Silva, E. C. Oliveira, and J. Neto. Application of multi-agent systems to shared transport services: A review. In *Proceedings of Iberian Conference on Information Systems* and *Technologies (CISTI)*, pages 1–6. IEEE, 2018.
- [9] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of International Conference on Knowledge Discovery* and Data Mining, pages 226–231, 1996.
- [10] A. E. Fano. Shopper's eye: using location-based filtering for a shopping agent in the physical world. In Agents, volume 98, pages 416–421, 1998.
- [11] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez. Next place prediction using mobility markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*, page 3. ACM, 2012.
- [12] R. Giunta, G. Pappalardo, and E. Tramontana. Aspects and annotations for controlling the roles application classes play for design patterns. In *Proc. of IEEE Asia Pacific Software Engineering Conference (APSEC)*, pages 306–314, Ho Chi Minh, Vietnam, December 2011.

- [13] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–71, 1992.
- [14] J. Z. Hernández, S. Ossowski, and A. Garcia-Serrano. Multiagent architectures for intelligent traffic management systems. *Transportation Research Part C: Emerging Technologies*, 10(5-6):473–506, 2002.
- [15] J.-R. Hwang, H.-Y. Kang, and K.-J. Li. Spatio-temporal similarity analysis between trajectories on road networks. In *Proceedings of International Conference on Conceptual Modeling*, pages 280–289. Springer, 2005.
- [16] B. Liu. Sentiment analysis and opinion mining. Synthesis lectures on human language technologies, 5(1):1–167, 2012.
- [17] B. Liu, Y. Fu, Z. Yao, and H. Xiong. Learning geographical preferences for point-of-interest recommendation. In *Proceedings of ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1043–1051, 2013.
- [18] W. Mathew, R. Raposo, and B. Martins. Predicting future locations with hidden markov models. In *Proceedings of ACM conference on ubiquitous computing*, pages 911–918, 2012.
- [19] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceeding of Eighteenth national conference on Artificial intelligence*, volume 23, pages 187–192, 2002.
- [20] K. Miyahara and M. J. Pazzani. Collaborative filtering with the simple bayesian classifier. In *Proceedings of Pacific Rim International conference on artificial intelligence*, pages 679–689. Springer, 2000.
- [21] R. Pérez-Torres, C. Torres-Huitzil, and H. Galeana-Zapién. Full ondevice stay points detection in smartphones for location-based mobile applications. *Sensors*, 16(10):1693, 2016.
- [22] H. Polat and W. Du. Svd-based collaborative filtering with privacy. In Proceedings of ACM Symposium on Applied Computing (SAC), pages 791–795, 2005.
- [23] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. ACM transactions on information and system security (TISSEC), 1(1):66–92, 1998.
- [24] B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl. Using filtering agents to improve prediction quality in the

grouplens research collaborative filtering system. In *Proceedings of the* ACM Conference on Computer Supported Cooperative Work (CSCW), GroupLens Research Collaborative Filtering System, 1998.

- [25] X. Su and T. M. Khoshgoftaar. Collaborative filtering for multi-class data using bayesian networks. *International Journal on Artificial Intelligence Tools*, 17(01):71–85, 2008.
- [26] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *Proceedings of IEEE Symposium* on Security and Privacy, pages 44–54, 1997.
- [27] E. Tramontana and G. Verga. Get spatio-temporal flows from gps data. In Proceedings of IEEE International Conference on Smart Computing (SMARTCOMP), pages 282–284, 2018.
- [28] L. H. Ungar and D. P. Foster. Clustering methods for collaborative filtering. In *Proceedings of AAAI workshop on recommendation systems*, volume 1, pages 114–129, 1998.
- [29] P. Xuan and V. Lesser. Multi-agent policies: from centralized ones to decentralized ones. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*, pages 1098–1105. ACM, 2002.
- [30] P. Xuan, V. Lesser, and S. Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the fifth international conference on Autonomous agents*, pages 616–623. ACM, 2001.
- [31] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Time-aware point-of-interest recommendation. In *Proceedings of the international* ACM SIGIR conference on Research and development in information retrieval, pages 363–372. ACM, 2013.
- [32] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma. Understanding mobility based on gps data. In *Proceedings of ACM international conference on Ubiquitous computing*, pages 312–321, 2008.
- [33] Y. Zheng, X. Xie, and W.-Y. Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data(base) Engineering Bulletin*, 33(2):32–39, 2010.
- [34] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of ACM international conference on World wide web*, pages 791–800, 2009.