Exploration of Interesting Dense Regions on Spatial Data

Behrooz Omidvar-Tehrani
NAVER LABS Europe (France)
behrooz.omidvar-tehrani@naverlabs.com

Plácido A. Souza Neto
Federal Institute of Rio Grande do Norte (Brazil)
placido.neto@ifrn.edu.br

Francisco B. Silva Júnior
Federal University of Rio Grande do Sul (Brazil)
fbsjunior@inf.ufrgs.br

Felipe F. Pontes
Federal Institute of Rio Grande do Norte (Brazil)
freire.pontes@academico.ifrn.edu.br

ABSTRACT

Nowadays, spatial data are ubiquitous in various fields of science, such as transportation and smart city management. A recent research direction in analyzing spatial data is to provide means for "exploratory analysis" of such data where users are guided towards interesting options in consecutive analysis iterations. Typically, the guidance component learns user's preferences using his/her explicit feedback, e.g., picking a spatial point of interest (POI) or selecting a region of interest. However, it is often the case that users forget or don't feel necessary to explicitly express their feedback in what they find interesting. Our approach captures implicit feedback on spatial data. The approach consists of observing mouse moves (as a means of user's interaction) to discover interesting spatial regions with dense mouse hovers. In this paper, we define, formalize, and explore Interesting Dense Regions (IDRs) which capture preferences of users, in order to automatically find interesting spatial highlights. Our approach involves a polygon-based abstraction layer for capturing preferences. Using these IDRs, we highlight POIs to guide users in the analysis process. We discuss the efficiency and effectiveness of our approach through realistic examples and experiments on Airbnb and Yelp datasets.

1 INTRODUCTION

Nowadays, there has been a meteoric rise in the generation of spatial datasets in various fields of science, such as transportation, lodging services, and smart city management [1, 2]. As each record in spatial data represents an activity in a precise geographical location, analyzing such data enables discoveries grounded on facts. Typically, spatial data analysis begins with an imprecise question in the mind of the user, i.e., *exploratory analysis*. The user requires to go through several trial-and-error iterations to improve his/her understanding of the spatial data and gain insights. Each iteration involves visualizing a subset of data on geographical maps using an off-the-shelf product (e.g., Tableau[1], Exhibit[2], Spotfire[3]) where the user can investigate on different parts of the visualization by zooming in/out and panning.

Spatial data are often voluminous. Hence the focus in the literature of spatial data analysis is on "efficiency", i.e., enabling fluid means of navigation in spatial data to facilitate the exploratory analysis. The common approach is to design pre-computed indexes which enable efficient retrieval of spatial data (e.g., [3, 4]). However, there has been less attention to the "value" derived from spatial data. Despite the decent progress on the efficiency front, a user may easily get lost in the plethora of geographical points of interest (POIs), mainly due to two following reasons:

- In an exploratory context, the user doesn't know a priori what to investigate next.
- Moreover, he/she may easily get distracted and miss interesting POIs by visual clutter caused by huge overlaps among POIs.

The main drawback of the traditional analysis models is that the user has a *passive role* in the process. In other words, the user's feedback (i.e., his/her likes and dislikes) is ignored and only the input query (i.e., his/her explicit request) is served. In case feedback is incorporated, the process can be more directed towards user's interests where his/her partial needs can be served earlier in the process [5]. In this paper, we advocate for a "guidance layer" on top of the raw visualization of spatial data to enable users know *"what to see next"*. This guidance should be a function of user feedback: the system should return options similar to what the user has already appreciated.

Various approaches in the literature propose methodologies to incorporate user's feedback in the exploration process of spatial data [6–10]. Typically, feedback is considered as a function which is triggered by any user's action on the map. The action can be "selecting a POI", "moving to a region", "asking for more details", etc. The function then updates a "profile vector" which records user's interests in a transparent way. The updated content in the profile vector enables the guidance functionality. For instance, if the user shows interest in a POI which describes a house with balcony, this choice of amenity will reflect his/her profile to prioritize other houses with balcony in future iterations.

Feedback is often expressed *explicitly*, i.e., the user clicks on a POI and mentions if he/she likes or dislikes the POI [11–13]. However, there are several cases that the feedback is expressed *implicitly*, i.e., the user does not explicitly click on a POI, but there exist correlations with other signals captured from the user which provide hint on his/her interest. For instance, it is often the case in spatial data analysis that users look at some regions of interest but do not provide an explicit feedback. Another example is frequent mouse moves around a region which is a good indicator of the user's potential interest in the POIs in that region. Users often hover their mouse in a region of interest to collect information on the background map (e.g., touristic places, parks, and nearby grocery stores, presented in the form of map layers and tooltips) before landing on a decision about picking a POI (such as a hotel) in that region. Implicit feedbacks are more challenging to capture and hence less investigated in the literature. The following example describes a use case of implicit feedbacks. This will be our running example which we follow throughout the paper.

[1] *http://www.tableau.com*
[2] *http://www.simile-widgets.org/exhibit/*
[3] *http://spotfire.tibco.com*

Example. *Benício is planning to live in Paris for a sabbatical leave. He decides to rent a home-stay from Airbnb website*[4]*. He likes to discover the city, hence he is open to any type of lodging in any region with an interest to stay in the center of Paris. The website returns 1500 different locations. As he has no other preferences, an exhaustive investigation needs scanning each location independently, which is nearly infeasible. While he is scanning the very first options, he shows interest in the region of Trocadero by hovering his mouse around the Eiffel tower and checking the amenities within that region. However, he forgets or doesn't feel necessary to click a POI (i.e., a home-stay) in that region. An ideal system should capture this implicit feedback in order to short-list a small subset of locations that Benício should consider as high priority.*

The above example shows in practice that implicit feedback capturing is crucial in the context of spatial data analysis, as a POI can easily remain out of sight and be missed. It is particularly the case in small-screen devices such as smart watches, smartphones, and tablets.

In this paper, we present a simple yet effective approach whose aim is to capture and analyze implicit feedback of users in spatial data analysis. Without loss of generality, we focus on "mouse moves" as the implicit feedback received from the user. Mouse moves are the most common way that users interact with geographical maps to collect information within a region of interest [14], such as information provided in the background map (e.g., parks, theaters, shopping centers) and the tooltip information which pop up by hovering (e.g., information about the POIs such as price and reviews). It is shown in [15] that mouse gestures have a strong correlation with "user engagement". Intuitively, a POI gets a higher weight in the user's profile if the mouse cursor moves around it frequently. However, our approach is independent from the type of feedback enabler and can be easily extended to other types such as gaze tracking, leap motions, as well as touch gestures in touch screens.

Contributions. In this paper, we make the following contributions:

- We define and explore the notion of "implicit user feedback" which enables a seamless navigation in spatial data.
- We define the notion of "information highlighting", a mechanism to highlight out-of-sight important information for users. A clear distinction of our proposal with the literature is that it doesn't aim for pruning (such as top-k recommendation), but leveraging the actual data with potential interesting results (i.e., highlights).
- We define and formalize the concept of Interesting Dense Regions (IDRs), a polygon-based approach to explore and highlight spatial data.
- We propose an efficient greedy approach to compute highlights on-the-fly.
- We show the effectiveness of our approach through a set of qualitative experiments.

The outline of the paper is the following. Section 2 describes our data model. In Section 3, we formally define our problem. Then in Section 4, we present our solution and its algorithmic details. Section 5 reports our experiments on the framework. We review the related work in Section 6. We present some limitations of our work in Section 7. Last, we conclude in Section 8.

[4] *http://www.airbnb.com*

2 DATA MODEL

We consider two different layers on a geographical map: "spatial layer" and "interaction layer". The spatial layer contains POIs from a spatial database $\mathcal{P}$. The interaction layer contains mouse move points $\mathcal{M}$.

Spatial layer. Each POI $p \in \mathcal{P}$ is described using its coordinates, *latitude* and *longitude*, i.e., $p = \langle lat, lon \rangle$. Note that in this work, we don't consider "time" for POIs, as our contribution focuses on their location. POIs are also associated to a set of domain-specific attributes $\mathcal{A}$. For instance, for a dataset of a real estate agency, POIs are properties (houses and apartments) and $\mathcal{A}$ contains attributes such as "surface", "number of rooms" and "price". The set of all possible values for an attribute $a \in \mathcal{A}$ is denoted as $dom(a)$. We also define user's feedback F as a vector over all attribute values (i.e., facets), i.e., $F = \overrightarrow{\cup_{a \in \mathcal{A}} dom(a)}$. The vector F is initialized by zeros and will be updated to express user's preferences.

Interaction layer. Whenever the user moves his/her mouse, a new point m is appended to the set $\mathcal{M}$. Each mouse move point is described using the pixel position that it touches and the clock time of the move. Hence each mouse move point is a tuple $m = \langle x, y, t \rangle$, where x and y specifies the pixel location and t is a Unix Epoch time. To conform with geographical standards, we assume $m = \langle 0, 0 \rangle$ sits at the middle of the interaction layer, both horizontally and vertically.

The user is in contact with the interaction layer. To update the feedback vector F, we need to translate pixel locations in the interaction layer to latitudes and longitudes in the spatial layer. We employ equirectangular projection to obtain the best possible approximation of a point $m = \langle x, y, t \rangle \in \mathcal{M}$ in the spatial layer, denoted as $p(m)$.

$$p(m = \langle x, y, t \rangle) = \langle lat = y + \gamma, lon = \frac{x}{cos\gamma} + \theta \rangle \quad (1)$$

The inverse operation, i.e., transforming a point $p = \langle lat, lon \rangle$ from the spatial layer to the interaction is done using Equation 2.

$$m(p = \langle lat, lon \rangle) = \langle x = (lon - \theta) \times cos\gamma, y = lat - \gamma \rangle \quad (2)$$

The reference point for the transformation is the center of both layers. In Equations 1 and 2, we assume that γ is the latitude and θ is the longitude of a point in the spatial layer corresponding to the center of the interaction layer, i.e., $m = \langle 0, 0 \rangle$.

3 PROBLEM DEFINITION

When analyzing spatial data, users require to obtain only a few options (so-called "highlights") to focus on. These options should be in-line with what they have already appreciated. In this paper, we formulate the problem of "information highlighting using implicit feedback", i.e., highlight a few POIs based on implicit interests of the user in order to guide him/her towards what he/she should concentrate on in consecutive iterations of the analysis process. We formally define our problem as follows.

Problem. *Given a time t_c and an integer constant k, update the feedback vector F using points $m \in \mathcal{M}$ where $m.t \leq t_c$, and choose k POIs $\mathcal{P}_k \subseteq \mathcal{P}$ as "highlights" where $\mathcal{P}_k$ satisfies two following constraints.*

(i) $\forall p \in \mathcal{P}_k$, similarity$(p, F)$ is maximized.

(ii) diversity$(\mathcal{P}_k)$ is maximized.

The first constraint guarantees that returned highlights are highly similar with user's interests captured in F. The second

Algorithm 1: Spatial Highlighting Algorithm

Input: Current time t_c, mouse move points $\mathcal{M}$
Output: Highlights $\mathcal{P}_k$

1 $\mathcal{S} \leftarrow$ *find_interesting_dense_regions*$(t_c, \mathcal{M})$ *// Section 4.1*
2 $\mathcal{P}_s \leftarrow$ *match_points*$(\mathcal{S}, \mathcal{P})$ *// Section 4.2*
3 $F \leftarrow$ *update_feedback_vector*$(F, \mathcal{P}_s)$ *// Section 4.3*
4 $\mathcal{P}_k \leftarrow$ *get_highlights*$(\mathcal{P}, F)$ *// Section 4.4*
5 **return** $\mathcal{P}_k$

constraint ensures that k POIs cover different regions and they don't repeat themselves. While our approach is independent from the way that *similarity* and *diversity* functions are formulated, we provide a formal definition of these functions in Section 4.

The aforementioned problem is hard to solve due to the following challenges.

Challenge 1. First, it is not clear how mouse move points influence the feedback vector. Mouse moves occur on a separate layer and there should be some meaningful transformations to interpret mouse moves as potential changes in the feedback vector.

Challenge 2. Analyzing all generated mouse moves is challenging and may introduce false positives. A typical mouse with 1600 DPI (Dots Per Inch) touches 630 pixels for one centimeter of move. Hence a mouse move from the bottom to the top of a typical 13-inch screen would yield 14,427 points which may not be necessarily meaningful.

Challenge 3. Beyond the two first challenges, finding the most similar and diverse POIs with F needs an exhaustive scan of all POIs in $\mathcal{P}$ which is prohibitively expensive: in most spatial datasets, there exist millions of POIs. Moreover, we need to follow bi-objective optimization considerations as we aim to optimize both similarity and diversity at the same time [16].

We recognize the complexity of our problem using the aforementioned challenges. In Section 4, we discuss a solution for the discussed problem and its associated challenges.

4 INTERESTING DENSE REGIONS

Our approach exploits user's implicit feedback (i.e., mouse moves) to highlight a few interesting POIs as future analysis directions. Algorithm 1 summarizes the principled steps of our approach.

The algorithm begins by mining the set of mouse move points $\mathcal{M}$ in the interaction layer to discover one or several Interesting Dense Regions, abbr., IDRs, in which most user's interactions occur (line 1). Then it matches the POIs $\mathcal{P}$ with IDRs using Equation 2 in order to find POIs inside each region (line 2). The attributes of resulting POIs will be exploited to update the user's feedback vector F (line 3). The updated vector F will then be used to find k highlights (line 4). These steps ensure that the final highlights reflect user's implicit interests. We detail each step as follows (Sections 4.1 to 4.4).

4.1 Discovering IDRs

The objective of this step is to obtain one or several regions in which the user has expressed his/her implicit feedback. There are two observations for such regions.

Observation 1. We conjecture that a region is more interesting for the user if it is denser, i.e., the user moves his/her mouse in that region several times, to collect information from the background map.

Algorithm 2: Find Interesting Dense Regions (IDRs)

Input: Current time t_c, mouse move points $\mathcal{M}$
Output: IDRs $\mathcal{S}$

1 $\mathcal{S} \leftarrow \emptyset$
2 $g \leftarrow$*number of time segments*
3 **for** $i \in [1, g]$ **do**
4 $\quad \mathcal{M}_i \leftarrow \{m = \langle x, y, t \rangle | (\frac{t_c}{g} \times i) \leq t \leq (\frac{t_c}{g} \times (i+1))\}$
5 $\quad C_i \leftarrow$ *mine_clusters*$(\mathcal{M}_i)$
6 $\quad O_i \leftarrow$ *find_ploygons*(C_i)
7 **end**
8 **for** O_i, O_j *where* $i, j \in [0, g]$ *and* $i \neq j$ **do**
$\quad \mathcal{S}.append(intersect(O_i, O_j))$
9 **return** $\mathcal{S}$

Observation 2. It is possible that the user moves his/her mouse everywhere in the map. This should not signify that all those locations have the same significance.

Following our observations, we propose Algorithm 2 for mining IDRs. We add points to $\mathcal{M}$ only every 200ms to prevent adding redundant points (i.e., Challenge 2). Following Observation 1 and in order to mine the recurring behavior of the user, the algorithm begins by partitioning the set $\mathcal{M}$ into g fixed-length consecutive segments $\mathcal{M}_1$ to $\mathcal{M}_g$. The first segment starts at time zero (where the system started), and the last segment ends at t_c, i.e., the current time. Following Observation 2, we then find dense clusters in each segment of $\mathcal{M}$ using a variant of DB-SCAN approach [17]. Finally, we return intersections among those clusters as IDRs.

For clustering points in each time segment (i.e., line 5 of Algorithm 2), we use ST-DBSCAN [18], a space-aware variant of DB-SCAN for clustering points based on density. For each subset of mouse move points $\mathcal{M}_i$, $i \in [1, g]$, ST-DBSCAN begins with a random point $m_0 \in \mathcal{M}_i$ and collects all density-reachable points from m_0 using a distance metric. As mouse move points are in the 2-dimensional pixel space (i.e., the display), we choose euclidean distance as the distance metric. If m_0 turns out to be a core object, a cluster will be generated. Otherwise, if m_0 is a border object, no point is density-reachable from m_0 and the algorithm picks another random point in $\mathcal{M}_i$. The process is repeated until all of the points have been processed.

Once clusters are obtained for all subsets of $\mathcal{M}$, we find their intersections to locate recurring regions (line 6). To obtain intersections, we need to clearly define the spatial boundaries of each cluster. Hence for each cluster, we discover its corresponding polygon that covers the points inside. For this aim, we employ Quickhull algorithm, a quicksort-style method which computes the convex hull for a given set of points in a 2D plane [19].

We describe the process of finding IDRs in an example. Figure 1 shows the steps that Benício follows in our running example to explore home-stays in Paris. Figure 1.A shows mouse moves of Benício in different time stages. In this example, we consider $g = 3$ and capture Benício's feedback in three different time segments (progressing from Figures 1.B to 1.D). It shows that Benício started his search around Eiffel Tower and Arc de Triomphe (Figure 1.B) and gradually showed interest in south (Figure 1.C) and north (Figure 1.D) as well. All intersections between those clusters are discovered (hatching regions in Figure 1.E) which will constitute the set of IDRs (Figure 1.F), i.e., IDR1 to IDR4.

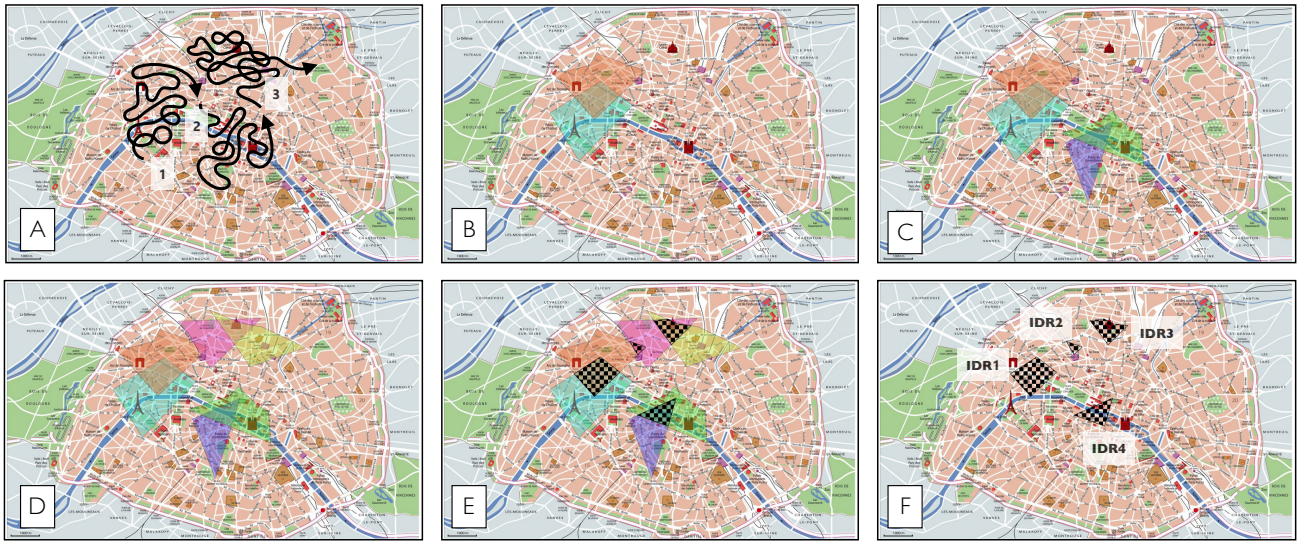

Figure 1: The process of finding IDRs on Airbnb dataset.

4.2 Matching Points

Being a function of mouse move points, IDRs are discovered in the interaction layer. We then need to find out which POIs in $\mathcal{P}$ fall into IDRs, hence forming the subset $\mathcal{P}_s$. We employ Equation 2 to transform those POIs from the spatial layer to the interaction layer. Then a simple "spatial containment" function can verify which POIs fit into the IDRs. Given a POI p and an IDR r, a function *contains*(p, r) returns "true" if p is inside r, otherwise "false". In our case, we simply use the implementation of *ST_Within*(p, r) module in PostGIS[5], i.e., our underlying spatial DBMS which hosts the data.

In the vanilla version of our spatial containment function, all POIs should be checked against all IDRs. Obviously, this depletes the execution time. To prevent the exhaustive scan, we employ Quadtrees [20] in a two-step approach.

(*i*) In an offline process, we build a Quadtree index for all POIs in $\mathcal{P}$. We record the membership relations of POIs and cells in the index.

(*ii*) When IDRs are discovered, we record which cells in the Quadtree index intersect with IDRs. As we often end up with few IDRs, the intersection verification performs fast. Then for matching POIs, we only check a subset which is inside the cells associated to IDRs and ignore the POIs outside. This leads to a drastic pruning of POIs in $\mathcal{P}$.

We follow our running example and illustrate the matching process in Figure 2. In the Airbnb dataset, POIs are home-stays which are shown with their nightly price on the map. We observe that there exist many matching POIs with IDR3 and absolutely no matching POI for IDR2. For IDR4, although there exist many home-stays below the region, we never check their containment, as they belong to a Quadtree cell which doesn't intersect with the IDR.

4.3 Updating user Feedback Vector

The set of matching POIs $\mathcal{P}_s$ (line 2 of Algorithm 1) depicts the implicit preference of the user. We keep track of this preference

[5] *https://postgis.net/docs/manual-dev/ST_Within.html*

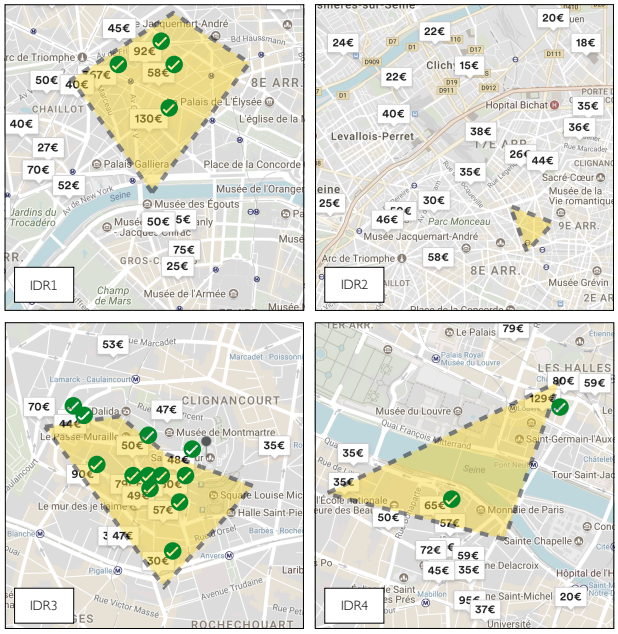

Figure 2: Matching POIs for IDR1 to IDR4.

in a feedback vector F. The vector is initialized by zero, i.e., the user has no preference at the beginning. We update F using the attributes of the POIs in $\mathcal{P}_s$. We enable transparency by choosing F's schema to be defined on the POI attributes. In exploratory analysis scenarios, it is often the case that users do not trust in what they get from the system (i.e., algorithmic anxiety [21]) and want to know what has been learned from them. Having a transparent user profile enables an easy examination of its content by the user.

We consider an *increment value* δ to update F. If $p \in \mathcal{P}_s$ gets v_1 for attribute a_1, we augment the value in the F's cell of $\langle a_1, v_1 \rangle$ by δ. Note that we only consider incremental feedback, i.e., we never decrease a value in F.

Table 1: Attributes of POIs in IDR1.

ID	Price	#Beds	Balcony	Air-cond.	Rating
1	130€	1	Yes	Yes	5/5
2	58€	1	Yes	No	5/5
3	92€	2	Yes	No	5/5
4	67€	1	Yes	No	4/5

Table 2: Updating user Feedback Vector

Attribute-value	Applying IDR 1	Normalized
⟨#Beds,1⟩	$+3\delta$	0.19
⟨#Beds,2⟩	$+\delta$	0.06
⟨#Beds,+2⟩	*(no update)*	0.00
⟨Balcony,Yes⟩	$+4\delta$	**0.25**
⟨Balcony,No⟩	*(no update)*	0.00
⟨Air-cond.,Yes⟩	$+\delta$	0.06
⟨Air-cond.,No⟩	$+3\delta$	0.19
⟨Rating,1⟩	*(no update)*	0.00
⟨Rating,2⟩	*(no update)*	0.00
⟨Rating,3⟩	*(no update)*	0.00
⟨Rating,4⟩	$+\delta$	**0.06**
⟨Rating,5⟩	$+3\delta$	0.19

We explain the process of updating the feedback vector using a toy example. Given the four matched POIs in IDR1 (Figure 2) with prices 130€, 58€, 92€ and 67€, we want to update the vector F given those POIs. A few attributes of these POIs are mentioned in Table 1. In practice, there are often more than 50 attributes for POIs. The cells of F are illustrated in the first column of Table 2. As three POIs get the value "1" for the attribute "#Beds", then the value in cell ⟨#Beds,1⟩ is augmented three times by δ. The same process is repeated for all attribute-values of POIs in $\mathcal{P}_s$. Note that all cells of F are not necessarily touched in the feedback update process. For instance, in the above example, 5 cells out of 12 remain unchanged.

By specifying an increment value, we can materialize the updates and normalize the vector using a Softmax function. Given $\delta = 1.0$, the normalized values of the F vector is illustrated in the third column of Table 2. Higher values of δ increase the influence of feedbacks.

The normalized content of the vector F captures the implicit preferences of the user. For instance, the content of F after applying POIs in IDR1 shows that the user has a high interest in having a balcony in his/her home-stay, as his/her score for the cell ⟨Balcony,Yes⟩ is 0.25, i.e., the highest among other cells. This reflects the fact that all POIs in IDR1 has balcony. It is important to note that although we only consider positive feedback, the Softmax function lowers the values of untouched cells once other cells get rewarded. The updated feedback vector is fully transparent and the user can easily comprehend what has been learned from his/her previous actions.

4.4 Generating Highlights

The ultimate goal is to highlight k POIs to guide users in analyzing their spatial data. The updated feedback vector F is the input to the highlighting phase. We assume that POIs in IDRs are already investigated by the user. Hence our search space for highlighting is $\mathcal{P} \setminus \mathcal{P}_s$. We seek two properties in k highlights: *similarity* and *diversity*. First, the highlights should be in the same direction of the user's implicit feedback, hence similar to the vector F. The similarity between a POI $p \in \mathcal{P}$ and the vector F is defined as follows.

$$similarity(p, F) = avg_{a \in \mathcal{A}}(sim(p, F, a)) \tag{3}$$

The *sim*() function can be any function such as Jaccard or Cosine. Each attribute can have its own similarity function (as string and integer attributes are compared differently.) Then *sim*() works as an overriding-function which provides encapsulated similarity computations for any type of attribute.

Second, highlighted POIs should also represent distinct directions so that the user can observe different aspects of data and decide based on the big picture. Given a set of POIs $\mathcal{P}_k = \{p_1, p_2 \dots p_k\} \subseteq \mathcal{P}$, we consider a pair-wise definition of *diversity* as follows.

$$diversity(\mathcal{P}_k) = avg_{\{p,p'\} \subset \mathcal{P}_k | p \neq p'} distance(p, p') \tag{4}$$

The function $distance(p, p')$ operates on geographical coordinates of p and p' and can be considered as any distance function of Minkowski distance family. However, as distance computations are done in the spherical space, a natural choice is to employ Haversine distance.

To solve our bi-objective problem (i.e., maximizing similarity and diversity), we employ the *ϵ-constraint* method [22], i.e., we maximize diversity while satisfying a linear constraint for similarity which only accepts values larger than or equal to ϵ. Algorithm 3 describes our approach for highlighting k similar and diverse POIs. We propose a best-effort greedy approach to efficiently compute highlighted POIs. We consider an offline step followed by the online execution of our algorithm.

In order to speed up the similarity computation in the online execution, we pre-compute an inverted index for each single POI $p \in \mathcal{P}$ in the offline step (as is commonly done in the Web search). Each index $\mathcal{L}_p$ for the POI p keeps all other POIs in $\mathcal{P}$ in decreasing order of their similarity with p. While index computation may be extremely time-consuming, we stop generating those lists if the similarity value is lower than the ϵ threshold.

The first step of Algorithm 3 is to find the most similar POI to F, so-called p^*. The POI p^* is the closest possible approximation of F in order to exploit pre-computed similarities. The algorithm makes sequential accesses to $\mathcal{L}_{p^*}$ (i.e., the inverted index of the POI p^*) to greedily maximize diversity. Algorithm 3 does not sacrifice efficiency in price of value. We consider a *time limit* parameter which determines when the algorithm should stop seeking maximized diversity. Scanning inverted indexes guarantees the similarity maximization even if time limit is chosen to be very restrictive. Our observations with several spatial datasets show that we achieve the diversity of more than 0.9 with time limit set to 200*ms*.

In line 2 of Algorithm 3, $\mathcal{P}_k$ is initialized with the k highest ranking POIs in $\mathcal{L}_{p^*}$. Function $get\_next(\mathcal{L}_{p^*})$ (line 3) returns the next POI p_{next} in $\mathcal{L}_{p^*}$ in sequential order. Lines 4 to 12 iterate over the inverted indexes to determine if other POIs should be considered to increase diversity while staying within the time limit.

The algorithm looks for a candidate POI $p_{current} \in \mathcal{P}_k$ to replace in order to increase diversity. The boolean function *diversity_improved*() (line 6) checks if by replacing $p_{current}$ by

Algorithm 3: Get k similar and diverse highlights *get_highlights*()

Input: POIs $\mathcal{P}$, Feedback vector F, k, *time_limit*
Output: $\mathcal{P}_k$

$p^* \leftarrow max\_sim\_to(\mathcal{P}, F)$
$\mathcal{P}_k \leftarrow top\_k(\mathcal{L}_{p^*}, k)$
$p_{next} \leftarrow get\_next(\mathcal{L}_{p^*})$
while *time_limit not exceeded* $\vee$ $p_{next} = \emptyset$ **do**
  for $p_{current} \in \mathcal{P}_k$ **do**
    if $diversity\_improved(\mathcal{P}_k, p_{next}, p_{current})$ **then**
      $\mathcal{P}_k \leftarrow replace(\mathcal{P}_k, p_{next}, p_{current})$
      break
    end
  end
  $p_{next} \leftarrow get\_next(\mathcal{L}_{p^*})$
end
return $\mathcal{P}_k$

p_{next} in $\mathcal{P}_k$, the overall diversity of the new $\mathcal{P}_k$ increases. It is important to highlight that for each run of the algorithm, we only focus on one specific inverted list associated to the input POI. Algorithm 3 verifies the similarity and diversity of each POI with all other POIs, and then processes the normalization.

5 EXPERIMENTS

We discuss two sets of experiments. The first set is on the usefulness of our approach. Then we focus more on IDR discovery and present some statistics and insights on the functionality of our approach. The experiments are done on a 2.8 GHz Intel Core i5 machine running a Mac OS operating system.

5.1 Usefulness

First off, we validate the usefulness of our approach. For this aim, we design a user study with 30 participants who are all students of Computer Science. Given an exploratory task to fulfill, we hypothesize that there are three prominent factors which influence the usefulness of our system: *user expertise*, *type of the task*, and *starting point*. To examine the user expertise, we recruited 15 "novice" participants who don't know the location under investigation, and also 15 "experts." We also define two different types of tasks on the Airbnb dataset of Paris with 1,000 POIs: *T1: "finding a POI in a requested location"* (e.g., find a home-stay in the "*Champ de Mars*" area), and *T2: "finding a POI with a requested profile"* (e.g., find a cheap home-stay.) Last, the participants may have different starting points, i.e., they may begin their navigation from a POI which is either *I1: "geographically close to the goal"* or *I2: "far from the goal"*. For each participant, we report a variant of time-to-insight measure, i.e., how long (in seconds) the participants interact with the tool before fulfilling the task. Evidently, less number of interactions are preferred as it means that the participant can reach insights faster.

The participants perform the tasks $T1$ and $T2$ on two different tools: Airbnb website (without information highlighting and without incorporation of implicit feedbacks, as our baseline) and our highlighting tool. Table 3 shows the results. Our initial observation is that the highlights enable users to fulfill tasks one order of magnitude faster than the baseline without the highlights. This shows that the highlighting approach with implicit feedback capturing is an effective mechanism which helps users to reach their goals in a reasonable time. Also expert participants need on average 6.25 seconds less time than the novice participants when using the highlights. Interestingly, starting POIs, i.e., *I1* and *I2*, do not have a huge impact on the number of steps. It is potentially due to the diversity component which provides distinct options and can quickly steer users towards their region of interest. We also observe that the task *T1* is an easier task than *T2*, as on average it took less time to be accomplished. This is potentially due to where the user can request options similar to what he/she has already observed and greedily move to his/her preferred regions.

Table 3: Interactions of "novice" and "expert" participants

	T1/I1	T2/I1	T1/I2	T2/I2
Novices using Airbnb	372*s*	421*s*	365*s*	430*s*
Experts using Airbnb	253*s*	302*s*	249*s*	298*s*
Novices using highlights	19*s*	24*s*	20*s*	25*s*
Experts using highlights	14*s*	17*s*	13*s*	17*s*

5.2 Effectiveness

In the second part of our experiments, we analyze the effectiveness of our approach in generating IDRs for information highlighting. We employ two different datasets, i.e., Airbnb and Yelp[6]. We pick a similar subset from both datasets, i.e., home-stays and restaurants in Paris, respectively. We consider four different sizes of those datasets, i.e., 100, 1000, 2000 and 4000 POIs, respectively. For each size of the datasets, we manually perform 20 sessions, and then present some statistics as the average of the sessions. We limit each session to 2 minutes where we seek for interesting POIs in the datasets. We capture the following information in each session:

- The number of regions created from the mouse moves during the session;
- The number of generated IDRs (intersection of regions);
- The number of POIs from the dataset presented in each IDR;
- The coverage of POIs (in the dataset) with IDRs collectively.

Tables 4 and 5 show the results for Airbnb and Yelp, respectively. In Table 4, we observe that the number of regions decreases when the number of POIs increases. On average, 7.7 regions are constructed per session. The average number of POIs presented in IDRs is 25.97, which shows that our approach highlights at least 8.05% of POIs from the dataset, on average. We notice an outlier in the experiment with 2000 POIs in Table 4. This happened due the fact that the user concentrated in a very small area generating a smaller number of IDRs, and consequently a smaller number of POIs.

More uniform results are observed in Table 5, i.e., for Yelp dataset vis-à-vis Airbnb. The average number of generated regions reaches 12.75 per session. Also, the number of regions decreases by increasing the number of POIs. The number of POIs presented in IDRs is on average 108.65 and it represents on average 13.11% of POIs highlighted from the dataset.

6 RELATED WORK

To the best of our knowledge, the problem of spatial information highlighting using implicit feedback has not been addressed before in the literature. However, our work relates to a few others in their semantics.

[6] *https://www.yelp.com/dataset*

Table 4: IDR statistics on Airbnb dataset

# POIs	# regions	# IDRs	# POIs in IDRs	% POIs
100	11.35	10.05	29.40	29.40%
1000	10.75	6.75	11.70	1.17%
2000	7.37	3.63	5.63	0.003%
4000	1.30	1.15	53.15	1.33%
average	**7.69**	**7.64**	**25.97**	**8.05%**

Table 5: IDR statistics on Yelp dataset

# POIs	# regions	# IDRs	# POIs in IDRs	% POIs
100	14.90	7.55	28.30	28.30%
1000	13.90	10.00	149.55	14.96%
2000	11.05	9.80	111.05	5.55%
4000	10.45	8.55	145.7	3.64%
average	**12.57**	**8.97**	**108.65**	**13.11%**

Information Highlighting. The literature contains a few instances of information highlighting approaches [23–26]. However, all these methods are objective, i.e., they assume that user's preferences are given as a constant input and will never change in the future. This limits their functionality for serving scenarios of exploratory analysis. The only way to fulfill "spatial guidance" is to consider the evolutionary and subjective nature of user's feedback. In our approach, the feedback vector gets updated in time based on the implicit feedback of the user.

Online recommendation approaches can also be considered as an information highlighting approach where recommended items count as highlights. Most recommendation algorithms are space-agnostic and do not take into account the spatial information. While a few approaches focus on the spatial dimension [27–29], they still lack the evolutionary feedback capturing. Moreover, most recommendation methods miss "result diversification", i.e., highlights may not be useful due to overlaps.

Feedback Capturing. Several approaches are proposed in the state of the art for capturing different forms of feedback [8, 9, 11, 12, 30, 31]. The common approach is a top-k processing methodology in order to prune the search space based on the explicit feedback of the user and return a small subset of interesting results of size k. A clear distinction of our proposal is that it doesn't aim for pruning, but leveraging the actual data with potential interesting results (i.e., highlights) that the user may miss due to the huge volume of spatial data. Moreover, in a typical top-k processing algorithm, user's choices are limited to k. On the contrary, our IDR approach enables a freedom of choice where highlights get seamlessly updated with new user's choices.

Few works formulate fusing approaches of explicit and implicit feedbacks to better capture user preferences [6, 7, 32]. Our approach functions purely on implicit feedback and does not require any sort of explicit signal from the user.

Region Discovery. Our approach finds interesting dense regions (IDRs) in order to derive user's implicit preferences. There exist several approaches to infer a spatial region for a given set of POIs [19, 33–37]. The common approach is to cluster POIs in form of concave and convex polygons. In [33], an algorithm is proposed to verify if a given POI p on the surface of a sphere is located inside, outside, or along the border of an arbitrary spherical polygon. In [34, 35], a non-convex polygon is constructed from a set of input POIs on a plane. In [36, 37], imprecise regions are delineated into a convex or concave polygon. In our approach, it is important to discover regions by capturing mouse move POIs. In case a concave polygon is constructed, the "dents" of such a polygon may entail POIs which are not necessarily in $\mathcal{M}$. In the IDR's algorithm, however, we adapt Quickhull [19], due its simplicity, efficiency and its natural implementation of convex polygons.

7 LIMITATIONS

In this paper, we presented a solution for highlighting out-of-sight information using a polygon-based approach for capturing implicit feedbacks. To the best of our knowledge, our work is the first effort towards formalizing and implementing information highlighting using implicit feedback. However, we consider our work as an on-going effort where we envision to address some limitations in the future, such as "customizability", "performance", "cold start", and "quantitative experiments".

Custom maps and feedbacks. One limitation is about the "customizable" use of geographical maps as an interaction means. As we only consider static maps, we plan to work on translations and rotations as a future work. We also consider adapting other types of feedback enablers such as gaze tracking and touch gestures as a direction of future work. Our research will investigate potential challenges which may arise when employing those feedback enablers.

Mobility. Our current focus is only on POIs, and hence trajectories and areas of mobility are not incorporated directly. While trajectories and areas can be broken into a subset of POIs (and hence be easily fed to our approach), we envision to enrich our model in a way that those mobility elements are considered as first-class citizens.

Cold start. Our problem bears similarities with recommendation algorithms where the quality of the output may be influenced by scarce availability of input. This problem is referred to as the cold start problem [38, 39]. While there is no guarantee for a meaningful highlight in case of the complete absence of implicit feedbacks, our approach can return a reasonable set of highlights even with one single iteration of mouse moves. In the future, we envision to tackle the no-input challenge by leveraging statistical properties of the spatial data to obtain a default view for highlights.

Performance. Another limitation is the medium-size datasets to be processed. Our algorithm processes similarity and diversity in an $O(n^2)$ complexity. Also Quickhull [19] uses a divide and conquer approach similar to that of Quicksort, and its worst complexity is $O(n^2)$. While processing a 10K-POI dataset is straightforward in our framework, we plan to experiment with larger datasets in the future by improving our algorithms towards better performance. Another direction for future work is to consider experiments which measure the quantitative and qualitative influence of each component separately.

8 CONCLUSION

In this paper, we present an approach to explore Interesting Dense Regions (IDRs) using implicit feedback in order to detect user latent preferences. The implicit feedbacks are captured from mouse moves of users over the geographical map while analyzing spatial data. We formalize a novel polygon-based mining algorithm which returns a few highlights in-line with user's implicit

preferences. The highlights enable users to focus on what matters the most and prevent information overload.

We consider various future directions for this work, including the limitations of our current approach which we listed in Section 7. First, we are interested to incorporate an "explainability" component which can describe causalities behind preferences. For instance, we are interested to find seasonal patterns to see why the preferences of users change from place to place during various seasons of the year. Another direction is to incorporate "Query by Visualization" approaches, where users can specify their intents alongside their implicit preferences, directly on the map [40].

REFERENCES

[1] John F. Roddick, Max J. Egenhofer, Erik G. Hoel, Dimitris Papadias, and Betty Salzberg. Spatial, temporal and spatio-temporal databases - hot issues and directions for phd research. *SIGMOD Record*, 33(2):126–131, 2004.

[2] Aditya Telang, Deepak Padmanabhan, and Prasad Deshpande. Spatio-temporal indexing: Current scenario, challenges and approaches. In *Proceedings of the 18th International Conference on Management of Data*, COMAD '12, pages 9–11, Mumbai, India, India, 2012. Computer Society of India.

[3] Lauro Lins, James T Klosowski, and Carlos Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.

[4] Jia Yu, Zongsi Zhang, and Mohamed Sarwat. Spatial data management in apache spark: the geospark perspective and beyond. *GeoInformatica*, pages 1–42, 2018.

[5] W. Bruce Croft. The importance of interaction for information retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1–2, 2019.

[6] Andrea Ballatore and Michela Bertolotto. Semantically enriching vgi in support of implicit feedback analysis. In Katsumi Tanaka, Peter Fröhlich, and Kyoung-Sook Kim, editors, *Web and Wireless Geographical Information Systems*, pages 78–93, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[7] Nathan N. Liu, Evan W. Xiang, Min Zhao, and Qiang Yang. Unifying explicit and implicit feedback for collaborative filtering. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1445–1448, New York, NY, USA, 2010. ACM.

[8] Dong Xin, Xuehua Shen, Qiaozhu Mei, and Jiawei Han. Discovering interesting patterns through user's interactive feedback. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 773–778. ACM, 2006.

[9] Mansurul Bhuiyan, Snehasis Mukhopadhyay, and Mohammad Al Hasan. Interactive pattern mining on hidden data: a sampling-based solution. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 95–104. ACM, 2012.

[10] Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, Eric Simon, Fabian Colque Zegarra, João LD Comba, and Viviane Moreira. Userdev: A mixed-initiative system for user group analytics. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, pages 1–8, 2019.

[11] Niranjan Kamat, Prasanth Jayachandran, Karthik Tunga, and Arnab Nandi. Distributed and interactive cube exploration. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, pages 472–483. IEEE, 2014.

[12] Behrooz Omidvar-Tehrani, Sihem Amer-Yahia, and Alexandre Termier. Interactive user group analysis. In *CIKM*, pages 403–412. ACM, 2015.

[13] Behrooz Omidvar-Tehrani, Plácido A Souza Neto, Felipe M Freire Pontes, and Francisco Bento. Geoguide: An interactive guidance approach for spatial data. In *Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2017 IEEE International Conference on*, pages 1112–1117. IEEE, 2017.

[14] Mon Chu Chen, John R. Anderson, and Myeong Ho Sohn. What can a mouse cursor tell us more?: Correlation of eye/mouse movements on web browsing. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '01, pages 281–282, New York, NY, USA, 2001. ACM.

[15] Ioannis Arapakis, Mounia Lalmas, and George Valkanas. Understanding within-content engagement through pattern analysis of mouse gestures. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 1439–1448, New York, NY, USA, 2014. ACM.

[16] Jacek Malczewski and Claus Rinner. *Multicriteria decision analysis in geographic information science.* Springer, 2015.

[17] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[18] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial-temporal data. *Data Knowl. Eng.*, 60(1):208–221, January 2007.

[19] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, December 1996.

[20] Raphael A. Finkel and Jon Louis Bentley. Quad trees a data structure for retrieval on composite keys. *Acta informatica*, 4(1):1–9, 1974.

[21] Shagun Jhaver, Yoni Karpfen, and Judd Antin. Algorithmic anxiety and coping strategies of airbnb hosts. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 421. ACM, 2018.

[22] R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.

[23] J. Liang and M. L. Huang. Highlighting in information visualization: A survey. In *2010 14th International Conference Information Visualisation*, July 2010.

[24] Anthony C. Robinson. Highlighting in geovisualization. *Cartography and Geographic Information Science*, 38(4):373–383, 2011.

[25] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *TVCG*, 22(1), 2016.

[26] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1129–1136, 2007.

[27] Jie Bao, Yu Zheng, David Wilkie, and Mohamed Mokbel. Recommendations in location-based social networks: a survey. *GeoInformatica*, 19(3):525–565, 2015.

[28] Justin J. Levandoski, Mohamed Sarwat, Ahmed Eldawy, and Mohamed F. Mokbel. Lars: A location-aware recommender system. In *ICDE*, pages 450–461, 2012.

[29] Marina Drosou and Evaggelia Pitoura. Disc diversity: result diversification based on dissimilarity and coverage. *PVLDB*, 6(1):13–24, 2012.

[30] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Aide: an active learning-based approach for interactive data exploration. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2842–2856, 2016.

[31] Mario Boley, Michael Mampaey, Bo Kang, Pavel Tokmakov, and Stefan Wrobel. One click mining: Interactive local pattern discovery through implicit preference and performance learning. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, pages 27–35. ACM, 2013.

[32] Eoin Mac Aoidh, Michela Bertolotto, and David C. Wilson. Analysis of implicit interest indicators for spatial data. In *15th ACM International Symposium on Geographic Information Systems, ACM-GIS 2007, November 7-9, 2007, Seattle, Washington, USA, Proceedings*, page 47, 2007.

[33] Michael Bevis and Jean-Luc Chatelain. Locating a point on a spherical surface relative to a spherical polygon of arbitrary shape. *Mathematical Geology*, 21(8):811–828, Oct 1989.

[34] Matt Duckham, Lars Kulik, Mike Worboys, and Antony Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41(10):3224 – 3236, 2008.

[35] M.J. Fadili, M. Melkemi, and A. ElMoataz. Non-convex onion-peeling using a shape hull algorithm. *Pattern Recognition Letters*, 25(14):1577 – 1585, 2004.

[36] Avi Arampatzis, Marc van Kreveld, Iris Reinbacher, Christopher B. Jones, Subodh Vaid, Paul Clough, Hideo Joho, and Mark Sanderson. Web-based delineation of imprecise regions. *Computers, Environment and Urban Systems*, 30(4):436 – 459, 2006. Geographic Information Retrieval (GIR).

[37] Antony Galton and Matt Duckham. What is the region occupied by a set of points? In Martin Raubal, Harvey J. Miller, Andrew U. Frank, and Michael F. Goodchild, editors, *Geographic Information Science*, pages 81–98, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[38] Vincent Leroy, Berkant Barla Cambazoglu, and Francesco Bonchi. Cold start link prediction. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 393–402, 2010.

[39] Behrooz Omidvar-Tehrani, Sruthi Viswanathan, Frederic Roulland, and Jean-Michel Renders. SAGE: Interactive state-aware point-of-interestrecommendation. In *WSDM Workshop on State-based User Modelling*, 2020.

[40] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment*, 10(4):457–468, 2016.