# MARS: A Multi-Aspect Recommender System for Point-of-Interest

Xin Li [1], Guandong Xu [*2], Enhong Chen [1], Lin Li [3]

[1] *University of Science and Technology of China,* [2] *University of Technology, Sydney,* [3] *Wuhan University of Technology*
[1] `{leexin,cheneh}@ustc.edu.cn,` [2] `Guandong.Xu@uts.edu.au (*corresponding author),`
[3] `cathylilin@whut.edu.cn`

*Abstract*—With the pervasive use of GPS-enabled smart phones, location-based services, e.g., *Location Based Social Networking* (LBSN) have emerged . *Point-of-Interests* (POIs) Recommendation, as a typical component in LBSN, provides additional values to both customers and merchants in terms of user experience and business turnover. Existing POI recommendation systems mainly adopt *Collaborative Filtering* (CF), which only exploits user given ratings (i.e., user overall evaluation) about a merchant while regardless of the user preference difference across multiple aspects, which exists commonly in real scenarios. Meanwhile, besides ratings, most LBSNs also provide the review function to allow customers to give their opinions when dealing with merchants, which is often overlooked in these recommender systems. In this demo, we present MARS, a novel POI recommender system based on multi-aspect user preference learning from reviews by using utility theory. We first introduce the organization of our system, and then show how the user preferences across multiple aspects are integrated into our system alongside several case studies of mining user preference and POI recommendations.

## I. INTRODUCTION

Recent years have witnessed the booming of GPS-enabled smart phones, thus the gap between physical and real world has been blurred and location-based services have attracted much attention, e.g., *Location Based Social Network* (LBSN). *Point-of-Interests* (POI) Recommendation as one of the key applications in LBSN, becomes a hot topic in both academia and industry, bringing in great benefits towards customers and merchants. On one hand, for customers, they can gain better user experiences through easy exploration of their interested merchants referred by other customers. On the other hand, merchants may attract more customer visits and increase the business turnover given the appraisal voted by customers.

Most existing POI recommender systems (POI-RS) rely on *Collaborative Filtering* (CF) [1][2], behind which the rationale is that user's interests on unrated merchants can be deduced from historical ratings of like-minded users. Despite the success of the existing systems, there are still limitations, which if solved would improve recommendations effectively. Firstly, most of current POI-RS systems rely merely on the user overall ratings rather than considering the user preference difference across multiple aspects. However, rating in our mind is in fact a fusion process of our preference over multiple aspects, which means various factors e.g., "price", "environment" or "food taste" in restaurant case jointly influence a user to make a final rating. More precisely, a same rating value given by customers on different merchants, say 4-

star, may be of different reasons. For example, students rate higher due to "price" while businessmen care more about "environment". Apparently, current POI-RS lack the capability to differentiate the variance of user preferences over multiple aspects. Likewise, on the other end, merchants may also possess varying qualities over different aspects even they are given the same ratings. Thus we argue that the key challenge in POI-RS is how to reveal the multi-aspect user preference and merchant quality, and in turn, making a better matching between customers and merchants from the corresponding aspects. Secondly, most of LBSNs also allow customers to write reviews in addition to ratings, forming a valuable source for us to capture user preferences, which is far inadequately exploited in current POI-RS.

Aiming to solve the aforementioned problems, we propose a novel utility-based approach to learn multiple aspects of user's preference. Specially, we present MARS (**M**ulti-**A**spect **R**ecommender **S**ystem for POI), which integrates the rating and review info to learn user preference for better recommendations. In summary, MARS encompasses the following benefits:

- It provides a multi-aspect recommendation framework in combination of ratings and reviews.
- It supports a multi-mode visualization for merchants to quantitatively understand their qualities across multiple aspects.
- It implements POI recommendations to users based on their multi-aspect preferences.

## II. SYSTEM OVERVIEW

In this section, we will present the architecture of MARS, which is illustrated in Figure 1. MARS provides functionalities in frontend for users, which we will detail below. The entire system can be divided into two parts - offline part is designed for training our model to provide meta data for the online part, while the online part responds to user's request and then make recommendations.

### A. Offline Part

There are three phases in the offline part. The output of the previous part is the input to the next phase.

*1) Data Collection:* In phase one, we collect data from real world, including user reviews and ratings. Since the raw data is often noisy and sparse, here in order to ensure the data quality, we filter out users and merchants with less than 5 reviews.
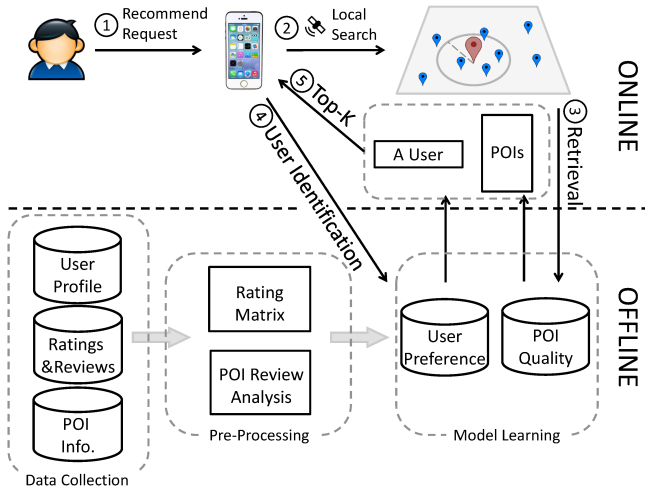
Fig. 1. System Overview



(a) For A Merchant     (b) For A User

Fig. 2. Review Analysis

*2) Pre-Processing:* With the cleaned data, a rating matrix is constructed naturally, which is the main data source for training a recommendation model. Apart from ratings, we also analyze reviews, which we believe convey the user preference information as an auxiliary data source.

We devise a component termed as **POI Review Analysis Component** for review analysis. The technical details will be discussed in III-A. This component is specially for analyzing the review and extract features from reviews. In this component, at merchant side, all the reviews given on the merchant are aggregated as a whole document. We adopt Python Natual Language Toolkit (nltk[1]) to serialize the document and then tokenize each word in the document, so that word frequency can be easily obtained. In this demo, we mainly take the restaurant as the target merchant due to the nature of POI dataset we used in the study. In this case, we define totally six aspects, i.e., *decor, service, taste, value, cleanliness* and *location* to represent the multiple aspects. Within each aspect, following the approach in [3], first several heuristic seed words are chosen, e.g., $decor = \{atmosphere, ambiance, feel, decor, ...\}$. Then more words that have the most similarity are added into the aspect-term set.

Next, we form a word vector for each aspect, and then word frequencies for each aspect in the document are counted, resulting in a word frequency matrix for each document. Since each document is equivalent to a merchant, this word frequency matrix is used to represent the merchant. Finally the rating matrix alongside the word frequency matrices (i.e., the representations of merchants) are input to phase three, i.e., the model training phase. In order to visualize the aspects mentioned by representative words, different colors are highlighted to show which aspects a review is focused on, which is shown in Figure 2. Same color denotes the same aspect.

*3) Model Learning:* In the third phase, recommendation models are trained according to strategies we proposed. As stated in previous sections, the model takes the rating matrix and word frequency matrices as input and outputs user's

[1] http://www.nltk.org/
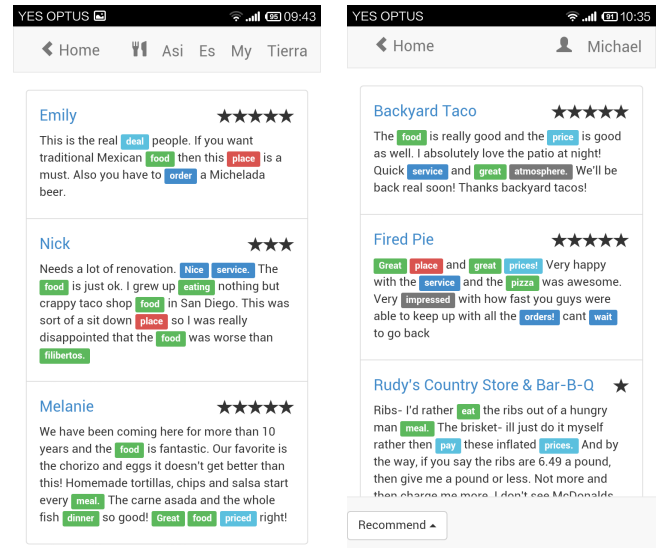
preference $\beta$ and merchant's quality $z$ together with user-feature and merchant-feature matrices derived from matrix factorization, where the former is termed as individual utility, while the latter is termed as collaborative utility in this work. In the demo, for easy illustration, we only demonstrate the use of individual utility to better visualize the difference of user preference and that of merchant quality. Examples of both $\beta$ and $z$ is illustrated in Figure 3(a) 3(b) and 3(c). Specially, radar charts are used here for comparison. By comparing Figure 3(b) and 3(c), we can conclude that different users exhibit distinct preference distribution, which is coincided with our assumption, and in turn, proves our model. For more details, please refer to Section III-B.

### B. Online Part

The online part in Figure 1 is designed as a mobile App so that users can play with the system in real cases. Our system is actually a hybrid App, which is first devised and implemented in web format as a web App and then is shelled within Android mobile devices. As seen from the figure, there are five steps when our system makes recommendation to a particular user.

*1) Recommendation Request:* At first, the user requests for a recommendation at a certain location by using our recommender system. There are several ways for a system to provide recommendations, such as push notification service [4] or request & respond service [5]. Here we adopt the conventional way to implement request & respond service so that it only responds when a user is requesting for a recommendation in order to avoid disturbing users too much. In implementation, a request button "Recommend" is designed at the bottom of the app, which can be seen in Figure 3(b) and 3(c).

*2) Local Search:* In step 2, the system responds to the user's request by locating his position via GPS, which is implemented through geolocation API in html5.

*3) POI Retrieval:* As soon as the location is acquired, it is sent to the server to retrieve all the POIs within a certain

| (a) Merchant's quality | (b) User 1's preference | (c) User 2's preference | (d) Recommendation |

Fig. 3. Demonstration

range of the area centered with his location. Those merchants are selected as candidates for recommendations. The common method by retrieving POIs is through MongoDB [6]. In case of the small size of our database, geographical locations are stored and retrieved using standard function in Mysql.

*4) User Identification:* Accordingly, user identification and merchants' ids are sent via APIs through remotely visiting the server. After that, the server returns user's preference $\beta$ and merchants' qualities $z$ to the device.

*5) Top-K Recommendation:* By following the algorithm given in Eqn. (2), the top-k merchants are recommended to the user to reflect the maximum utility satisfaction of users. In particular, we implemented a top-K selecting algorithm in [7]. In MARS, $K$ is set to 20. A folded button is designed at the bottom of each user's page and by clicking it users can choose which qualitative aspect to sort the results. There are totally seven modes for users' choices, which are displayed in Figure 3(d).

## III. TECHNICAL BACKGROUND

In this section, we will brief the technical details in MARS. Formally, user $u_i$ can give a rating, denoted as $R_{ij}$ in a range of 1-5, to merchant $v_j$. Note that we use POI and merchant interchangeably in this paper. Besides rating, a user can write a review about his experience with this merchant, denoted as review $D_{ij}$. In our implementation, the aggregated reviews from merchant's perspective are utilized, i.e., $D_{*j}$.

### A. Review Process

A review contains multiple aspects, which are high level concepts derived from user reviews to explicitly show what factors may influence user's rating, denoted by $A = \{A^1, A^2, ..., A^k, ..., A^{|A|}\}$. As aforementioned, six aspects are defined in our system, i.e., $|A| = 6$. Each aspect $A^k$ can be represented as a collection of words, i.e., $A^k = \{w_1^k, w_2^k, ..., w_n^k\}$. Words in the same set may be synonyms.

As mentioned before, different users may place disparate weights on those aspects. Thus for a given user $u_i$, user aspect preference is defined as a collection of weights over each aspect, denoted as: $\beta_i = (\beta_i^1, ..., \beta_i^k, ..., \beta_i^{|A|})$. Correspondingly, every aspect of each merchant should have a score to indicate how well the merchant do in such aspect, namely merchant's quality. Scores of all aspects w.r.t. merchant $v_j$ form a vector, termed as $z_j = \{z_j^1, ..., z_j^k, ...z_j^{|A|}\}$. Note that, a final score by multiplying $\beta_i$ and $z_j$ indicates the coupling between the user and the merchant and also it reflects how interested a user in a merchant. Higher score suggests more interested.

Particularly, in order to obtain $z_j$, we adopt the approach proposed in [3]. Thus, scores of all aspects w.r.t. a particular merchant is modeled as:

$$z_j = \sum_{w=1}^{n} \gamma \odot W_j \qquad (1)$$

where $W_j$ is the word frequency matrix for $D_{*j}$ with each column representing an aspect. $\gamma$ is the word sentiment polarity matrix corresponding to each word and $n$ is the length of word vector for each aspect. By Hadamard product $\odot$, we sum all the word sentiment polarities in each aspect to describe the aspect sentiment orientation. $\beta$ and $\gamma$ are parameters, which will be learnt from model training in next subsection.

### B. Model Training

In this subsection, we will detail how to automatically learn parameters. In economics, discrete choice models are derived from random utility model (RUM), where user behavior is under the assumption that maximizes the utility. Utility is a representation of satisfactions over a set of considerations. In [8], utility $\mathcal{U}$ is modeled as $\beta z^T + \varepsilon$, where $z$ is a feature vector comprised by the user considerations, $\beta$ is a parameter vector to weight the utility of each considerations and $\varepsilon$ is the error term without observations.

We term $\beta_i z_j$ in section III-A as an individual utility $\mathcal{U}^i$ because it reflects a user's own preference on a merchant. On the other hand, collaborative behaviors also play an important

**Algorithm 1:** POI recommendations based on utility

---

**input** : Reviews $D$ containing $W$, Rating Matrix $R$
**output**: $\Omega = \{U, V, \beta, \gamma\}$

1 Random $\Omega$;
2 **for** *step = 1 to MAX STEP* **do**
3     $\Omega_l \leftarrow \Omega_l - \eta\nabla_{\Omega_l}$;
4     **if** *converge* **then**
5        return $utility = UV^T + \sum \beta\gamma W$;
6     $\eta \leftarrow \frac{\eta}{1+step/MAXSTEP}$;
7 **for** *each user* **do**
8     sort $utility$ of all POIs;
9     recommend top-K;

---

role in affecting user rating process, which can be obtained through matrix factorization and is termed as collaborative utility, denoted by $\mathcal{U}^c$. Through normalizing, the both utilities are additive. Then linearly combining the individual and collaborative utility, we immediately obtain:

$$\mathcal{U}_{ij} = \mathcal{U}_{ij}^c + \mathcal{U}_{ij}^i + \varepsilon_{ij} \qquad (2)$$

where $\mathcal{U}_{ij}^c$ is the collaborative utility and equals to $U_i V_j^T$, while $\mathcal{U}_{ij}^i$ is the individual utility and equals to $\beta_i z_j^T$. A tuning parameter should be added to control the contributions of two parts, which has been integrated into the individual utility parameter $\beta$, thus making $\beta$ a global parameter and no need to further tuning the both side.

Under the assumption that the error term follows the Gaussian distribution and by placing spherical multivariate Gaussian prior on the parameter $\Omega = \{U, V, \beta\}$ with zero means, we could obtain the loss function as:

$$log\mathcal{L} = \sum_{i,j} I_{ij}(R_{ij} - U_i V_j^T - \beta_i z_j^T)^2 + ||\Omega||_F^2 \qquad (3)$$

A local minimum of the objective function can be achieved by performing gradient descent.

### C. Recommendations

After learning the model, we get the parameters $\beta$ and $\gamma$ along with latent feature matrices $U$ and $V$, thus in turn to obtain both individual and collaborative utilities at hand. Recommendations could be made according to any of the utility or the both, which is shown in Algorithm 1.

## IV. DEMONSTRATION DETAILS

The demonstration is a simulation of the real situation in POI recommendation, that is to recommend merchants to a particular user in a location. The system is implemented in Python and PHP on a MySQL database running on SAE (Sina App Engine), a cloud computing platform. MARS is built based on a real Yelp data set and demonstration attendees can play with our recommend system both from user's and merchant's perspectives.

### A. Dataset

We adopt the data set published by Yelp [2], which contains the data from Phoenix, AZ within last 10 years. In order to

---

[2]http://www.yelp.com.au/dataset_challenge

make the system more efficient, we filter out those users and merchants with less than 5 reviews and finally get 10971 users and 5671 merchants. Totally 136636 reviews along with their ratings have been analyzed.

### B. Audience Participation

The demonstration attendees can participate either from user's or the merchant's perspective. From user's side, our system will simulate to login as a random user. At this user's page, 20 reviews along with their ratings of his historical data will be listed. Attendees can get a clear view about what aspects does a single review talk about due to the color tags around aspect words. Another radar charts is shown at the top of the page to give a high level idea of the user's preference. By clicking the name of merchants regarding to the reviews, it will jump into merchant's page, which we will demonstrate below.

There is a button at the bottom of the page called "recommend", which provides seven modes (all aspects and six individual aspect) to recommend merchants according to qualitative metrics. Users can choose several of them for recommendation. At the recommendation page, the top 20 results of user specified interested aspect(s) will be listed along with their scores, which are calculated by our model.

At the merchant's page, it displays as a dashboard for merchant's owner to know customers' concerns. All the strengths and weakness will be displayed with radar charts. Aiming to provide better customer experiences, the owner could refer to this analysis.

## V. CONCLUSION

In this demonstration, we present a recommend system by learning user's multi-aspect preference through reviews. In order to modeling this, a novel utility based approach is proposed. This system can recommend merchants according to several qualitative particular aspects, which significantly improves users' experiences and provides a novel way to evaluate the merchant.

### REFERENCES

[1] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *ICDM*, 2008, pp. 263–272.
[2] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, pp. 30–37, 2009.
[3] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis on review text data: A rating regression approach," in *Proceedings of the 16th ACM SIGKDD*, 2010, pp. 783–792.
[4] Parse, "Android push notifications," https://parse.com/tutorials/android-push-notifications.
[5] H. Su, K. Zheng, J. Huang, T. Liu, H. Wang, and X. Zhou, "A crowd-based route recommendation system-crowdplanner." in *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*, 2014, pp. 1178–1181.
[6] S. Francia, *MongoDB and PHP - Document-Oriented Data for Web Developers.* O'Reilly, 2012.
[7] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware." *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 614–656, 2003.
[8] J. Neumann and O. Morgenstern, *Theory of games and economic behavior*, commemorative edition ed. Princeton Univ. Press, 2007.