Serendipity-based Points-of-Interest Navigation

XIAOYU GE, University of Pittsburgh
PANOS K. CHRYSANTHIS, University of Pittsburgh
KONSTANTINOS PELECHRINIS, University of Pittsburgh
DEMETRIOS ZEINALIPOUR-YAZTI, University of Cyprus
MOHAMED A. SHARAF, United Arab Emirates University

Traditional venue and tour recommendation systems do not necessarily provide a *diverse* set of recommendations and leave little room for *serendipity*. In this paper, we design MPG, a Mobile Personal Guide that recommends: i) a set of diverse yet surprisingly interesting venues that are aligned to user preferences; and ii) a set of routes, constructed from the recommended venues. We also introduce EPUI, an Experimental Platform for Urban Informatics. Our comparison with the state-of-the-art schemes indicates that MPG is capable of providing high-quality venues and route recommendations while incorporating seamlessly both the notion of diversity and that of serendipity.

Additional Key Words and Phrases: POIs Recommendation, relevance, diversity, serendipity

ACM Reference Format:
Xiaoyu Ge, Panos K. Chrysanthis, Konstantinos Pelechrinis, Demetrios Zeinalipour-Yazti, and Mohamed A. Sharaf. 2020. Serendipity-based Points-of-Interest Navigation. *ACM Trans. Internet Technol.* 1, 1, Article 1 (January 2020), 31 pages. https://doi.org/10.1145/3391197

1 INTRODUCTION

The rapid developments in mobile computing lead to the transformation of traditional Yellow pages to mobile applications that are conveniently reachable, up-to-date, localized, and personalized, connected to the surroundings, and versatile in many other ways. Platforms such as Yelp and Foursquare allow their users to generate multimedia content (e.g., text, image) and share their experiences with their peers.

Many systems have been developed and built on top of these platforms for recommending specific venues or Points of Interest (POIs) to be visited by users, i.e., *digital travel guides*. Given that this digitization results in a richer and up-to-date content, the possibilities for providing flexible, personalized guides are huge. Personalized tour systems have also appeared in the literature (e.g., [28]) that take into consideration spatiotemporal constraints, users' interests, etc. Nevertheless, many of the approaches to date are monolithic and myopic to user preferences, returning generic recommendations where every location is treated equally (e.g., [6, 16, 20, 38, 58]).

Authors' addresses: Xiaoyu Ge, University of Pittsburgh, Department of Computer Science, xiaoyu@cs.pitt.edu; Panos K. Chrysanthis, University of Pittsburgh, Department of Computer Science, panos@cs.pitt.edu; Konstantinos Pelechrinis, University of Pittsburgh, Department of Informatics and Network Systems, kpele@pitt.edu; Demetrios Zeinalipour-Yazti, University of Cyprus, Department of Computer Science, dzeina@cs.ucy.ac.cy; Mohamed A. Sharaf, United Arab Emirates University, Department of Computer Science, msharaf@uaeu.ac.ae.

1533-5399/2020/1-ART1 $15.00
https://doi.org/10.1145/3391197

Example: The following scenario best illustrates how a well-designed digital travel guide will help travelers. Pam is a businesswoman that visits Pittsburgh for the first time on a business trip. After finishing all her business meetings in the morning, she wants to have a cup of coffee before finding a fulfilling restaurant for lunch, so she turns to her smartphone to quickly find 100 coffee shops and restaurants near her via a digital travel guide. Since the digital travel guide already knows her coffee preference, via a *personalized* profile, it is able to recommend precisely the nearest Starbucks as Pam likes Starbucks's cappuccino. As she finishes her coffee, Pam wants to proceed with the lunch, so she queries the digital travel guide for the top-2 restaurants through a user-friendly query interface. This time the digital travel guide provides her with two *diverse* restaurants, Souvlaki and Pizza Hut, each from one of her two favorite food types, Greek and Pizza, respectively. After lunch, Pam wanted to spend some relaxing time and visit some of the interesting venues of Pittsburgh, so she again turns to her digital travel guide, and this time for a recommendation of four interesting venues. The digital travel guide knows Pam is a history and science enthusiast, and she loves to visit museums, so it first recommends two great museums, namely the Heinz History Center and the Carnegie Science Center which are about the same distance from her current location. It also recommends for the third spot a smaller museum at the Fort Pitt, established by the British between 1759-61 and later developed as Pittsburgh. However, for the fourth spot on the recommendation list, instead of recommending another museum, the digital travel guide *surprises* her with the recommendation of the Nationality Rooms of the Cathedral of Learning [1], which is something that Pam has never heard of, but instantly recognized as a *must-go* venue after reading their descriptions.

Clearly, one of the key components of such a digital travel guide is its venue recommendation algorithm. A common approach to venue recommendation among prior works—personalized or not—is the ranking of venues based on some quality features (e.g., [21, 28, 40, 51, 55]). Consequently, the top-k venues are returned. There are two major drawbacks to this approach. First, it does not allow for a **diverse** set of recommendations; because similar venues tend to have a similar ranking, and thus the top venues will all be similar to each other with high probability. Second, these systems are overwhelmingly focused on optimizing an **efficiency objective**, such as minimizing the distance covered, and maximizing the benefit obtained from the route as captured by a measure of venue quality (e.g., [28, 40, 55]). It is only in recent years that efforts have been made to consider objectives that go beyond the pure efficiency (e.g., [21, 51]) but they are still in a nascent stage.

The first drawback at a high level translates to a poor recommendation since the *effective* choice of the user is reduced, given that many of the recommended venues will offer similar experiences. The second drawback leads to recommendations that do not capture the great and livable aspect of a city, as the features that contribute to capturing a livable environment are missing in these systems. That is, traditional recommendation systems do not encourage users to step outside their comfort zones and discover unexpected but interesting venues. The objective in this paper is to design a *Mobile Personal Guide* (MPG) that addresses both drawbacks of the existing systems and delivers an interactive response (i.e., sub-second) for each query to guarantee smooth user interactions.

MPG Overview To address the first drawback of the common approach, MPG takes into consideration the user's preferences and provides a set of venues that satisfy the imposed constraints with maximized diversity. The diversity (as formally defined later in Section 3) is essentially a measure of dissimilarity of the venues based on external attributes. Simply put, MPG outputs a set of high-quality yet diverse venues. To illustrate this objective, let us consider again our example scenario above, where 35 venues satisfy Pam's preferences, depicted in Figure 1. Assume that Pam only has time to visit 4 of them, as in our example scenario above. The venues represented by the

[1]The Nationality Rooms are a collection of 31 classrooms in the University of Pittsburgh's main building Cathedral of Learning depicting and donated by the national and ethnic groups that helped build the city of Pittsburgh. [2]

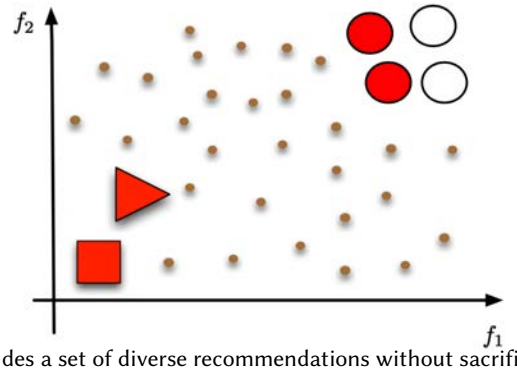


Fig. 1. MPG provides a set of diverse recommendations without sacrificing the quality.

large circles correspond to the top-4 venues ranked; for instance, based on their popularity (i.e., top-4 venues with highest popularity). The venues represented by the triangle and the square are ranked 5th and 6th, respectively. The rest of the venues are ranked lower and are represented by the brown dots. The space corresponds to two external features (f_1, f_2) that define the similarity of a pair of venues. In particular, the top-4 venues are very close in this space and hence are similar (or in other words, they have low diversity in the space defined by f_1 and f_2). A system that does not consider the diversity of the recommended venues would most certainly choose these venues as output. However, MPG allows the user to explore the available venues in this latent space—without sacrificing the quality of the recommendations—and so, it would return to the user the top-2 venues, as well as the 5th and 6th, ranked venues (the venues with a red fill). As we can see, this set is more widespread in this latent space as compared to the top-4 venues (whether these are restaurants or museums). Furthermore, since it is impossible for any recommendation system to guarantee 100% user satisfaction, it is also important to construct a set of recommendations with broad coverage of the initial candidate set of venues (i.e., recommend venues that represent a large portion of the candidate venues). This will help the system to increase the chance of recommending venues that are interesting to the user and provide the user with the possibility to zoom into venues that they are interested in and discover more similar venues if they prefer.

In a nutshell, our approach consists of the following basic steps. First, we begin with assigning intensity value I_p^v to venue v based on its *popularity*. We also assign a distance intensity value $I_{d,q}^v$, which captures the distance between the current location q of the mobile user and venue v. By combining I_p^v and $I_{d,q}^v$, we obtained a composite intensity value, $I_{p,d}^v$. We further tune these intensity values based on the preferences of user u obtaining $I_{p,d,u}^v$, which forms our composite intensity value space I_k^v. Finally, I_k^v along with a vector $\mathbf{f}_v$ that represents venue v in the latent space (i.e., external attributes) form the input to our modified *P*refDiv (*Preferential Diversity*) algorithm [23] whose output is the required recommendations. One of the advantages of *P*refDiv is that it offers the mobile user the ability to adjust the balance between relevance and diversity in the returned results.

To address the second drawback of the common approach, MPG introduces the important concept of Serendipity, which has been identified as a characteristic that can significantly improve the quality of experience that a city-dweller has [41]. We define serendipity as the recommendation of items that are relevant to a user but unexpected and happily surprising. Recent studies (e.g., [31, 39]) have shown that recommending items that are relevant while unexpected is an essential factor in increasing user satisfaction and prevent filter bubbles.

It is worth pointing out that serendipity is a very different concept from diversity, which constitutes ex-post process and can be combined with serendipity. Specifically, diversity in the context of

the recommendation systems can generally be understood as a property that applies to a set of recommended items that measures how dissimilar items are to each other in a set, which involves measuring the average pairwise dissimilarity of items through some dissimilarity metrics. Even though a consensus definition for serendipity in the context of the recommendation system has not yet been formed, based on previous studies (e.g., [3, 46, 61]), relevance and unexpectedness have been identified as the two most essential elements in defining the serendipity in the recommendation context. In particular, the relevance refers to the quantitative measure of the benefit of recommending an item, and the unexpectedness requires the recommended items to be unexpected by the user and is typically achieved through randomness and non-determinism algorithms. Unlike diversity, serendipity does not prevent similar items to be recommended as long as the recommended items are relevant to and unexpected by the user (e.g., the recommendation of two historical museums to a history lover where he/she is not even aware that the second museum exists).

In other words, diversity and serendipity are complementary to each other. To better illustrate the difference between diversity and serendipity, let us consider a recommendation request for a set of k items that are both relevant and diverse from a database with N items. Typically, the number of mutual dissimilar items C in the database that are relevant is much larger than the requested k items (i.e., $k < C \leq N$). A typical recommendation system will employ some method to recommend a subset of k item from C that has the best trade-off between relevance and diversity, such as top-k. In this scenario, adding serendipity means picking the k recommendation from these C relevant yet diverse items in a non-deterministic fashion to fulfill the unexpectedness. Clearly, from this example, one can tell that serendipity is a stand-alone concept that can be applied in conjunction with the diversity and relevance.

MPG incorporates serendipity in two layers, both of which explore *randomness*. At the first layer, serendipity is still achieved through the recommendations of venues based on *P*refDiv. We develop a probability-based variant of *P*refDiv, namely, *Probabilistic Preferential Diversity* (*p*PrefDiv), which incorporates the necessary serendipity for a surprisingly interesting recommendation of venues. The second layer of serendipity is achieved through the recommendation of routes. Specifically, we utilize random walks to generate a set of initial routes. The randomness of this process essentially captures the desired serendipity, since the venues to be included in the route are not chosen in a way that optimizes a pre-defined objective. Lastly, the Pareto front is deployed to pick final recommendations that show high quality with respect to both diversity and serendipity.

Contributions In summary, this paper's contributions are as follows:

- We introduce a new mobile service, coined MPG, which is capable of generating venue recommendations that are not only *popular* and *relevant* to user's preference but are also *diverse*. Our method ranks venues based on user preferences, the distance to the venue, and the popularity of a venue based on check-in information, and achieves diversity with a novel semantic distance function.
- We formally introduce the notion of *Serendipity*, and effectively integrate serendipity in MPG into both venue and route recommendation by means of randomness while still preserving the relevance and diversity aspects of recommendations. We described in detail *p*PrefDiv (*Probabilistic Preferential Diversity*), which supports serendipity in venues, and formally introduced a random-walk based routing method that achieves the serendipity in route recommendation. Specifically, we propose four novel schemes for achieving serendipity in route recommendations, namely *PrefDiv+RandomWalk, pPrefDiv+RandomWalk, pPrefDiv+ShortestDistance* and *pPrefDiv+HighestRelevence*. We also propose a new metric to measure their degree of serendipity, which is the deviation of a route from the most anticipated, non-deterministic, top route.

- We present as a proof-of-concept the design and implementation of two prototype systems: i) MPG for Android devices and ii) EPUI, our Experimental Platform for Urban Informatics that supports the personalized venue and route recommendation in an interactive manner. EPUI supports a number of state-of-the-arts approaches for the venue and tour recommendations and enables their comparison by presenting the recommended venues and tours visually on a street map as well as displaying evaluation metrics through summary dashboards. EPUI's implementation is flexible and allows for different diversity and indexing schemes to be incorporated as well as new recommendation algorithms to be uploaded. Hence, EPUI can be used by the broader research community to facilitate the evaluation of alternative POIs recommendations and route construction algorithms.
- We experimentally compare our proposed solutions to other state-of-the-arts alternatives (i.e., K-Meroids, DisC Diversity, and Random for venue recommendations, and two deterministic baseline schemes for serendipity based route recommendations) using three data sets (i.e., NYC, San Francisco, and Pittsburgh). In our comparisons, in addition to the standard metrics (e.g., coverage, intensity value), we used our proposed metric that capture the degree of serendipity in route recommendation. Our extensive experiments verify the effectiveness of our proposed solutions, showing that MPG can successfully increase coverage of the result-set compared to other alternatives, achieve a significantly better *Relevance-Diversity* trade-off than other methods and offer high-degree of serendipity. They also verify its ability to support interactive response time.

The rest of the paper is organized as follows: In the next section, we review related work. In Section 3, we formally define our problem and describe the basic components of the MPG design. We introduce the serendipity venue and route recommendation algorithms in Sections 4. The MPG prototype and EPUI experimental platform are presented in Section 5. Our thorough experimental evaluation and its findings are reported in Section 6. Section 7 concludes our work and briefly describes our future directions.

2 RELATED WORK

In this section, we discuss existing works that are strongly related to our own. In particular, we will present studies related to trip planning as well as methods for query personalization.

2.1 Trip planning and spatial recommendations

During the last years, there has been a large volume of studies that focus on methods for personalized location/Point-of-Interest (POI) recommendations to social-network users [8, 47, 57]. The majority of existing work utilizes collaborative-filtering techniques [57], geometric embeddings [8], or they even incorporate features present in the users' social network [47] to associate every venue with a score, which is representative of the probability of a user enjoying (or liking) a particular venue. The aforementioned studies consider and evaluate each venue independently. Hence, motivated by this monolithic view of these methods, recent work has focused on recommending *tours* of locations. For instance, De Choudhury et al. [16] focus on segmenting streams of spatiotemporally tagged photos into paths and then assembling these paths into itineraries. Similar studies by Kurashima et al. [40] and Yoon et al. [58], are based on geotagged content from photo-sharing media (e.g., photo streams, GPS trajectories) to recommend future travel paths. However, these approaches come with their drawbacks. For example, in order to be applicable, the presence of training sequences of spatiotemporally tagged photographs (or other similar traces) is required. These approaches cannot handle multiple types of venues that cater to different user needs. The same is true for interactive systems [6, 20, 38], which iteratively personalize or improve a tour based on user feedback.

The support for multiple types of venues is considered by Ardissono et al. [5], where the user *manually* selects a venue from each desired type, and then a tour traversing the selected venues is proposed. More recently, Gionis et al. [28] developed a system based on dynamic programming algorithms, which provides spatially constrained tours based on user preferences of the category of venues. Other spatial recommendation approaches focus on reconstructing and recommending routes based on existing location trajectories (e.g., [11, 55]).

Existing works (e.g., [29, 42, 43, 57]) that employed a collaborative filtering-based approach for venue recommendation identify a set of similar users based on user previous interactions (e.g., feedbacks, ratings) with venues and then predict a user's rating for a venue that they have not rated based on the observed ratings of the venue of similar users and the properties of the venue. In recent years, one commonly used algorithm for collaborative filtering based recommender systems is matrix factorization (e.g., [29, 42]) that decomposes a user-item rating matrix into a product of two-factor matrices U and I. The prior represents the relation between some latent factors and the users, and the latter represents the relation between latent factors and the items. Such decomposition essentially maps both users and items to a joint latent factor space of dimensionality f, such that recommendations are derived based on the user-item interactions modeled as inner products in that space. Matrix factorization can also be generalized to accommodate more information (e.g., contextual) going beyond just the information of users and items. Such a generalization of matrix factorization is called tensor factorization [32], which allows for more flexible and generic integration of multi-dimensional data. Instead of the traditional 2-dimensional user-item matrix, tensor factorization integrates additional information by modeling the data as an N-dimensional tensor (e.g., User-Item-Context).

Recently, Lu et al. [43] has proposed methods that aim to promote serendipity in matrix factorization based recommender systems. However, these methods use "serendipity" as a way to recommend unpopular items that are relevant to a user's preference, and do not not capture the non-deterministic, unexpected characteristic of true serendipity. Although matrix or tensor factorization based techniques provide a high degree of accuracy on the estimation of the preferences between users and items, they do not provide any guarantees for diversity or serendipity of the results. Furthermore, matrix or tensor factorization based recommender systems suffer from cold start related problems (e.g., when a new user or item has been added to the system) as well as low scalability due to their superlinear time and space complexity [52]. Therefore, they are not suitable for interactive exploratory range queries in digital travel guides that demand sub-second latency.

Other recent works (e.g., [7, 60]) have studied the problem of recommending a trip with a diverse set of venues by employing taxonomies or categorical hierarchies to diversify the venues. Their approaches only capture the coarse-grained difference between venues, and thus, they are unable to distinguish between venues that are within the same category. As such, for a top-4 query that recommends four venues from two venue categories coffee shop and gym, they could recommend {Starbucks 1, Starbucks 2, 24 Hour Fitness A, and 24 Hour Fitness B} rather than the more diverse recommendation of {Starbucks 1, Costa Coffee, 24 Hour Fitness A, and Anytime Fitness}.

Existing work addresses the problem of finding spatially diverse routes in two different ways. Lu et al. [44] proposed to integrate the output of several different location recommender systems to form the top-k recommendations with two aggregation strategies: i) to use the scaled score produced by each recommender systems, and ii) to rely on the ranked position of a location in each recommender system's recommendation, and assign higher scores to the top locations. Both these aggregation strategies have their drawbacks. The score-based approach requires a proper scaling between the scores produced by different recommendation algorithms, which is typically a challenging task, and the position-based approach is less effective when the overlap between the recommendations produced by different algorithms is small. Furthermore, it does not provide

explicitly guarantee on the diversity of the final recommendations. Xu et al. [56] propose to generate the desired subset of spatially diverse routes from a large set of relevant but semantically similar routes (i.e., routes that pass through POIs of the same categories given by the user). Since this approach only focuses on finding routes that are diverse with each other based on the spatial distance between the two routes, it does not take into consideration the semantic diversity between POIs or serendipity. To the best of our knowledge, there is no study to date that considers the venue coverage, semantic diversity (in a latent space) in combination with serendipity when it comes to recommendations.

2.2 Query personalization

Relevance Relevance rankings constructed from the combination of user preferences and result diversification techniques have been previously proposed to deal with the problem of information overload (i.e., avoid overwhelming users with a large volume of irrelevant results). Ranking techniques are comprehensively surveyed in [53].

Mostly these techniques can be distinguished based on the type of preferences they support for filtering and ordering data. These techniques primarily handle only one type of preference, either *quantitative* or *qualitative*. However, each preference type has its own advantages and disadvantages. Hybrid schemes that support both qualitative and quantitative preferences have been proposed in an attempt to exploit the advantages of both types of preferences while eliminating their disadvantages [27, 37]. In this work, our proposed algorithms can work with any existing relevance ranking model that returns a set of sorted tuples/objects along with their scores/intensity values.

More recently, in [10], the author studied the problem of producing rankings while preserving a given set of fairness constraints. In particular, the proposed algorithm takes as input, a utility function, a collection of sensitive attributes (e.g., gender, race) of each item, and a collection of fairness constraints that bound the number of items with each sensitive attribute that are allowed to appear in the final results. It outputs a ranking that maximizes the relevance with respect to the given utility function while respecting the fairness constraints.

Diversity In recent years, result diversification has been extensively studied in many different contexts (e.g., [18, 33–36]), and with various definitions such as similarity, semantic coverage [4], and novelty [15]. In our work, we focus on the similarity definition and use MaxMin and MaxSum, which are two widely used diversification models, as baselines.

The goal of these two diversification models is to select a subset S from the object space R, so that the minimum or the total pairwise distances of objects in S is maximized. Recently, a number of variations of the MaxMin and MaxSum diversification models have also been proposed (e.g., [17, 48]) to address the problem of diversifying continuous data. Formally, MaxMin and MaxSum are defined as follows:

Definition 1. *MaxMin* generates a subset of R with the maximum $f = min_{p_i, p_j \in S} dt(p_i, p_j)$ where dt is some distance function $p_i \neq p_j$ for all subsets with the same size.

Definition 2. *MaxSum* generates a subset of R with the maximum $f = \Sigma_{o_i, o_j \in S} dt(o_i, o_j)$ where dt is some distance function $o_i \neq o_j$ for all subsets with the same size.

DisC Diversity (or *DisC for short*) [18] is the most recently proposed diversity framework and solves the diversification problem from a different perspective. In DisC Diversity, the number of retrieved diverse results is not an input parameter. Instead, users define the desired degree of diversification in terms of a tuning parameter r (radius). DisC Diversity considers two objects o_i and o_j in the query result R to be similar objects if the distance between o_i and o_j is less than or equal to a tuning parameter r (radius). It selects the representative subset $S \in R$ according to the

following conditions: i) For any objects in R, there should be at least one similar object in S; and ii) All objects in S should be dissimilar to each other. These two conditions ensure both the coverage and the dissimilarity property of a diverse result set.

The key differences between PrefDiv algorithms used by the MPG and DisC Diversity are: i) PrefDiv algorithms follow the Top-k paradigm, which provides users with the option to specify the size of the final result set by assigning a value to parameter k, whereas DisC Diversity adjusts the size of the result set by changing its radius parameter r. ii) The PrefDiv algorithms focus on both the *relevance* of the result set with respect to the users' preference and the *diversity* of the result set. DisC Diversity focuses mainly on the most diverse representative subset with two scenarios that only illustrate the possibility of using DisC Diversity to handle such relevance-aware diversity requests; however, they do not mention any specific strategies on how one can dynamically change the radius r.

Another way to generate a diverse, representative set of results is through clustering. One example of this would be *k-Medoids*, which is a well-known clustering algorithm that attempts to minimize the distance between points in a cluster and the center point (medoid element) of that cluster. The k-Medoids algorithm can be classified into two stages: In its first stage, it generates a set of k clusters C = {c1, c2, ..., ck} based on some distance function dt. In the second stage, one element from each cluster is selected to be part of the result set R. Several strategies for selecting an element from each cluster could be employed. For instance, one strategy is to choose the center point of each cluster, which is expected to deliver high diversity, and another strategy would be to choose the point that has the highest intensity value for each cluster. However, since there is no parameter that can be tuned either manually or automatically to balance the trade-off between relevance and diversity, k-medoids is unable to balance such a trade-off in fine granularity.

Multi-Criteria Objective Optimization Even though the goal of diversity is to avoid loss of potentially important data due to its low ranking, the result of diversification does not automatically imply relevance for the users. This was the underlying motivation for the top-k diversification techniques, such as *P*refDiv [23], *Swap* [59], the *Query Manifold* (*QM*) framework [62]. This was also the motivation for the multi-objective optimization approaches in which the first objective is relevance and the second objective is dissimilarity [63]. As opposed to *P*refDiv, *Swap*, and the multi-objective optimizations that recommend relevant and diverse data, *QM* recommends a set of relevant and diverse queries.

The difference between *P*refDiv and Swap is that the latter seeks diversity through pairwise distances of items among the result-set and filters out items that contribute less to diversity. Swap ensures relevance by removing items that drop the relevance below a pre-defined threshold. In contrast, *P*refDiv seeks diversity by eliminating similar items and ensures relevance by using a relevance-focused greedy algorithm that can reflect the user-specified relevance distribution.

The most widely known approach that is targeted directly at optimizing the trade-off between diversity and relevance was introduced by [9]. In this work, the authors have proposed the twin-objective function called *Maximal Marginal Relevance* (MMR), which combines both relevance and diversity aspects in a single comprehensive, objective function with a scaling factor λ. When $\lambda = 1$, the MMR function equals to a standard relevance ranking function, when $\lambda = 0$ it computes a maximal diversity ranking. Compared to PrefDiv, MMR always requires to compute its objective function against the entire set of initial candidate items for each of its k representative items, whereas, PrefDiv does not rely on a single comprehensive, objective function and in most cases only requires a very small number of comparisons for each of its k representative items. Recently, a new MMR function that integrates regret minimization was proposed to generate the relevance

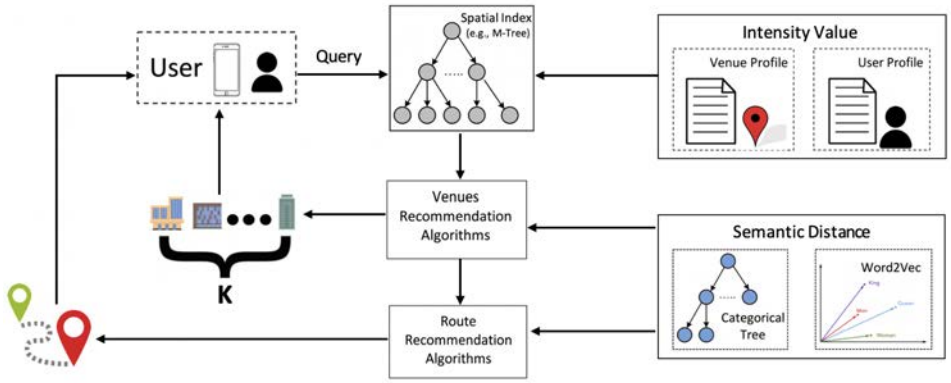


Fig. 2. MPG System Architecture.

score [30]. This new score attempts to minimize the disappointment of users when they see k representative tuples rather than the whole database.

Div-Astar [50] is another multi-criteria objective optimization solution that is graph-based, in which each node corresponds to one item in the original data. This diversity graph is sorted according to the relevance score, and a a^* algorithm is used to find the exact solution for diversifying top-k results. That is, *Div-Astar* considers the problem as finding the optimal solution for the maximum weight independent set problem, which has been proven to be NP-hard.

Our proposed MPG system differs from all of the above works, as MPG leverages the capability of *p*PrefDiv, the nondeterministic version of *P*refDiv, to provide venue recommendation that optimizes three important aspects of venue recommendation (i.e., relevance, diversity, and coverage), while delivering results that are unexpected by the user. Furthermore, MPG is capable of providing serendipity-based route recommendations with relevance and diversity in mind.

3 MPG DESIGN

In this section, we introduce the formal underpinnings of MPG and its core components (Figure 2). We begin by formally defining relevance and diversity that are central to our work.

3.1 Relevance, Intensity, Diversity, Similarity, and Coverage

Relevance We represent the degree or score of relevance of an item o to a user u by the *Preference Intensity Value* (I_u^o).

Definition 3. Preference Intensity Value *(I) is a real value between* -1 *and* 1. *Negative preferences are expressed using any value in* $[-1, 0)$; -1 *is used to express complete dislike. Positive preferences are expressed using any value in* $(0, 1]$; 1 *is used to capture the most likability. Equality/indifference is expressed using* 0.

Diversity We measure the diversity of a set of items S by measuring how dissimilar, i.e., the semantic distance beyond a threshold, each item in S is with respect to all other.

Definition 4. Dissimilarity: *Let O be the set of items in the database. Two objects o_i and $o_j \in O$ are dissimilar to each other $dsm_\varrho(o_i, o_j)$, if $dt(o_i, o_j) > \varrho$ for some distance function dt and a real number ϱ, where ϱ is a distance parameter, which we call radius.*

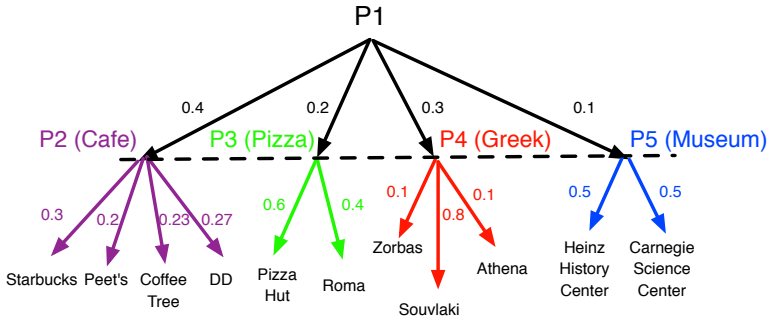


Fig. 3. Pam's sample hierarchical user profile. The first level corresponds to the coarse-grain preference profile (P_1), while each one of the sub-trees stemming from P_1 corresponds to the preferences within each category (e.g., preference P_2 corresponds to the "Cafe" venue type).

Definition 5. Similarity: *Two objects o_i and $o_j \in O$ are similar to each other, if $dt(o_i, o_j) \leq \varrho$ for some distance function dt and a real number ϱ. We use $sim_\varrho(o_i, O)$ to denote a set of items in O that are similar to an item o_i, such that $\forall o_j \in sim_\varrho(o_i, O), o_j \neq o_i$.*

Definition 6. Coverage: *Given a set of items S and a result-set R, where $R \subseteq S$, coverage corresponds to the percentage of items in S that satisfies $dt(o_S, o_R) \leq \varrho$, such that $o_R \in R$ and $o_S \in S$ where dt is some distance function and ϱ is a real number parameter.*

3.2 MPG Objective - Problem Statement

We first formulate the algorithmic problem that lays in the epicenter of MPG, and then discuss the components of our solution.

Problem 1. *Given a set of geographical points $G = \{g_1, g_2, \ldots, g_l\}$, a popularity index ξ_{g_i} for location g_i, a query point q, a reach r, a desired level of serendipity ψ and a profile set that encodes user preferences $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, identify a set $G^* \subseteq G$ ($|G^*| = k$) with maximized diversity $\Delta(\mathcal{S})$, while satisfying the constraint set $h(G^*, \mathcal{P}, q, r, \psi, \xi)$.*

In our setting, the set G corresponds to the set of available venues/Points-of-Interest (PoI). The query point q corresponds to the current location of the mobile user, while r represents the maximum allowed distance between q and any point in the chosen solution V^*, and ψ indicates whether serendipity would be involved in the query or not. The set of preferences $\mathcal{P}$ captures the profile of the mobile user with respect to his/her interests. Finally, the constraints described by function h include: (i) a geographic constraint that ensures that the maximum distance between the currently location of the mobile user and any venue recommended does not exceed r (i.e., $d(q, v_i) \leq r, \forall v_i \in V$); and (ii) a personalization constraint that ensures that the output set of venues is compatible with the user preferences (i.e., $V \models \mathcal{P}$).

Given this problem setting, the actual MPG system includes an interface that will obtain: i) the current location of the user q (e.g., through the GPS sensors, NFC sensors); ii) the user inputs such as the reach r; and iii) the set of types of venues she is interested in and the number of venues k she would like to know about. The preference of the user will be stored either in the system (i.e., bound to the user account) or stored on the mobile device and uploaded to the system/server at the time of the request. In response to a request, MPG provides a set V^* of the recommended venues based on the definition of Problem 1. These recommended venues can further be used for route recommendations if desired or requested.

3.3 Venue Preferences

MPG utilizes a hierarchical profile $\mathcal{P}$ that mirrors the typical venue classification. In the typical classification every venue v is associated with a categorical type $\mathcal{T}_v$, and every category may be a subcategory of another category. For example, an Italian pizzeria belongs to the category "Italian restaurant", which can belong to the higher level category "Restaurants", which can itself belong to the category "Food" and so on. At the top level of the hierarchy there are ten categories, namely, *Arts & Entertainment*, *College & University*, *Food*, *Nightlife Spots*, *Outdoors & Recreation*, *Events*, *Professional & Other Places*, *Residences*, *Shops & Services* and *Travel & Transport*. However, in order to build highly personalized and specific profiles, we use the bottom layer of hierarchy, as well as the specific venues visited.

In particular, given the set of check-ins C_u of mobile user u, we build a hierarchical profile $\mathcal{P}$. At the top level, the preferences of the user are expressed in terms of the (normalized) frequencies of this user's visitations with respect to the types of venues. The second layer of the user profiles further provides the normalized frequencies of venues for the different types of locations visited by u. Figure 3 presents a sample profile for user Pam in our example scenario (in Section 1). Preference P_1 is a coarse-grain preference profile, which informs the system that Pam prefers to spend 40% of her time in coffee shops, 10% in museums, 20% in burger joints and 30% in Greek restaurants. Preferences $P_2 - P_5$ are able to distill further Pam's preferences. For instance, she appears to prefer Starbucks more compared to Peet's coffee. Such a preference profile can nowadays easily be constructed with a variety of services, e.g., Google Maps Timeline; a detailed discussion around the construction remains outside the scope of this paper. In our prototype implementation, we combine the user check-in information from Foursquare with the foursquare category hierarchy to build the user profiles [1].

3.4 MPG Intensity Value

MPG utilizes a preference intensity value for relevance that is a composite of syntactic-based and semantic-based intensity values: *distance-based, popularity-based*, and *profile-based* intensity values.

3.4.1 Distance-based Intensity Value. The physical distance between the current location q of the mobile user and venue v can also be used to obtain an intensity value for v. In particular with d_q^v being the normalized distance between q and v's location the distance-based intensity value can be defined as:

$$I_d^v = 1 - \frac{d_q^v}{r} \tag{1}$$

In the above equation, the distance has been normalized based on the maximum allowed distance from Problem 1, that is, r. Note here that d_q^v can be, in principle, equal to 0. However, this happens when the current user location q is at a venue v. Given that the user is already at this location, these venues are not considered by our system.

3.4.2 Popularity-based Intensity Value. An important factor that can impact the choice of a venue v from MPG is its popularity. With c_v being the number of total visits in venue v, i.e., the number of check-ins in v, and s_v being the number of unique visitors in v, we define the popularity-based intensity value of v as:

$$I_p^v = \lambda \cdot \frac{c_v}{\max\limits_{i \in V} c_i} + (1 - \lambda) \cdot \frac{s_v}{\max\limits_{i \in V} s_i},\ \lambda \in [0, 1] \tag{2}$$

where V is the set of all the venues. λ is a scaling factor between the total number of visitors and the number of unique visitors. When $\lambda = 1$, only the number of total visitors determine the popularity-based Intensity value of a venue and when $\lambda = 0$, only the number of unique visitors determine the popularity-based Intensity value of a venue. The default is $\lambda = 0.5$. This intensity

value essentially corresponds to the popularity index ξ used in the formal definition of the MPG problem.

3.4.3 Profile-based Intensity Value. The degree or strength of relevance of a venue v is expressed by the preference-based intensity value I_u^v derived from the user's profile. In particular, the profile-based intensity value is a combination of the score of the type of the venue (i.e., the coarse-grain preference) with the specific venue (i.e., fine-grain preference) score. As stated above, since these scores are derived from the user's check-ins C_u, the profile-based intensity value I_u^v for venue v and user u is computed as follows:

$$I_u^v = \omega \cdot \frac{C_u^v}{\sum_{v_j \in t} C_u^{v_j}} + \omega \cdot \frac{\sum_{v_j \in t} C_u^{v_j}}{\sum_{t \in T} \sum_{v_j \in t} C_u^{v_j}} \tag{3}$$

where, ω is the trade-off parameter between the coarse-grain and fine-grain preference scores with a default of 0.5, C_u^v is the number of check-ins that u had in v, t is the venue type of v and T is the set of all venue types. Going back to Pam's case (Figure 3), the coarse-grain preference score would reflect her preference towards the nodes in the second level of her hierarchical user profile (i.e., Cafe, Pizza, Greek, and Museum), and the fine-grain preference score would be her preference towards the leaf nodes of her hierarchical user profile.

3.4.4 Composite Intensity Value. Having distance-based intensity value I_d^v and popularity-based intensity value I_p^v, we can combine them in one intensity score as follows, with γ the scaling factor:

$$I_{d,p}^v = \gamma \cdot I_d^v + (1 - \gamma) \cdot I_p^v, \ \gamma \in [0, 1] \tag{4}$$

We can further combine $I_{d,p}^v$ with profile-based I_u^v in a manner similar to Equ. 4 and obtain a value that combines the profile preference, the popularity and the distance of the venue from the current location of the user. More specifically:

$$I_{u,p,d}^v = \alpha \cdot I_u^v + (1 - \alpha) \cdot I_{d,p}^v, \ \alpha \in [0, 1] \tag{5}$$

Eq. 5 combines three different elements (user preference through I_u^v, venue popularity through I_p^v or $I_{p,\pi}^v$ and geography through I_d^v) into a single intensity score. This combined score is the composite MPG intensity value of v, I_k^v. Here, we would like to emphasize that the order with which we combine the three intensity values (i.e., I_u^v I_p^v (or $I_{p,\pi}^v$) and I_d^v) to obtain I_k^v (or $I_{k,\pi}^v$) does not impact the recommendations of MPG. This is due to the fact that MPG outputs a total order of the venues based on these three factors. The absolute values themselves for I_k^v will be different, but the order will always be the same. Furthermore, to provide an additional degree of personalization, each of the above functions includes adjustable trade-off parameters (i.e., α, γ, λ, ω) that are personalizable with respect to user interest.

Based on the above discussion, the results of the composite intensity value (i.e., Equation 5) can be obtained in constant time O(1). Therefore, the overall runtime per query for computing composite intensity value for each user query is linear with respect to the number of the initial set candidate venues involved in the user's range query.

3.5 The Semantic Similarity Measurement

In order for MPG to produce semantically diverse results, there must be a way to measure the semantic distance between two venues. Due to the ambiguous nature of venues, it is challenging to obtain a precise measure between an arbitrary pair of venues. To address this challenge, we propose a two-part semantic similarity measure that combines both the categorical information of the venue

and the semantic of the venue's name to produce the accuracy measure of semantic similarity between any pair of venues. Next, we discuss the details of our semantic distance function.

3.5.1 Category Tree. As mentioned previously, MPG leverages the Foursquare Category Hierarchy [1] to derive the user preferences and build user profiles. However, such information can also be used to determine the similarity among venues. In particular, MPG accelerates both operations by constructing a *category tree* to capture the category structure of venues in Foursquare as a tree. Each internal node in the category tree represents a type of venue, where each internal node represents the subcategory of the parent node with each leaf node representing the actual venue. There are in total of 10 categories at the top-level of this hierarchy. Each internal node in a category tree contains the following attributes: ID of the category it represents, the name of the category, a pointer to the parent node, and a list of pointers to each of its children nodes. Since a category tree can have an unlimited number of degrees, all of the children node pointers are stored as hash tables, with the key being venue ID and the value is the actual pointer.

The user profiles are further derived from the preference hierarchy, as described above in Section 3.3. The preference hierarchy consists of the top-level categories and the leaf nodes of the category tree (Figure 3). The category tree can be used to calculate the similarity distance between two venues v_i and v_j as follows:

$$Sim_{Tree}(v_i, v_j) = 1 - \frac{Ancestors\_Path}{Longest\_Path} \tag{6}$$

where *Ancestors_Path* is the number of common ancestors between the venues v_i and v_j and *Longest_Path* is the number of nodes on the longest path to the root from either v_i and v_j.

3.5.2 Name Semantics. Although the category tree is able to measure the similarity between two venues, this measurement is not very accurate as it only provides a coarse granularity distance between two venues. Specifically, this measurement cannot distinguish the difference between two venues that are under the same subcategory, for example, "McDonald's" and "Burger King", as both of them share the exact same ancestors.

Our hypothesis is that the names of venues reflect their identity/semantics in a finer granularity than their category type. Therefore, to overcome this limitation, MPG utilizes Word2Vec [45], an advanced NLP technique, which supports fine granularity distance calculation between two venues by going beyond syntactic comparisons. Word2Vec provides the implementation of two word-vector representation computing models: *Continuous Bag-of-Words* model (CBOW), which predicts the current word based on the sourcing words, and *Continuous Skip-gram* model, which seeks to use the current words to predict surrounding words. Both of these models are based on the Neural Net Language Model.

With Word2Vec, the similarity of word representations goes beyond simple syntactic regularities. Specifically, word vectors capture many linguistic regularities. For example, after obtaining the word representation in vector space, the resulting vector can have the following properties, such that vector('King') - vector('Man') + vector('Woman') results in a vector that is closest to the vector('Queen'). MPG uses CBOW model to generate all word vectors.

The difference between two words under Word2Vec are calculated through the cosine similarity of two-word vectors, such that cosine similarity is defined as following:

$$Sim_{Vec}(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2}\sqrt{\sum_{i=1}^{n} {B_i}^2}} \tag{7}$$

where n is the length of vector, A_i and B_i are elements of vector A and B, respectively.

The current word vectors we adopted support phrases that consist of up to two words. For venue names that have more than two words or are not contained in the word vectors, we split the phrases into single words and then obtain word vectors for each individual word in the phrases. The final vector of a phrase is obtained through the average of all vectors for each word in this phrase. Since the accuracy of Word2Vec strongly depends on the quality of the word vectors, a large real-world corpus is needed in order to obtain high-quality word vectors. We have experimented with various corpuses in an attempt to generate the highest quality word vectors. The best suitable word vectors we obtained were generated from the entire English Wikipedia that consists of 55 GB of plain text. The resulting word vectors contain over 4 million entries. In order to effectively query the word vectors, MPG stores all the word vectors in memory as a hash map.

Similar to category-tree based similarity, the Word2Vec based similarity has its own biases. We were able to overcome these biases of the individual similarity metrics by combining them (Eqs. 6 and 7) and measuring the similarity between two venues v_i and v_j as follows:

$$Sim(v_i, v_j) = \frac{Sim_{Tree}(v_i, v_j) + Sim_{Vec}(A, B)}{2} \tag{8}$$

where A and B are representing the vector representation of venue v_i and v_j, respectively.

Note that for efficiency purposes, both Category Tree and Word2Vec have been cached in main memory as hash tables. This allows MPG to compute the above composited semantic distance (i.e., Equation 8) in constant time for each venue and, in turn, enable a linear runtime complexity for each user query.

3.6 Spatial Indexing Structure

One of the main operations in MPG is to generate a set of nearby neighbors. In order to speed up this process, MPG utilizes the widely adopted *M-tree* spatial index structure [13]. M-tree uses triangle inequality for efficient range queries, similar to those required in MPG. An M-tree is a balanced tree index that is designed to handle a large scope of multi-dimensional dynamic data in general metric spaces. An M-tree partitions the space in such a way that it generates spherical boundary regions around some of the indexed items, called *pivots*, with some bounding radius r. Each internal node has at most N entries and contains the following attributes: a pivot p_v, the bounding radius r around p_v, a pointer pt to the subtree that is rooted in the pivot p_v, and the distance between p_v and its parent pivot. The distance of a subtree from p_v is guaranteed to be within the bounding radius r. Each leaf node in the tree has two attributes: the item that is being indexed and the distance between this leaf node and the parent pivot. The efficiency of M-tree has been proven in a large body of existing works (e.g., [19]), and a detailed cost model for M-Tree's search complexity can be found in [14].

4 MPG'S SERENDIPITY ALGORITHMS

In this section, we introduce the key algorithmic components of MPG, particularly the venue and route recommendation algorithms, which produce relevant, diverse, and yet unpredictable results within a specified latency requirement.

4.1 Venue Recommendation Algorithm

As discussed earlier in Section 1, serendipity is an important concept in helping recommendation algorithms improve user satisfaction, which is focused on recommending items that are relevant while unexpected. To achieve the conflicting objectives of serendipity in recommend *relevant* while *unexpected* venues, we incorporated the serendipity under the algorithmic framework of our previously proposed algorithm *Preferential Diversity* (PrefDiv) [23] that produces a set of

recommendations, which balances the trade-off between relevance and diversity in an efficient manner. In fact, *P*refDiv provides parameter A to tune the balance between relevance and diversity in the returned result set of recommendations R. When $A = 1$, R would simply be the top k items from the initial set, i.e., the items with the k highest intensity values, and when $A = 0$, R contains k dissimilar items from the initial set. Since *P*refDiv is an iterative algorithm, in the case A is in between 0 and 1, the final results will contain at least $A * k$ items from every iteration, and in each iteration, A will be divided by half.

The basic logic of *P*refDiv (Algorithm 1) is as follows: it first sorts the objects in the initial set $O = \{o_1, ..., o_n\}$ in descending order of their intensity values and splits them into groups of k objects. In each iteration, it evaluates the objects in a group for diversity, starting with the group that contains the highest intensity objects (Lines 4 & 5). Item o_i with the highest $I_u^{o_i}$ in the group T_O is moved into the final result set R, if there is no object in R similar to o_i, i.e., $sim_\varrho(o_i, R)$ is empty; otherwise it is marked as "Eliminated". Also, all objects in $sim_\varrho(o_i, T_O)$ are marked as "Eliminated". While there are still objects left in T_O that are not being marked as "Eliminated", it processes the next unmarked one o_j with the highest $I_u^{o_j}$ in the same manner (Lines 6 - 13). Note that the $Accept()$ function on Line 9 always returns True for *P*refDiv, as it is designed for *p*PrefDiv, as discussed below. PrefDiv ends an iteration by finalizing the moving of objects into R according to A, as mentioned above. If fewer than the required $A * k^{iter}$ objects were moved in R, then the difference s is covered by moving the top s objects with the highest intensity values that have been marked as "Eliminated" in T_O into R (Lines 14 - 16). The iterations continue until either k objects are selected ($|R| = k$), or if all items in O are examined. If the size of R is still less than k, $k - |R|$, items with the highest intensity values that have been marked as "Eliminated" will be selected and added to R (Lines 17 - 22).

The adoption of *P*refDiv helps to ensure the recommended results are relevant to the user's intent. However, *P*refDiv is a deterministic algorithm, and thus, it does not incorporate the concept of serendipity. To introduce serendipity in *P*refDiv, we devise a non-deterministic version of *P*refDiv, coined, *Probabilistic Preferential Diversity* (*p*PrefDiv). *p*PrefDiv offers serendipity through probability weighted by relevance. In contrast to *P*refDiv, when a venue x is qualified to be one of the recommendations for a range query q, x is not automatically included in the result set (Line 9: $Accept(x) = True$). Instead, *p*PrefDiv decides whether or not x can be added to the result based on the following probability:

$$p(x \text{ is accepted}) = \frac{CI(x)}{\text{Argmax}_{i:O}\, CI(i)} \tag{9}$$

where $CI(x)$ is the composited intensity value of I_d^v, I_p^v and I_u^v of a venue x, and O is the set of all venues within q. In the case of *p*PrefDiv, this probability (i.e., Eq. 9) is computed for each x by the $Accept()$ function on Line 9 of Algorithm 1. If x is accepted (Line 9), it will be presented as one of the recommendations. Otherwise, x will be discarded, and *p*PrefDiv would proceed to the next venue.

For example, assume $O = \{o_1, ..., o_n\}$ is the initial set of venues, R is the set of result that *p*PrefDiv is producing, o_u is an unmarked venue that has a composited intensity value of 0.6, which is currently the highest among all unmarked venue in O, and there is no object in R similar to o_i (i.e., $sim_\varrho(o_i, R)$ is empty). According to *P*refDiv, o_u should be included in the result-set. However, *p*PrefDiv would use intensity value weighted probability to determine the acceptance of o_u. For instance, if the highest composited intensity value in O to be 0.8, then o_u would have a probability of $0.6/0.8 = 75\%$ to be included in the result-set.

The introduction of randomness by means of a composited intensity value-based probability allows *p*PrefDiv to incorporate serendipity into the venue recommendations, while still preserving the high-intensity value and semantic distance feature of the *P*refDiv algorithms.

ALGORITHM 1: PrefDiv/pPrefDiv Main

Require:
Set of objects O, a size k, a radius ϱ and A tuning parameter
Ensure:
One subset R of O
$S \leftarrow \emptyset$
while there exists unmarked items in O and $|R| < k$ **do**
  $S \leftarrow$ Pick k items with highest composited intensity from O
  for all $o_i \in R$ **do**
    mark $\forall o_j \in sim_\varrho(o_i, S)$ as "Eliminated"
  for all unmarked items o_i in S **do**
    if $Accept(o_i)$ **then**
      $R = R \cup o_i$, s.t. $o_i \in S$ is unmarked and $I_u^{o_i} \geq I_u^{o_j} : \forall o_j \in S$
      for all unmarked $o_u \in S$ **do**
        if $o_u \in sim_\varrho(o_i, S)$ **then**
          mark o_u as "Eliminated"
  while number of unmarked items in $S < A \cdot k$ **do**
    $R = R \cup o_i$, s.t. $o_i \in S$ is unmarked and $I_u^{o_i} \geq I_u^{o_j} : \forall o_j \in S$
  $A = A \cdot 0.5$
  if first iteration **then**
    create new set $G \leftarrow \forall o_j \in S$, s.t. o_j is marked
  $O = O - (O \cap S)$
if $|R| < k$ and $\forall o_j \in O$, s.t. o_j are marked **then**
  while $|R| < k$ **do**
    $R = R \cup o_j$, s.t. $o_j \in G$ and $I_u^{o_j} \geq I_u^{o_i} : \forall o_i \in G$
Return R

Time Complexity According to the above discussion (Algorithm 1), we can observe that the worst case complexity of *p*PrefDiv is $O(kN)$. Fortunately, as the size of k is usually a small number, *p*PrefDiv should typically behave as a linear algorithm. Furthermore, as we will show in our empirical studies (Section 6), depending on the diversity constraints, *p*PrefDiv typically does not need to examine all original items. That is, a very small set of items would be sufficient enough to produce R if the radius is appropriately defined.

4.2 Route Recommendation Algorithm

By employing *p*PrefDiv, MPG is capable of producing informative recommendations, which significantly benefits the user's exploration of cities. However, the best route that the user can take to visit these recommended venues remains to be decided. Thus, MPG further supports route recommendations that are constructed based on the recommended venues with variable length.

To incorporate serendipity in the recommendation of routes, for a user-defined route length k, where k is the number of venues that comprise a route, we utilize random walks to generate sets of initial routes.

Random Walks The random walks are performed on a weighted graph between the venues. The weight w_e of edge e considers the distance between the current location of the user, the number of visits to the venue, and the user's preference towards a certain type of venue. The probability that our random walk will go through edge e is proportional to $1/w_e^\gamma$, where γ is a system parameter.

The weight assigned is a tunable parameter that users can explore. Before generating the final route recommendations, we perform a number of ξ random walks to yield the initial set of candidate routes. Such serendipity incorporated by the random walks is essential to enable the MPG to yield unexpected, yet interesting route recommendations to the user. Since the complexity for generating one random walk path is simply O(k), the total time complexity for random walks-based routing is simply O(ξk) per each user query.

Using this initial set of ξ routes, we determine our final recommendations by introducing the quantitative measurements of serendipity. Since serendipity implies that the recommended routes should be unexpected by the user, therefore, to capture this unexpectedness, we first have to determine the most intended (i.e., expected) route, and then measure the difference between the recommended route and the most intended route. As we have modeled the user's intention using the intensity score, thus, the routes r_I^* constructed based on the ranking of the intensity value naturally represents the most intended route.

HighestRelevence We propose *HighestRelevence* as the method to construct routes based on the ranking of intensity value. For a given set of object O, such that $|O| = k$, and a length k', where $k' < k$, HighestRelevence constructs a route of length k' by selecting the first k' objects with highest composited intensity value from O, then order each object in descending order according to their combined intensity value. Whenever possible HighestRelevence always seeks to separate two venues with the same category (according to the category tree), thus to improve the usability of the route by preventing two venues with the same category to be visited in sequence.

Serendipity Metric To measure the serendipity of a given route r, we first find the set of overlapping venues O_{r,r_I^*} between r and the route that maximized the intensity value r_I^*. Based on the set of overlap venues we compute the *Overlap Factor (OvF)* between r and r_I^* as following:

$$OvF(r, r_I^*) = 1 - \frac{|O_{r,r_I^*}|}{|r|} \tag{10}$$

Then we compute the *Normalized Longest Common Subsequence* (*NLCS*) between r and the route that maximized the overall intensity value r_I^* as follows:

$$NLCS(r, r_I^*) = \frac{|L_{r,r_I^*}|}{|r|} \tag{11}$$

Thus, the serendipity σ_r of r would be:

$$\sigma_r = (1 - NLCS(r, r_I^*)) * OvF(r, r_I^*) \tag{12}$$

The idea for quantifying the serendipity of a route by measuring its normalized longest common subsequence and overlap factor with the route r_I^* that has the maximum overall intensity value is because r_I^* is the most anticipated route based on prior knowledge. Therefore, routes with large deviation from the most anticipated route can be considered as less expected, and thus, have higher serendipity.

Pareto Front The quality of the recommended venues with respect to the relevance is ensured by the venue recommendation algorithm employed. However, as the length of the routes could be much shorter than the number of recommended venues, diversity becomes an issue as routes that consist of only venues which are semantically close to each other (e.g., belongs to the same category). Such routes are typically considered as not interesting routes. To prevent this issue, rather than optimizing over a single dimension, MPG computes the Pareto Front of the random walks generated based on their serendipity σ_r, as described above (Eq. 12) and their diversity δ_r [24]. The Pareto Front includes the non-dominated points, which is a set of points that can no longer be improved along one dimension without sacrificing the others, i.e., a route r is part of the Front *iff* $\sigma_r \geq \sigma_{r'}$ and $\delta_r \leq \delta_{r'}$, $\forall r' \in \mathcal{R}$, where $\mathcal{R}$ is the set of the random walks.

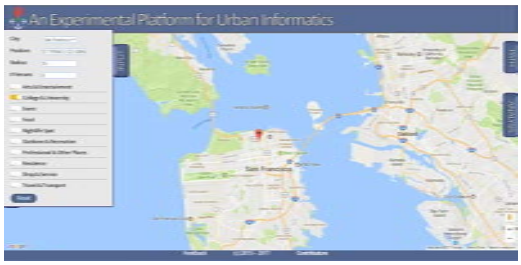


Fig. 4. EPUI Input Panel

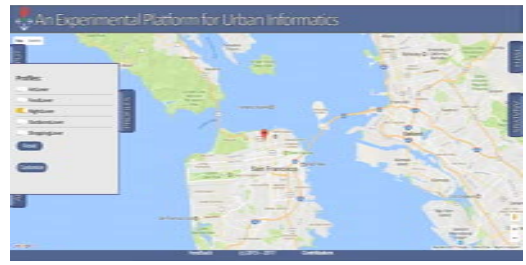


Fig. 5. EPUI Profile Panel

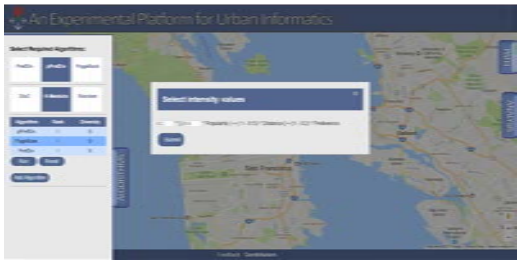


Fig. 6. EPUI Algorithms Panel

Fig. 7. EPUI Display Results

Given the fact that the computed Pareto Front might include a large number of routes, we divide it into n equal parts (through projections on the dimension with the maximum range) where n can be adjusted by the user, and choose a representative route for each part of the front based on the shortest distance to be covered. This ensures our system will provide the user with multiple alternative routes that capture different levels of the trade-off between both serendipity and diversity dimensions.

5 PROTOTYPE IMPLEMENTATION

We have implemented a prototype MPG both as a web service and a mobile app for Android devices. We further developed a testbed, namely, EPUI [2] (Experimental Platform for Urban Informatics) [26] to evaluate the effectiveness of our proposed mobile service MPG and explore our ideas for capturing serendipity.

EPUI consists of a user-friendly front-end interface and an efficient back-end server that incorporates our proposed venues recommendation algorithms as well as other components of MPG. Further, EPUI supports a number of different approaches for the venue and tour recommendations based on the user's current location, which enables the user to compare the recommendations of different recommendation engines by both presenting the recommended venues and tours visually on the map as well as displaying evaluation metrics through summary dashboards.

Specifically, EPUI implements the following well-known venue recommendation algorithms: DisC Diversity [19], K-Medoid, and Random Selection based recommendation approach. In addition to these algorithms, EPUI enables the user to compare the performance of different venue recommendations and route construction approaches, including our proposed MPG service, as well as the user's own customized algorithm to facilitate the research and development in both venue

[2] An introduction video can be found at http://tiny.cc/sf0d7y.

recommendation and route construction. For example, while we have implemented the same diversity scheme for all algorithms, our implementation is flexible and allows for different diversity and indexing schemes that can easily be adjusted by the user.

The front-end of EPUI is constructed from javascript and PHP with the help of Google Maps API for visualizing the results on a map. It communicates with the back-end server through JSON, and currently supports the cities of New York, San Francisco and Pittsburgh. The recommended venues or POIs are numbered and colored to match the number and the color of the algorithm making the recommendation, along with different provisions to drive or walk to those recommendations. The interface consists of five different panels: "Input", "Profile", "Algorithms", "Path" and "Analysis" (Figures 4-6).

The "Input" panel provides the options for the user to specify the inputs that describe the basic information for each query, such as the radial distance they are willing to travel, the number of venues they would like to get returned, and the types of venues they are interested in exploring (Figure 4).

The "Profile" panel provides the options for the user to specify their own preferences by selecting any of the predefined profiles (i.e., *ArtLover, FoodLover, OutdoorsLover*, etc.) (Figure 5), or to customize a selected preference profile by adjusting the values on the corresponding entry of the category tree.

The "Algorithms" panel (Figure 6) allows users to choose, customize and upload the recommendation algorithm(s) that would be involved in the location query. To upload an algorithm, EPUI simply requires the user to providing the name and the corresponding template java program, and then it will be include in the algorithms list along with other existing algorithms. For each algorithm, the user has an option for adjusting the composition of the ranking (Eq. 5) and semantic distance (Eq. 8) function.

The "Analysis" panel visualizes the performance characteristics of the recommended venues from the selected algorithms in tabular form as well as in a scatterplot. The listed characteristics in terms of quality are the relevance score of the recommended venues, their diversity and the radius of gyration for each set of the recommended venues. We also report the run time taken for each algorithm as an indicator of interactivity.

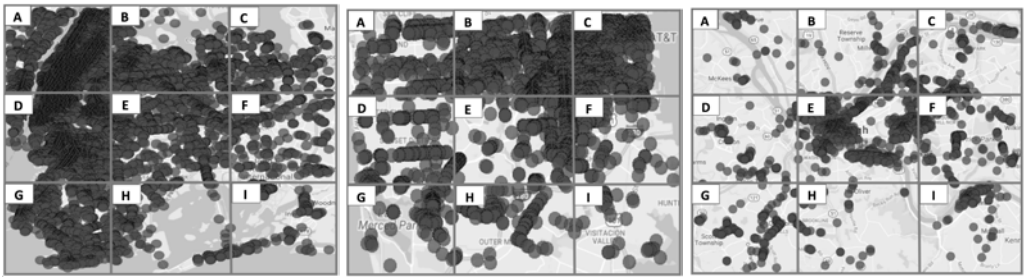
The "Path" panel (Figure 7) allows users to select the construction method of the routes based on sets of recommended venues and a route network graph G. Once a route construction method has been selected, it would determine the visit sequence of each recommended venues. Later the route that connects each venue (according to the specified sequence) would be constructed based on the weights assigned on G. Furthermore, users is able to assign any weights to the edges of G by upload their customized weight modules as well as specify their desired trade-off between weights that are currently assigned to each edge of G. Finally, this panel also includes a statistics table that would display basic informations (e.g., originate, destination) or evaluation measures (e.g., distance, risk, relevance, diversity) according to user's configuration.

6 PERFORMANCE EVALUATION

In this section, we present the detail evaluation of our proposed mobile service `MPG` using our experimental platform `EPUI` and real-life data. Before discussing our findings, we introduce the datasets used in our experimental evaluation.

6.1 Datasets

In our experiments, we used data collected across five months from Foursquare. Foursquare is a major digital, location-based social network where the main interaction among its users is the voluntary sharing of one's whereabouts through *check-ins*. Foursquare has a rich, user-curated



(a) NYC Venues Distribution (b) SF Venues Distribution (c) PIT Venues Distribution

Fig. 8. Spatial visualization of datasets population for an indicative 3x3 partitioning.

venue database through which users can choose to notify their friends of their current location. Furthermore, as mentioned in Section 19, we have used a Word2vec model that is trained on the entire English Wikipedia. Note, this is the only MPG component that requires training data for model training, and since it is used as a semantic distance measure, we did not create a separate English Wikipedia testing data to test the performance of the word vector obtained from Word2vec. In particular, our study utilizes the following information:

Venue database: We used Foursquare's public venue API and queried information for **14,011,045** venues. Each reading has the following tuple format: `<ID, latitude, longitude, # check-ins, # unique users, type>`. The purpose of this dataset is two-fold: (a) obtain a database of all points-of-interest (POIs) in a city, and (b) to obtain information that can be used as a proxy for the quality of a venue (e.g., the number of unique users that have checked-in to the venue or the total number of check-ins). We have queried the Foursquare venue database and have obtained the relevant information for all the venues in New York City (NYC), San Francisco (SF), and Pittsburgh (PIT) (depicted in Figure 8).

Figure 8 shows the population for the three respective cities when splitting into nine equal-width partitions. The total number of venues for each city is i) 471052 in NYC, ii) 73623 in SF, and iii) 33975 in PIT, and the standard deviation among the buckets for each city is i) 80K in NYC, ii) 8.6K in SF, and iii) 5.5K in PIT.

User check-in information: User preferences can be indirectly revealed through their historic visitations (e.g., frequent visits at Chinese restaurants by Pam is a strong signal for her appeal to this cuisine). In order to build realistic user profiles for our evaluations, we used a dataset collected by Cheng et al. [12] that includes geo-tagged, user-generated content from a variety of social media between September 2010 and January 2011. This dataset includes 11,726,632 Foursquare check-ins generated by 188,450 users.

6.2 Evaluation of POIs Recommendation

In this section, we will provide a detailed comparison between our proposed *p*PrefDiv with a number of baseline algorithms (as listed in Table 1) for POIs recommendations.

6.2.1 Methodology. In our evaluation, we have employed as baselines a number of algorithms, including DisC Diversity, which is the state-of-the-art diversification algorithm that optimizes the coverage of the result-set. We also included one well-known clustering algorithm K-Medoids [49], which aims to group a set of data objects into clusters through some distance measure, so each object within a cluster are close to each other and objects outside of the cluster are disclosed to the objects inside the cluster. In our experiment, we have implemented K-Medoids based on [49].

Table 1. MODEL ABBREVIATION

Models	Description
PD(composite)	Uses composite intensity value as the relevance score for *P*refDiv.
pPD(composite)	Uses preference-based intensity value as the relevance score for *p*PrefDiv.
DisC	Only uses the result of DisC Diversity [19] as the final ranking without using *P*refDiv.
K-Medoids	Only uses the result of K-Medoids as the final ranking without using *P*refDiv.
Random	Only uses the randomly selected result as the final ranking without using *P*refDiv.

Since K-Medoids does not capture the relevance in any regard, thus, we improved the performance of K-Medoids in balancing the relevance VS. diversity trade-off by choosing the object with the highest intensity value as the final recommendation from each of its k clusters. This improvement significantly enhances the performance of the K-Medoids with respect to relevance while exhibiting the minimum decrease in diversity. In addition, we also employed the random selection of venues as another baseline to show the effectiveness of our proposed methods.

For all algorithms involved in our experiments, we utilized the semantic distance measure (Eq. 8) as the way to measure the distance between two given points. Table 1 summarizes all models employed in our experiments, and the values of the parameters used are summarized in Table 2.

We ran all our experiments on a computer with Intel Core i7 2.5Ghz CPUs, 16GB Memory, and 512GB SSD and used the Foursquare datasets described in Section 6.1. We created individual Foursquare user profiles as described above and three super-user profiles with more fine-grained preferences by merging the profiles: i) 1000 Foursquare users (Super-user A); ii) 500 Foursquare users (Super-user B); and iii) 350 Foursquare users (Super-user C). Note that the Super-user A contains both Super-user B and Super-user C, where Super-user B and Super-user C are two disjoint sets of profiles.

6.2.2 Experimental Evaluation Metrics. In our experimental evaluation, we use three well-known metrics: *Normalized Relevance* [54], *Average Similarity Distance* (based on semantic distance measure, Eq. 8), and *Coverage* [18] .

Definition 7. *(*Normalized Relevance*). Let O be a set of venues and $O_k^* \subseteq O$ such that $|O_k^*| = k$. The Normalized Relevance of O_k^* is defined as the total relevance score of O_k^* over the sum of top-k highest relevance scores of O.*

In our experiments, Normalized Relevance is represented through two type of intensity values, *Normalized Composite Intensity Value* (NCI) and *Normalized Preference-based Intensity Value* (NPI).

We defined Normalized Composite Intensity Value as:

$$NCI(x) = \frac{CI(x)}{\text{Argmax}_{i:O}\, CI(i)} \tag{13}$$

where $CI(x)$ is the composited intensity value (Eq. 5) of a venue x, and O is the set of all venues within the currently range query q.

Normalized Preference-based Intensity Value is defined as:

$$NPI(x) = \frac{PI(x)}{\text{Argmax}_{i:O}\, PI(i)} \tag{14}$$

where $PI(x)$ is the Preference-based intensity value (Eq. 2) of a venue x, and O is the set of all venues within the currently range query q.

The reason to employ NCI and NPI as the relevance measure is to show how much the relevance has been preserved with respect to the given utility function. By comparing the aggregated intensity

Table 2. PARAMETER CONFIGURATION

Parameters	Value
λ	0.5
μ	0.7
γ	0.7
α	0.5
ω	0.5
ξ	50
A	0.3
ϱ	0.7
K	10 - 50
Radius	1.5 km
Number of Locations	15, 50
Number of Categories of POIs Selected	5, 10

score of the recommended representative subset O_k^* with the k most relevant item (i.e., item with highest intensity according to the given utility function) of the original set O, one can clearly know the degree of the intensity value being preserved in the current representative subset O_k^*.

Definition 8. *(*Average Similarity Distance*) Let O be a set of venues, the average similarity distance of O represents the average of the pairwise distances of the venues in O.*

Due to the fact that two distinct instances in our data set may be semantically duplicate venues (e.g., two difference Starbucks), our experiments normalize Average Similarity Distance (ASD) to take into consideration that different methods may return as a result of a list of venues with semantic duplicates rather than a set and expressed as Redundancy Normalized Pairwise Distance (RNPD):

$$RNPD(L) = (1 - \frac{|Unique(L)|}{|L|}) * ASD \tag{15}$$

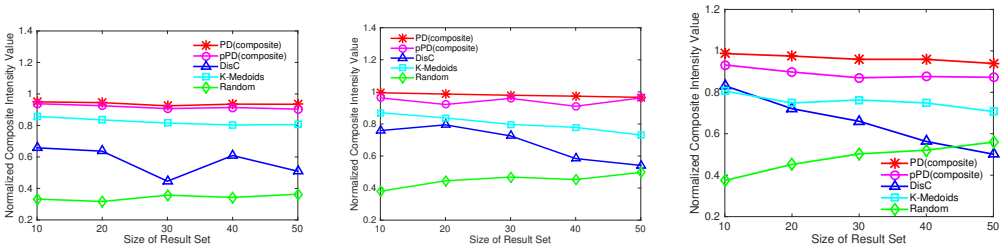
where $Unique(O)$ represents O without duplicates.

The reason to employ RNPD as the diversity measure is to show how each item contained in a given representative set O_k^* is different compared to the rest of the items in O_k^*. The most intuitive way to do so is to measure the pairwise distance (with a given distance measure) between each pair of items, which is essentially the ASD defined above. Since recommending unique venues is important in the case of venue recommendation, we included a penalty (i.e., $(1 - \frac{|Unique(L)|}{|L|})$) for recommending two or more duplicate venues in the recommendation result set, which forms the Redundancy Normalized Pairwise Distance (RNPD).

Definition 9. *(*Coverage*) Let O be a set of venues, $O_k^* \subseteq O$ such that $|O_k^*| = k$ and $S \subseteq O$ be defined as the union of $sim_\varrho(v_i, O)$ for all $v_i \in O_k^*$. The coverage of a subset O_k^* is defined as the percentage of venues in S over the total number of venues in O, i.e., $|S|/|O|$.*

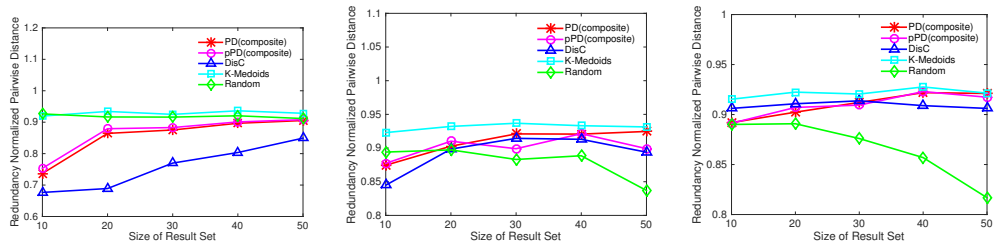
As discussed with more detail in [18], the reason to use coverage for evaluating the degree of coverage provided by a representative subset O_k^* is simply as it naturally represents the percentage of the item has been covered by O_k^* with respect to a given radius ϱ.

6.2.3 Experimental Results. In this section, we present the evaluation of all models in our experiments. Specifically, we will show a comprehensive set of comparisons between our proposed *p*PrefDiv and other alternative algorithms that show the capability of *p*PrefDiv in handling the POIs recommendation.



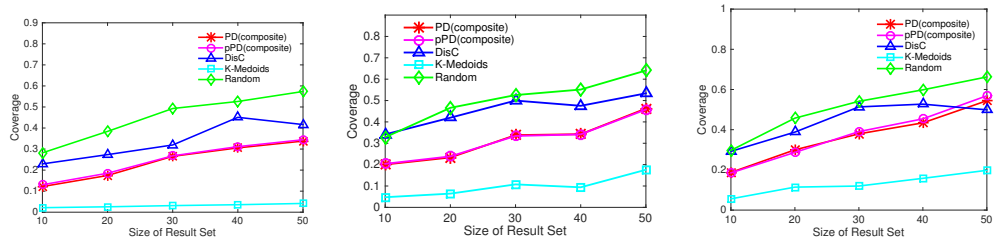
(a) Relevance: NCI (b) Relevance: NCI (c) Relevance: NCI

Fig. 9. Compare the relevance of different recommendation algorithms for NYC, SF and PIT (Super-user A).



(a) Diversity: RNPD (b) Diversity: RNPD (c) Diversity: RNPD

Fig. 10. Compare the diversity of different recommendation algorithms for NYC, SF and PIT (Super-user A).

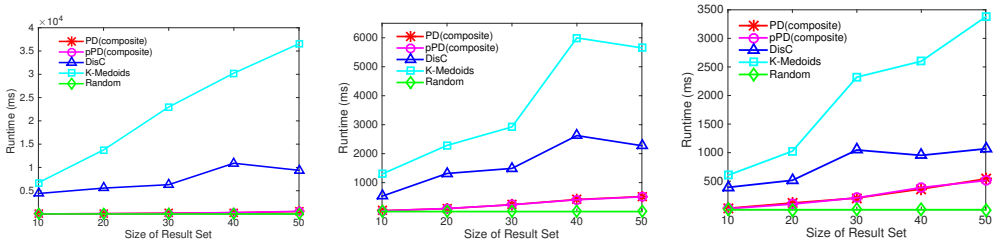


(a) Diversity: Coverage (b) Diversity: Coverage (c) Diversity: Coverage

Fig. 11. Compare the Coverage of different recommendation algorithms for NYC, SF and PIT (Super-user A).

As we have demonstrated previously, PD(composite) has exhibited the best performance when compared to other PrefDiv based models. We extend the same composition of the intensity value to the non-deterministic variance pPrefDiv (1), which is a variance of PrefDiv that incorporates the serendipity. Further, we compare PD(composite) (i.e., PrefDiv) and pPD(composite) (i.e., pPrefDiv) to other well-known algorithms to show the performance of both PrefDiv and pPrefDiv based recommendation engine when compared to other existing alternatives.

Figures 9a, 9b, and 9c shown the average normalized composite intensity value (NCI) of all models for New York City (NYC), San Fransisco (SF), and Pittsburgh (PIT). In these results, both PrefDiv and pPrefDiv have constantly outperformed other alternatives in all three cities. In particular, PrefDiv and pPrefDiv have outperformed random by up to 263%, DisC by up to 93%, and K-Medoids by up to 36%. Among all involved algorithms, K-Medoids has Performed second best after the PrefDiv



(a) Runtime(ms) (b) Runtime(ms) (c) Runtime(ms)

Fig. 12. Compare the Runtime of different recommendation algorithms for different cities (Super-user A).

and pPrefDiv, which is due to the reasons that we have implemented K-Medoids in a way that enhanced its ability to capture the relevance. Random achieves the worst performance because the number of highly preferred venues is much less than the non-preferred venues; thus, blindly selecting venues would not deliver a good performance in terms of relevance.

Figures 10a, 10b and 10c illustrated the Redundancy Normalized Pairwise Distance (RNPD) of all models for all three cities. Here we have seen that all algorithms have achieved overall better performance for RNPD when compare to NCI. This is as expected since all involved algorithms except Random have been optimized towards the diversity. We noticed that Random also performs very well with respect to RNPD, which can be explained as follows. Since the number of neighbors of given venues (in terms of semantic distance) is always going to be much less than non-neighbors, those it is much easier for Random to generate a set of results that are semantically diversity than to generate a set of semantically similar results. Here, we have seen that although PrefDiv and pPrefDiv did not outperform all other alternatives, however, the price that both PrefDiv and pPrefDiv have paid to achieve the high relevance score (in terms of relevance and diversity trade-off) is extremely small. When compared to the K-Medoids, which achieves the highest performance in semantic diversity, PrefDiv and pPrefDiv are still just slightly behind.

Figures 11a, 11b and 11c demonstrate the performance of Coverage of all models. In this set of results, we have seen that DisC and Random achieve the best performance in terms of coverage. This is as expected as the DisC utilized a greedy-based approach that is specifically optimized towards the coverage. As for the random, since it builds the result-set by evenly sample the venues across all candidate venues in a given spatial query, thus, it has a high likelihood to achieve excellent coverage when dealing with datasets where data objects are distributed evenly across space. In the case of urban datasets, since semantically closer venues are more or less distributed evenly across the city to prevent excessive competition, thus, enable random to achieve high coverage. However, the difference between PrefDiv (as well as pPrefDiv) with random and DisC is not significant. Considering the performance advantage that PrefDiv and pPrefDiv have gained in the relevance, therefore both PrefDiv and pPrefDiv still served as a much better venue recommendation engine than other alternatives.

Figures 12a, 12b and 12c demonstrate the runtime of all models for New York City, San Fransisco, and Pittsburgh. Since the POIs recommendation involves humans in the loop, it is extremely important for the system to maintain an interactive response time. To achieve such a goal, it requires the algorithm to be as efficient as possible. From these experiments, we can see that both PrefDiv and pPrefDiv have outperformed DisC and K-Medoids in runtime by orders of magnitudes. The only alternative that is faster than our PrefDiv and pPrefDiv is the Random method; this is

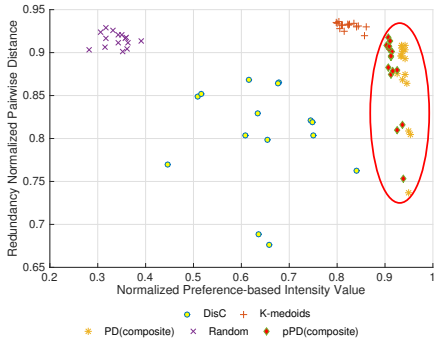


Fig. 13. Relevance vs. Diversity (NYC)

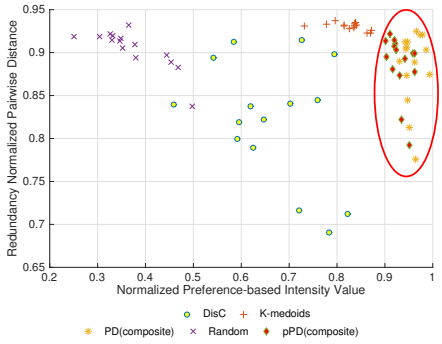


Fig. 14. Relevance vs. Diversity (SF)

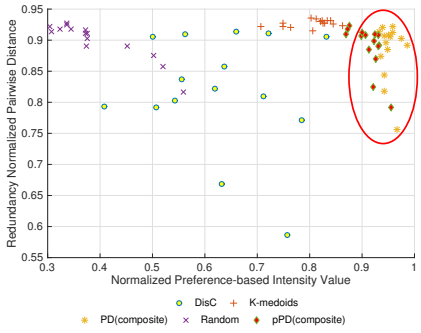


Fig. 15. Relevance vs. Diversity (PIT)

because it does not require any complex computations. Such a huge difference in efficiency has undoubtedly made the PreDiv and pPrefDiv to be better choices for POIs recommendations.

Figures 13, 14 and 15 present three scatter plots from New York City, San Fransisco, and Pittsburgh respectively that showed the capability of each algorithm in balancing the trade-off between relevance and diversity. Each point in these three plots also represents an average of 15 different locations queries. For each algorithm, we have drawn 15 points based on three super-user profiles. Thus, each super-user profile is responsible for five points, and each of these five points corresponds to a different result-set size. As cycled in red, these results have clearly shown that both our PrefDiv and pPrefDiv have delivered the best performance in terms of both relevance and diversity as the points generated by both algorithms are located on the most upper right corner of the plot.

From our empirical studies, we have observed that six of the eight tunable parameters listed in Table 2 are task-dependent parameters (i.e., λ, μ, γ, α, ω, ξ), which means that they are much more sensitive to different recommendation tasks (e.g., recommend venues/POIs, recommend movies, or recommend documents) than different users or datasets. For this reason, the provided default values for each of these six parameters in Table 2 can serve as a good starting point for any real-world venues/POIs recommendation tasks.

The other two tunable system parameters (i.e., A, ϱ), which are required by the PrefDiv algorithm, are user and datasets dependent. To address this issue, we are currently investigating methods to facilitate the tuning of these parameters and automatically compute the optimal value for these

two parameters based on the given context [22]. These new optimizations of PrefDiv would be incorporated seamlessly into MPG as well as any other application that utilizes PrefDiv.

6.3 Evaluations of Route Construction

We further study the cost of incorporating serendipity through the random walk in route recommendations and evaluate the performance of different route construction schemes.

6.3.1 Routing Schemes. In particular, we propose four specific schemes to combine our random walk based routing method with our venue recommendation engines. These schemes are: i) PrefDiv+RandomWalk; ii) pPrefDiv+RandomWalk; iii) pPrefDiv+ShortestDistance; and iv) pPrefDiv+HighestRelevence.

PrefDiv+RandomWalk: This scheme aims to incorporate serendipity in route recommendation by combining a random walk based non-deterministic routing method with the deterministic venue recommendation engine PrefDiv. In PrefDiv+RandomWalk, 50 random walks would be performed based on the result-set of PrefDiv to generate 50 initial routes, each of length K'. Later, one random route with the best overall performance in terms of the average composited intensity value and travel distance is selected from the Pareto Front (constructed using all 50 initial routes) as the final recommendation. Therefore, this scheme combines deterministic venues recommendation with the non-deterministic route recommendation.

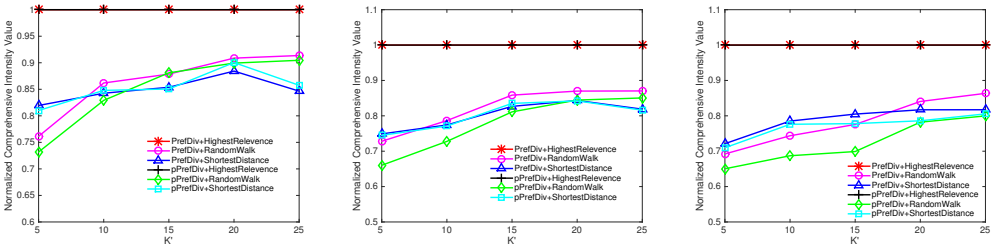
pPrefDiv+RandomWalk: The pPrefDiv+RandomWalk scheme is similar to the previous scheme except that the non-deterministic algorithm pPrefDiv has been employed as the recommendation engine to achieve serendipity in both venue and route recommendation level. Thus, this scheme combines non-deterministic venues recommendation with the non-deterministic route recommendation.

pPrefDiv+ShortestDistance: For this scheme, serendipity is incorporated only in venue recommendations. As such, the shortest distance-based routing method is employed to replace the random walk based routing. Particularly, once the set of recommendations have been generated by pPrefDiv, the shortest route with size K' is constructed from these recommended venues, which minimize the psychical distance according to the Euclidean distance.

pPrefDiv+HighestRelevence: This scheme is the same as the pPrefDiv+ShortestDistance scheme, except that the route is constructed according to the ranking of the venue's average composite intensity value. Similar to the above scheme, this scheme combines deterministic venues recommendation with the non-deterministic route recommendation.

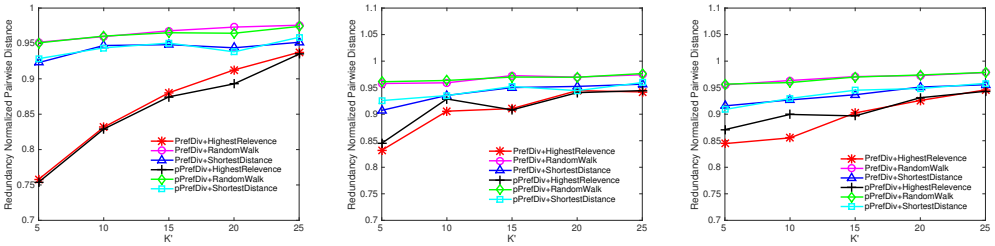
Baseline schemes: To evaluate the trade-off between relevance, travel distance and serendipity, for a given set of recommended venues V, st. $|V| \leq k$ and a length of the route K', st $K' \leq K$, we introduce two additional deterministic routing methods, i) *highest relevance routing*, and ii) *shortest distance routing*. The former, introduced in Section 4.2, takes the given set of recommended venues and forms a route recommendation by visiting the top K' venues with the highest composite intensity value according to their ranking of composited intensity value (i.e., normalized relevance). The latter constructs the shortest travel distance route of size K' from V in a greedy fashion. Further, we combine these two routing methods with PrefDiv to form as baseline two deterministic schemes, namely, *PrefDiv+HighestRelevence* and *PrefDiv+ShortestDistance*.

6.3.2 Evaluation Metrics. To measure such costs we use the *normalized relevance* (Def. 7) of all venues contained in the route, the *average similarly distance* (Def. 8) of each pair of venues in the route, and the *total physical distance* that user has to travel by following the route. As explained in Sec 4.2, to capture serendipity, we use the *degree of serendipity* (Eq. 12) of each route generated by MPG.



(a) Relevance: NCI (b) Relevance: NCI (c) Relevance: NCI

Fig. 16. Compare relevance of different routing algorithms for NYC, SF and PIT with Super-user A.



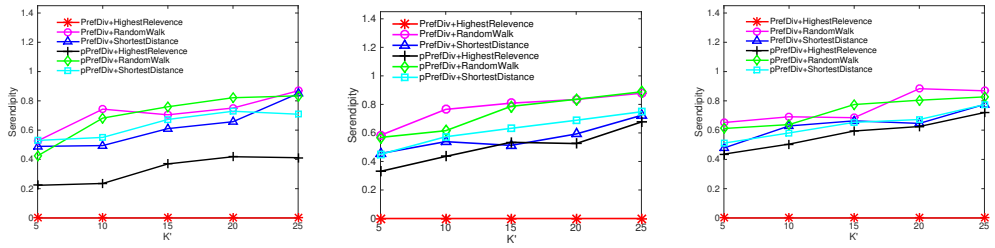
(a) Diversity: RNPD (b) Diversity: RNPD (c) Diversity: RNPD

Fig. 17. Compare diversity of different routing algorithms for NYC, SF and PIT with Super-user A.

Similar to Section 6.2.2, we have used the average composite intensity value as a measure of Normalized Relevance and Redundancy Normalized Pairwise Distance for Average Similarity Distance.

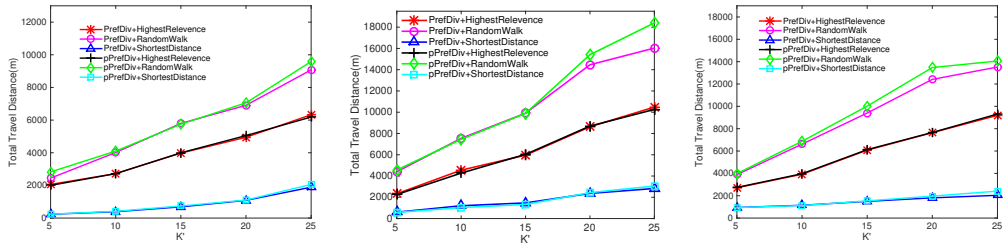
6.3.3 Experimental Results. We now present the evaluation of all route recommendation schemes. Figures 16a, 16b and 16c illustrates the average normalized composite intensity value (i.e., relevance score) of routes generated from each scheme for New York City, San Francisco, and Pittsburgh, respectively. In this set of experiments, all schemes are normalized with respect to the PrefDiv+HighestRelevence, which is the upper bound for the relevance scores. From these experiments, we observed that both random walk and shortest distance-based schemes perform comparably with respect to the relevance for all three cities. In addition, the serendipity-based scheme (i.e., PrefDiv+RandomWalk and pPrefDiv+RandomWalk) still maintained, on average, 82% of the composited intensity value when compared to the upper bound. This indicates that the serendipity introduced by employ random walk as a routing algorithm did not significantly affect the relevance of the routing. This is expected since all schemes constructed their routes based on the highly relevant recommendations produced by our venue recommendation algorithms (i.e., PrefDiv and pPrefDiv).

Figures 17a, 17b and 17c show the Average Similarity Distance between each schemes. The results presented here are normalized with respect to the upper bound of the average redundancy normalized pairwise distance for each range query and result-set size involved in the experiment. From these results, PrefDiv+RandomWalk and pPrefDiv+RandomWalk demonstrate the highest performance in maintaining the diversity of the generated routes. This is due to the reason both HighestRelevence and ShortestDistance optimize towards a certain aspect of the result, such as maximizing relevance or minimizing travel distance, these optimizations often come at



(a) Serendipity (b) Serendipity (c) Serendipity

Fig. 18. Compare serendipity of different routing algorithms for NYC, SF and PIT with Super-user A.



(a) Travel Distance(m) (b) Travel Distance(m) (c) Travel Distance(m)

Fig. 19. Compare travel distance of different routing algorithms for NYC, SF and PIT with Super-user A.

the price of sacrificing other contradictory aspects. However, the random walk does not suffer from this issue as it is not optimized towards any specific aspects. Amount all schemes, both PrefDiv+HighestRelevence and pPrefDiv+HighestRelevence achieve the worst with respect to the diversity, which is expected, since venues that are highly relevant to a user's preference have a higher chance to be semantically close venues.

Figures 18a, 18b and 18c demonstrate the performance with respect to the serendipity for each schemes. These results indicated that the random walk based schemes achieved the highest serendipity. One interesting observation from these experiments is that incorporating the serendipity in either the venue recommendation or the route construction alone is still able to introduce a considerable amount of increase in the serendipity of the constructed routes. Further, incorporating the serendipity into both venue recommendation and route construction (i.e., through pPrefDiv+RandomWalk) does not achieve a noticeable improvement over the PrefDiv+RandomWalk. This shows that the randomness incorporated in the venue recommendation and route construction does not monotonically increase with respect to the amount of randomness of the scheme.

Figures 19a, 19b and 19c shown the average travel distance of the routes produced by each scheme. From these figures, we observed that compared to the shortest distance-based scheme, the random walk based scheme does increase the travel distance. However, exploring cities with shortest routes does not necessarily indicate more interestingness. In fact, users may benefit more from longer travel distance when exploring a city, as it helps them to discover more places around the city. We also noticed that PrefDiv+HighestRelevence and pPrefDiv+HighestRelevence had produced routes that are shorter than random walk based schemes. This is due to the fact that the popularity of a venue is considered as part of it's composited intensity value, as popular venues tend

to be co-located with each other in a relatively small region (e.g., downtown, shopping complex, etc), thus, reduced the total travel distance for relevance focused routing schemes.

7 CONCLUSIONS AND FUTURE WORK

In this paper we proposed and designed MPG, a mobile service that aims at three objectives: i) to provide a set of diverse and surprisingly interesting venue recommendations that are better aligned with user preferences; ii) to achieve sub-second interactive runtime in providing venue recommendations with large dataset; and iii) to form routes within venues recommended by our proposed venue recommendation engine while still maintaining balance between *Relevance, Diversity,* and *Serendipity*.

We achieved our first objective through the design of MPG, which incorporates the user habits, the reach willing to cover, the types of venues interested in exploring, the popularity, the diversity of venues with multiple venue recommendations engines. The second objective is achieved with the capability of *p*PrefDiv to efficiently produce the venue recommendations and the use of multiple optimizations (e.g., efficient query indexing structures, hashing). The third objective is achieved by proposing a set of routing schemes each target at different objectives, each routing scheme is a unique combination of one venue recommendation algorithms (e.g., *P*refDiv or *p*PrefDiv) and one routing algorithms (e.g., random walk, shortest distance) that incorporates difference levels of serendipity while preserving the relevance and diversity aspects of the routes.

To evaluate our proposed mobile service MPG, we designed the EPUI (Experimental Platform of Urban Informatics) testbed, which provides user-friendly interfaces for both end-users and researchers. Our experimental evaluation through EPUI with large Foursquare datasets. Our proposed *P*refDiv and *p*PrefDiv algorithms have enabled sub-second interactive response time while still maintaining excellent balancing between Relevance and Diversity.

Recently, MPG and EPUI were extended to support route construction while optimizing multiple simultaneous objectives (e.g., diversity, safety, happiness, serendipity) [25]. Furthermore, the extended EPUI also serves as a testbed for exploring and evaluating venues and route recommendation solutions by enabling researchers to upload their own algorithms and compare them to well-known algorithms using different performance metrics.

ACKNOWLEDGEMENT

We would like to thank Ameya Daphalapurkar, Manali Shimpi, Chihao Sun and Darpun Kohli for their help in the development of EPUI as part of their MS and First Experience in Research program, as well as the anonymous reviewers. The first three authors' research work in this paper was partially supported by the US National Science Foundation CPS-1739413 and the National Institutes of Health U01HL137159 awards. The fourth author's research work in this paper was supported by the Alexander von Humboldt-Foundation, Germany and by the Cyprus RPF EnterCY (INTEGRATED/0916/0020). The fifth author's research work in this paper was supported by the UAE University grant G00003352. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Science Foundation and the National Institutes of Health."

REFERENCES

[1] 2015. Foursquare Category Hierarchy. https://developer.foursquare.com/categorytree.
[2] 2019. The Nationality Rooms. https://en.wikipedia.org/wiki/Nationality_Rooms.
[3] Making P Adamopoulos. 2016. *Unexpectedness and Non-Obviousness in Recommendation Technologies and Their Impact on Consumer Decision.* Ph.D. Dissertation. New York University.
[4] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *WSDM*. 5–14.

[5] Liliana Ardissono, Anna Goy, Giovanna Petrone, Marino Segnan, and Pietro Torasso. 2003. Intrigue: Personalized Recommendation of Tourist Attractions for Desktop and Hand Held Devices. *Applied Artificial Intelligence* 17, 8-9 (2003).

[6] S. Basu Roy, G. Das, S. Amer-Yahia, and C. Yu. 2011. Interactive itinerary planning. In *2011 IEEE 27th International Conference on Data Engineering*. 15–26.

[7] I. Benouaret and D. Lenne. 2016. A Composite Recommendation System for Planning Tourist Visits. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. 626–631.

[8] Betim Berjani and Thorsten Strufe. 2011. A Recommendation System for Spots in Location-based Online Social Networks. In *Proceedings of the 4th Workshop on Social Network Systems*. Article 4, 6 pages.

[9] Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *SIGIR*. 335–336.

[10] L. Elisa Celis, Damian Straszak, and Nisheeth K. Vishnoi. 2018. Ranking with Fairness Constraints. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. 28:1–28:15.

[11] Zaiben Chen, Heng Tao Shen, Xiaofang Zhou, Yu Zheng, and Xing Xie. 2010. Searching Trajectories by Locations: An Efficiency Study. In *SIGMOD*. 255–266.

[12] Z. Cheng, J. Caverlee, K. Lee, and D.Z. Sui. 2011. Exploring Millions of Footprints in Location Sharing Services. In *ICWSM*.

[13] Paolo Ciaccia, Marco Patella, and Pavel Zezula. 1997. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. In *VLDB*. 426–435.

[14] Paolo Ciaccia, Marco Patella, and Pavel Zezula. 1998. A Cost Model for Similarity Queries in Metric Spaces. In *Symposium on Principles of Database Systems*. 59–68.

[15] Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan BÃijttche, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *SIGIR*. 659–666.

[16] Munmun De Choudhury, Moran Feldman, Sihem Amer-Yahia, Nadav Golbandi, Ronny Lempel, and Cong Yu. 2010. Automatic Construction of Travel Itineraries Using Social Breadcrumbs. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia*. 35–44.

[17] Marina Drosou and Evaggelia Pitoura. 2012. Dynamic diversification of continuous data. In *EDBT*. 216–227.

[18] Marina Drosou and Evaggelia Pitoura. 2012. Result Diversification based on Dissimilarity and Coverage. In *VLDB*. 13–24.

[19] Marina Drosou and Evaggelia Pitoura. 2015. Multiple Radii DisC Diversity: Result Diversification Based on Dissimilarity and Coverage. *ACM TODS* 40, 1 (2015).

[20] Simon Dunstall, Mark E. T. Horn, Philip Kilby, Mohan Krishnamoorthy, Bowie Owens, David Sier, and Sylvie Thiébaux. 2003. An Automated Itinerary Planning System for Holiday Travel. *J. of IT & Tourism* 6, 3, 195–210.

[21] Esther Galbrun, Konstantinos Pelechrinis, and Evimaria Terzi. 2016. Urban navigation beyond shortest route: The case of safe paths. *Information Systems* 57 (2016), 160–171.

[22] Xiaoyu Ge and Panos K. Chrysanthis. 2020. PrefDiv: Efficient Algorithms for Effective Top-k Result Diversification. In *EDBT*. 335–346.

[23] Xiaoyu Ge, Panos K. Chrysanthis, and Alexandros Labrinidis. 2015. Preferential Diversity. In *Proceedings of the Second International Workshop on Exploratory Search in Databases and the Web (ExploreDB '15)*. 9–14.

[24] Xiaoyu Ge, Panos K. Chrysanthis, and Konstantinos Pelechrinis. 2016. MPG: Not So Random Exploration of a City. In *IEEE 17th International Conference on Mobile Data Management, MDM 2016, Porto, Portugal, June 13-16, 2016*. 72–81.

[25] Xiaoyu Ge, Panos K. Chrysanthis, Konstantinos Pelechrinis, and Demetrios Zeinalipour-Yazti. 2018. EPUI: Experimental Platform for Urban Informatics. In *SIGMOD*. 1761–1764.

[26] Xiaoyu Ge, Ameya Daphalapurkar, Manali Shimpi, Darpun Kohli, Konstantinos Pelechrinis, Panos K. Chrysanthis, and Demetrios Zeinalipour-Yazti. 2017. Data-Driven Serendipity Navigation in Urban Places. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. 2501–2504.

[27] Roxana Gheorghiu, Alexandros Labrinidis, and Panos K. Chrysanthis. 2014. A user-friendly framework for database preferences. In *CollaborativeCom*. 205–214.

[28] Aristides Gionis, Theodoros Lappas, Konstantinos Pelechrinis, and Evimaria Terzi. 2014. Customized Tour Recommendations in Urban Areas. In *WSDM*. 313–322.

[29] Jean-Benoit Griesner, Talel Abdessalem, and Hubert Naacke. 2015. POI Recommendation: Towards Fused Matrix Factorization with Geographical and Temporal Influences. In *RecSys*. 301–304.

[30] Zaeem Hussain, Hina A. Khan, and Mohamed A. Sharaf. 2015. Diversifying with Few Regrets, But too Few to Mention. In *ExploreDB*. 27–32.

[31] Marius Kaminskas and Derek Bridge. 2016. Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems. *ACM Trans. Interact. Intell. Syst.* 7, 1 (2016), 2:1–2:42.

[32] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *RecSys*. 79–96.

[33] Hina A. Khan, Marina Drosou, and Mohamed A. Sharaf. 2013. Scalable diversification of multiple search results. In *CIKM*. 775–780.

[34] Hina A. Khan and Mohamed A. Sharaf. 2015. Progressive diversification for column-based data exploration platforms. In *ICDE*. 327–338.

[35] Hina A. Khan and Mohamed A. Sharaf. 2017. Model-Based Diversification for Sequential Exploratory Queries. *Data Science and Engineering* 2, 2 (2017), 151–168.

[36] Hina A. Khan, Mohamed A. Sharaf, and Abdullah Albarrak. 2014. DivIDE: efficient diversification for interactive data exploration. In *SSDBM*. 15:1–15:12.
[37] Werner Kiessling and Gerhard Kostler. 2002. Preference SQL: design, implementation, experiences. In *VLDB*. 990–1001.
[38] Jinyoung Kim, Hyungjin Kim, and Jung-hee Ryu. 2009. TripTip: A Trip Planning Service with Tag-based Recommendation. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*. 3467–3472.
[39] Denis Kotkov, Joseph A Konstan, Qian Zhao, and Jari Veijalainen. 2018. Investigating serendipity in recommender systems based on real user feedback. In *SAC*. 1341–1350.
[40] Takeshi Kurashima, Tomoharu Iwata, Go Irie, and Ko Fujimura. 2010. Travel Route Recommendation Using Geotags in Photo Sharing Sites. In *CIKM*. 579–588.
[41] D.M. Levinson. 2016. A random walk down Main Street. In *Journal of Land Use, Mobility and Environment*. 31–40. Issue 2.
[42] Defu Lian, Cong Zhao, Guangzhong Sun Xing Xie, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*. 831–840.
[43] Qiuxia Lu, Tianqi Chen, Weinan Zhang, Diyi Yang, and Yong Yu. 2012. Serendipitous Personalized Ranking for Top-N Recommendation. In *WI-IAT*. 258–265.
[44] Ziyu Lu, Hao Wang, Nikos Mamoulis, Wenting Tu, and David W. Cheung. 2017. Personalized Location Recommendation by Aggregating Multiple Recommenders in Diversity. *Geoinformatica* 21, 3, 459–484.
[45] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *NIPS*. 3111–3119.
[46] Makoto Nakatsuji, Yasuhiro Fujiwara, Akimichi Tanaka, Toshio Uchiyama, Ko Fujimura, and Toru Ishida. 2010. Classical music for rock fans?: novel recommendations for expanding user interests. In *CIKM*. 949–958.
[47] Anastasios Noulas, Salvatore Scellato, Neal Lathia, and Cecilia Mascolo. 2012. A Random Walk Around the City: New Venue Recommendation in Location-Based Social Networks. In *SOCIALCOM-PASSAT*. 144–153.
[48] Debmalya Panigrahi, Atish Das Sarma, Gagan Aggarwal, and Andrew Tomkins. 2012. Online selection of diverse results. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*. 263–272.
[49] Hae-Sang Park and Chi-Hyuck Jun. 2009. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications* 36, 2 (2009), 3336–3341.
[50] Lu Qin, Jeffrey Xu Yu, and Lijun Chang. 2012. Diversifying Top-K Results. In *VLDB*. 1124–1135.
[51] Daniele Quercia, Rossano Schifanella, and Luca Maria Aiello. 2014. The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city. In *the 25th ACM conference on Hypertext and social media*. ACM, 116–125.
[52] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). 926–934.
[53] Kostas Stefanidis, Georgia Koutrika, and Evaggelia Pitoura. 2011. A survey on representation, composition and application of preferences in database systems. *ACM TODS* 36, 19 (2011).
[54] Hanghang Tong, Jingrui He, Zhen Wen, Ravi Konuru, and Ching-Yung Lin. 2011. Diversified ranking on large graphs: an optimization viewpoint. In *KDD*. 1028–1036.
[55] Ling-Yin Wei, Yu Zheng, and Wen-Chih Peng. 2012. Constructing Popular Routes from Uncertain Trajectories. In *KDD*. 195–203.
[56] Hongfei Xu, Yu Gu, Jianzhong Qi, Jiayuan He, and Ge Yu. 2019. Diversifying Top-k Routes with Spatial Constraints. *J. Comput. Sci. Technol.* 34, 4 (2019), 818–838.
[57] Mao Ye, Peifeng Yin, and Wang-Chien Lee. 2010. Location Recommendation for Location-based Social Networks. In *GIS*. 458–461.
[58] Hyoseok Yoon, Yu Zheng, Xing Xie, and Woontack Woo. 2012. Social Itinerary Recommendation from User-generated Digital Trails. *Personal Ubiquitous Comput.* 16, 5, 469–484.
[59] Cong Yu, Laks Lakshmanan, and Sihem Amer-Yahia. 2009. It Takes Variety to Make a World: Diversification in Recommender Systems. In *EDBT*. 368–378.
[60] Chenyi Zhang, Hongwei Liang, and Ke Wang. 2016. Trip Recommendation Meets Real-World Constraints: POI Availability, Diversity, and Traveling Time Uncertainty. *ACM Trans. Inf. Syst.* 35, 1 (Sept. 2016), 5:1–5:28.
[61] Zhu Zhang, Daniel Zeng, Ahmed Abbasi, Jing Peng, and Xiaolong Zheng. 2013. A Random Walk Model for Item Recommendation in Social Tagging Systems. *ACM Transactions on Management Information Systems* 4, 8 (2013).
[62] Xiaofei Zhu, Jiafeng Guo, Xueqi Cheng, Pan Du, and Huawei Shen. 2011. A unified framework for recommending diverse and relevant queries. In *WWW*. 37–46.
[63] Cai-Nicolas Ziegler, Joseph A. Konstan Sean M. McNee, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *WWW*. 22–32.