

# Polygon-based Feedback Capturing on Spatial Data

Behrooz Omidvar-Tehrani  
University of Grenoble Alpes  
(France)  
[behrooz.omidvar-tehrani@univ-grenoble-alpes.fr](mailto:behrooz.omidvar-tehrani@univ-grenoble-alpes.fr)

Francisco B. Silva Junior  
Federal Institute of Rio Grande  
do Norte (Brazil)  
[bento.francisco@academico.ifrn.edu.br](mailto:bento.francisco@academico.ifrn.edu.br)

Plácido A. Souza Neto  
Federal Institute of Rio Grande  
do Norte (Brazil)

[placido.neto@ifrn.edu.br](mailto:placido.neto@ifrn.edu.br)

Tiago Oliveira Lisboa  
Federal Institute of Rio Grande  
do Norte (Brazil)  
[tiago.oliveira@academico.ifrn.edu.br](mailto:tiago.oliveira@academico.ifrn.edu.br)

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

## ABSTRACT

Nowadays, spatial data are ubiquitous in various fields of science, such as transportation and social science. Discovering spatial patterns and trends provides improved insights into planning and decision making in various applications such as smart city management. A recent research direction in analyzing spatial data is to provide means for “exploratory analysis” of such data where analysts are guided towards interesting options in consecutive analysis iterations. Typically, the guidance component learns analyst’s preferences using her explicit feedback, e.g., picking a spatial point or selecting a region of interest. However, it is often the case that analysts forget or don’t feel necessary to explicitly express their feedback in what they find interesting. In this paper, we propose GEOPOLY, an approach to capture implicit feedback on spatial data. The approach consists of observing mouse moves (as a means of analyst’s interaction) in order to discover interesting spatial regions where the analyst has frequent mouse hovers. For an efficient discovery, we extend ST-DBSCAN and introduce a polygon-based abstraction layer for captured interactions. Using interesting regions, we highlight few points for the analyst to guide her in the analysis process. While we show the process of GEOPOLY through a realistic example, we also briefly report on its efficiency and effectiveness.

### PVLDB Reference Format:

Behrooz Omidvar-Tehrani, Plácido A. Souza Neto, Tiago Oliveira Lisboa, Francisco B. Silva Junior, Felipe F. Pontes. Polygon-based Feedback Capturing on Spatial Data. *PVLDB*, 11 (9): xxxx-yyyy, 2018.

DOI: <https://doi.org/TBD>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Articles from this volume were invited to present their results at The 44th International Conference on Very Large Data Bases, August 2018, Rio de Janeiro, Brazil.

*Proceedings of the VLDB Endowment*, Vol. 11, No. 9  
Copyright 2018 VLDB Endowment 2150-8097/18/5.  
DOI: <https://doi.org/TBD>

## 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 social science. As each record in spatial data represents an activity in a precise geographical location, analyzing such data enables discoveries grounded on facts. Analysts are often interested to observe spatial patterns and trends to improve their decision making process. Spatial data analysis has various applications such as smart city management, disaster management and autonomous transport [26, 28].

Typically, spatial data analysis begins with an imprecise question in the mind of the analyst, i.e., *exploratory analysis*. The analyst requires to go through several trial-and-error iterations to improve 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<sup>1</sup>, Exhibit<sup>2</sup>, Spotfire<sup>3</sup>) where the analyst can investigate on different parts of the visualization by zooming in/out and panning on the map.

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., [20]). However, there has been fewer attention to the “value” derived from spatial data. Despite the huge progress on the efficiency front, an analyst may easily get lost in the plethora of geographical points due to two following reasons.

- In an exploratory context, the analyst doesn’t know apriori what to investigate next.
- Moreover, she may easily get distracted and miss interesting points by visual clutter caused by huge point overlaps.

The main drawback of the traditional analysis model is that the analyst has a *passive role* in the process. In other words, the analyst’s feedback (i.e., her likes and dislikes) is

<sup>1</sup><http://www.tableau.com>

<sup>2</sup><http://www.simile-widgets.org/exhibit/>

<sup>3</sup><http://spotfire.tibco.com>

ignored and only the input query (i.e., her explicit request) is served. In case feedback is incorporated, the process can be more directed towards analyst’s interests where her partial needs can be served earlier in the process. In this paper, we advocate for a “guidance layer” on top of the raw visualization of spatial data to enable analysts know “*what to see next*”. This guidance should be a function of analyst feedback: the system should recommend options similar to what the analyst has already appreciated.

Various approaches in the literature propose methodologies to incorporate analyst’s feedback in the exploration process of spatial data. Typically, feedback is considered as a function which is triggered by any analyst’s action on the map. The action can be “selecting a point”, “moving to a region”, “asking for more details”, etc. The function then updates a “profile vector” which keeps tracks of analyst’s interests. The updated content in the profile vector enables the guidance functionality. For instance, if the analyst shows interest in a point which describes a house with balcony, this choice of amenity will reflect her profile to prioritize other houses with balcony in future iterations.

Feedback is often expressed *explicitly*, i.e., the analyst clicks on a point and mentions if she likes or dislikes the point [17, 22, 24]. In [24], we proposed an interactive approach to exploit such feedback for enabling a more insightful exploration of spatial data. However, there are several cases that the feedback is expressed *implicitly*, i.e., the analyst does not explicitly click on a point, but there exists correlations with other signals captured from the analyst which provide hint on her interest. For instance, it is often the case in spatial data analysis that analysts 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 analyst’s potential interest in the points 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.

**Example.** *Benício is planning to live in Paris for a season. He decides to rent a home-stay from Airbnb website<sup>4</sup>. 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 few first options, he shows interest in the region of Trocadero (where the Eiffel tower is located in) but he forgets or doesn’t feel necessary to click a point there. 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. While text-boxes, combo-boxes and other input elements are available in analyzing other types of data, the only interaction means between the analyst and a spatial data analysis system is a geographical map spanned on the whole screen. In this context, a point can be easily remained out of sight and missed.

---

<sup>4</sup><http://www.airbnb.com>

In this paper, we present an approach called GEOPOLY whose aim is to capture and analyze implicit feedback of analysts in spatial data analysis. Without loss of generality, we focus on “mouse moves” as the implicit feedback received from the analyst. Mouse moves are the most common way that analysts interact with geographical maps [11]. It is shown in [3] that mouse gestures have a strong correlation with “user engagement”. Intuitively, a point gets a higher weight in the analyst’s profile if the mouse cursor moves around it frequently. However, our approach can be easily extended to other types of inputs such gaze tracking, leap motions, etc.

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. Last, we conclude in Section 7.

## 2. DATA MODEL

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

**Spatial layer.** Each point  $p \in \mathcal{P}$  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 spatial points as our contribution focuses on their location. Points are also associated to a set of domain-specific attributes  $\mathcal{A}$ . For instance, for a dataset of a real estate agency, points are properties (houses and apartments) and  $\mathcal{A}$  contains attributes such as “surface”, “number of pieces” and “price”. The set of all possible values for an attribute  $a \in \mathcal{A}$  is denoted as  $dom(a)$ . We also define analyst’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 manipulated to express analyst’s preferences.

**Interaction layer** Whenever the analyst moves 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 analyst 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. While there is no precise transformation from planar to spherical coordinates, we employ equirectangular projection to obtain the best possible approximation. Equation 1 describes this formula to transform a point  $m = \langle x, y, t \rangle$  in the interaction layer to a point  $p = \langle lat, lon \rangle$  in the spatial layer. Note that the resulting  $p$  is not necessarily a member of  $\mathcal{P}$ .

$$lon = \frac{x}{cos\gamma} + \theta; lat = y + \gamma \quad (1)$$

The inverse operation, i.e., transforming from the spatial layer to the interaction is done using Equation 2.

$$x = (\text{lon} - \theta) \times \cos\gamma; y = \text{lat} - \gamma \quad (2)$$

The reference point for the transformation is the center of both layers. In Equations 1 and 2, we assume that  $\gamma$  is the latitude and  $\theta$  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

The large size of spatial data hinders its effective analysis for discovering insights. Analysts require to obtain only 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 few spatial points based on implicit interests of the analyst in order to guide her towards what 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  $k$ , obtain an updated feedback vector  $F$  using points  $m \in \mathcal{M}$  where  $m.t \leq t_c$  and choose  $k$  points  $\mathcal{P}_k \subseteq \mathcal{P}$  as “highlights” where  $\mathcal{P}_k$  satisfies two following constraints.

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

The first constraint guarantees that returned highlights are highly similar with analyst’s interests captured in  $F$ . The second constraint ensures that  $k$  points 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 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.** Even if an oracle provides a mapping between mouse moves and the feedback vector, 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 provide 14,427 points which may not be necessarily meaningful.

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

In Section 4, we discuss a framework called GEOPOLY as a solution for the aforementioned problem and its associated challenges.

---

#### Algorithm 1: GEOPOLY Algorithm

---

```

Input: Current time  $t_c$ , mouse move points  $\mathcal{M}$ 
Output: Highlights  $\mathcal{P}_k$ 
1  $\mathcal{S} \leftarrow \text{find\_interesting\_dense\_regions}(t_c, \mathcal{M})$ 
2  $\mathcal{P}_s \leftarrow \text{match\_points}(\mathcal{S}, \mathcal{P})$ 
3  $F \leftarrow \text{update\_feedback\_vector}(F, \mathcal{P}_s)$ 
4  $\mathcal{P}_k \leftarrow \text{get\_highlights}(\mathcal{P}, F)$ 
5 return  $\mathcal{P}_k$ 

```

---

### 4. GEOPOLY FRAMEWORK

GEOPOLY is an approach which exploits analyst’s implicit feedback (i.e., mouse moves) to highlight few interesting points 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 analyst’s interactions occur (line 1). Then it matches the spatial points  $\mathcal{P}$  with IDRs using Equation 2 in order to find points inside each region (line 2). The attributes of resulting points will be exploited to update the analyst’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 analyst’s implicit interests. We detail each step as follows.

#### 4.1 Interesting Dense Regions

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

■ **Observation 1.** We believe that a region appeals more interesting to the analyst if it is denser, i.e., the analyst moves her mouse in that region several times.

■ **Observation 2.** It is possible that the analyst moves her mouse everywhere in the map. This should not signify that everywhere in the map has 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. Following Observation 1 and in order to mine the recurring behavior of the analyst, the algorithm begins by partitioning the set  $\mathcal{M}$  into  $g$  fixed-length consecutive segments  $\mathcal{M}_0$  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 [14] approach. 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 [9], a space-aware variant of DB-SCAN for clustering points based on density. For each subset of mouse move points  $\mathcal{M}_i$ ,  $i \in [0, g]$ , 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 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.

---

**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  $g \leftarrow \text{number of time segments}$ 
3 for  $i \in [0, g]$  do
4    $\mathcal{M}_i \leftarrow \{m = \langle x, y, t \rangle | (\frac{t_c}{g} \times i) \leq t \leq (\frac{t_c}{g} \times (i + 1))\}$ 
5    $\mathcal{C}_i \leftarrow \text{mine\_clusters}(\mathcal{M}_i)$ 
6    $\mathcal{O}_i \leftarrow \text{find\_polygons}(\mathcal{C}_i)$ 
7 end
8 for  $\mathcal{O}_i, \mathcal{O}_j$  where  $i, j \in [0, g]$  and  $i \neq j$  do
9    $\mathcal{S}.\text{append}(\text{intersect}(\mathcal{O}_i, \mathcal{O}_j))$ 
9 return  $\mathcal{S}$ 

```

---

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 [6].

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 movements 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.

## 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 points in  $\mathcal{P}$  fall into IDRs, hence forming the subset  $\mathcal{P}_s$ . We employ Equation 2 to transform those points from the spatial layer to the interaction layer. Then a simple “spatial containment” function can verify which points fit into the IDRs. Given a point  $p$  and an IDR  $r$ , a function  $\text{contains}(p, r)$  returns “true” if  $p$  is inside  $r$ , otherwise “false”. In our case, we simply use the implementation of  $\text{ST\_Within}(p, r)$  module in PostGIS<sup>5</sup>, i.e., our underlying spatial DBMS which hosts the data.

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

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

<sup>5</sup>[https://postgis.net/docs/manual-dev/ST\\_Within.html](https://postgis.net/docs/manual-dev/ST_Within.html)

Then for matching points, we only check a subset which is inside the cells associated to IDRs and ignore the points outside. This leads to a drastic pruning of points in  $\mathcal{P}$ .

We follow our running example and illustrate the matching process in Figure 2. In the Airbnb dataset, points are home-stays which are shown with their nightly price on the map. We observe that there exist many matching points with IDR3 and absolutely no matching point 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 Analyst Feedback Vector

The set of matching points  $\mathcal{P}_s$  (line 2 of Algorithm 1) depicts the implicit preference of the analyst. We keep track of this preference in a feedback vector  $F$ . The vector is initialized by zero, i.e., the analyst has no preference at the beginning. We update  $F$  using the attributes of the points in  $\mathcal{P}_s$ .

We consider an *increment value*  $\delta$  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  $\delta$ . Note that we only consider incremental feedback, i.e., we never decrease a value in  $F$ .

We explain the process of updating the feedback vector using a toy example. Given the four matched points in IDR1 (Figure 2) with prices 130€, 58€, 92€ and 67€, we want to update the vector  $F$  given those points. Few attributes of these points are mentioned in Table 1. In practice, there are often more than 50 attributes for points. The cells of  $F$  is illustrated in the first column of Table 2. As three points get the value “1” for the attribute “#Beds”, then the value in cell  $\langle \text{#Beds}, 1 \rangle$  is augmented three times by  $\delta$ . The same process is repeated for all attribute-values of points 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. We always normalize  $F$  in a way that all cell values sum up to 1.0. Given  $\delta = 1.0$ , the normalized values of the  $F$  vector is illustrated in the third column of Table 2. Higher values of  $\delta$  increase the influence of feedbacks.

The normalized content of the vector  $F$  captures the implicit preferences of the analyst. For instance, the content of  $F$  after applying points in IDR1 shows that the analyst has a high interest in having a balcony in her home-stay, as her score for the cell  $\langle \text{Balcony}, \text{Yes} \rangle$  is 0.25, i.e., the highest among other cells. This reflects the reality as all points in IDR1 has balcony. Note that although we only consider positive feedback, the Softmax function lowers the values of untouched cells once other cells get rewarded.

An important consideration in interpreting the vector  $F$  is that the value “0” does not mean the lowest preference, but *irrelevance*. For instance, consider the cell  $\langle \text{Rating}, 2 \rangle$  in Table 2. The value “0” for this cell shows that the analyst has never expressed her implicit feedback on this facet. It is possible that in future iterations, the analyst shows interest in a 2-star home-stay (potentially thanks to its price), hence this cell gets a value greater than zero. However, cells with lower preferences are identifiable with non-zero values tending to zero. For instance, the value 0.06 for the cell  $\langle \text{Rating}, 4 \rangle$  shows a lower preference towards 4-star home-

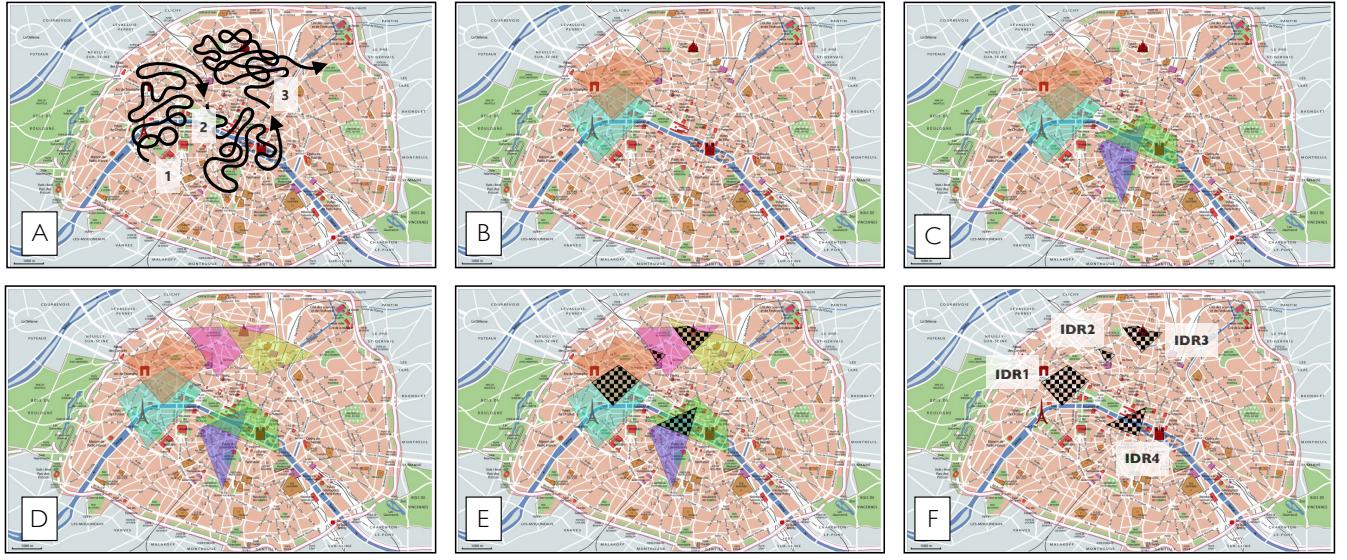


Figure 1: The process of finding IDR<sub>s</sub> on Airbnb dataset.

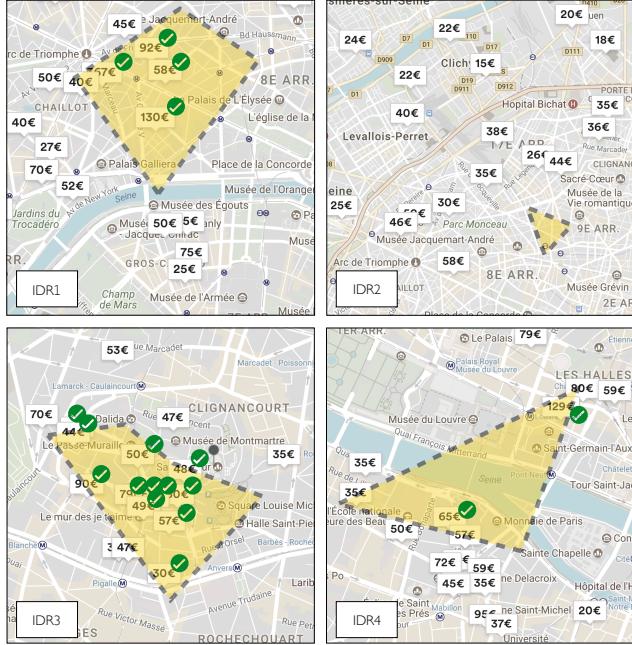


Figure 2: Matching points for IDR<sub>1</sub> to IDR<sub>4</sub>.

stays comparing to the ones with 5 stars, as only one point in  $\mathcal{P}_s$  is rated 4 in IDR<sub>1</sub>.

#### 4.4 Generating Highlights

The ultimate goal of GEOPOLY is to highlight  $k$  points to guide analysts in analyzing their spatial data. The updated feedback vector  $F$  is the input to the highlighting phase. We assume that points in IDR<sub>s</sub> are already investigated by the analyst. Hence our search space for highlighting is  $\mathcal{P} - \mathcal{P}_s$ .

We seek two properties in  $k$  highlights, i.e., *similarity* and *diversity*. First, highlights should be in the same direction

Table 1: Attributes of points in IDR<sub>1</sub>.

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 Analyst Feedback Vector

Attribute-value	Applying IDR 1	Normalized
$\langle \#Beds, 1 \rangle$	$+3\delta$	0.19
$\langle \#Beds, 2 \rangle$	$+\delta$	0.06
$\langle \#Beds, +2 \rangle$	(no update)	0.00
$\langle \text{Balcony}, \text{Yes} \rangle$	$+4\delta$	<b>0.25</b>
$\langle \text{Balcony}, \text{No} \rangle$	(no update)	0.00
$\langle \text{Air-cond.}, \text{Yes} \rangle$	$+\delta$	0.06
$\langle \text{Air-cond.}, \text{No} \rangle$	$+3\delta$	0.19
$\langle \text{Rating}, 1 \rangle$	(no update)	0.00
$\langle \text{Rating}, 2 \rangle$	(no update)	0.00
$\langle \text{Rating}, 3 \rangle$	(no update)	0.00
$\langle \text{Rating}, 4 \rangle$	$+\delta$	<b>0.06</b>
$\langle \text{Rating}, 5 \rangle$	$+3\delta$	0.19

of the analyst's implicit feedback, hence similar to the vector  $F$ . The similarity between a point  $p \in \mathcal{P}$  and the vector  $F$  is defined as follows.

$$\text{similarity}(p, F) = \text{avg}_{a \in \mathcal{A}}(\text{sim}(p, F, a)) \quad (3)$$

The *sim()* function can be any function such as Jaccard and 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 points should also represent distinct directions so that the analyst can observe different aspects

of data and decide based on the big picture. Given a set of points  $\mathcal{P}_k = \{p_1, p_2 \dots p_k\} \subseteq \mathcal{P}$ , we define *diversity* as follows.

$$\text{diversity}(\mathcal{P}_k) = \text{avg}_{\{p, p'\} \subset \mathcal{P}_k | p \neq p'} \text{distance}(p, p') \quad (4)$$

The function  $\text{distance}(p, p')$  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 shown in Equation 5.

$$\begin{aligned} \text{distance}(p, p') &= \text{acos}(\cos(p.\text{lat}) \times \cos(p'.\text{lat}) \times \cos(p.\text{lon}) \\ &\quad \times \cos(p'.\text{lon}) + \cos(p.\text{lat}) \times \sin(p'.\text{lat}) \\ &\quad \times \cos(p.\text{lon}) \times \sin(p'.\text{lon}) \\ &\quad + \sin(p.\text{lat}) \times \sin(p'.\text{lat})) \\ &\quad \times \text{earth\_radius} \end{aligned} \quad (5)$$

Algorithm 3 describes our approach for highlighting  $k$  similar and diverse points. We propose a best-effort greedy approach to efficiently compute highlighted points. 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 point  $p \in \mathcal{P}$  in the offline step (as is commonly done in the Web search). Each index  $\mathcal{L}_p$  for the point  $p$  keeps all other points in  $\mathcal{P}$  in decreasing order of their similarity with  $p$ .

The first step of Algorithm 3 is to find the most similar point to  $F$ , so-called  $p^*$ . The point  $p^*$  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 point  $p^*$ ) 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 200ms.

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

The algorithm looks for a candidate point  $p_{\text{current}} \in \mathcal{P}_k$  to replace in order to increase diversity. The boolean function  $\text{diversity\_improved}()$  (line 6) checks if by replacing  $p_{\text{current}}$  by  $p_{\text{next}}$  in  $\mathcal{P}_k$ , the overall diversity of the new  $\mathcal{P}_k$  increases.

## 5. EXPERIMENTS

We perform a few experiments on GEOPOLY to validate its efficiency and effectiveness. In the interest of space, we only present a glimpse of our experiments here. More will be discussed in an extended version.

First off, we validate the “usability” of our framework. For this aim, we design a user study with 14 participants

---

**Algorithm 3:** Get  $k$  similar and diverse highlights  $\text{get\_highlights}()$

---

```

Input: Points  $\mathcal{P}$ , Feedback vector  $F$ ,  $k$ , time_limit
Output:  $\mathcal{P}_k$ 
1  $p^* \leftarrow \text{max\_sim\_to}(\mathcal{P}, F)$ 
2  $\mathcal{P}_k \leftarrow \text{top\_k}(\mathcal{L}_{p^*}, k)$ 
3  $p_{\text{next}} \leftarrow \text{get\_next}(\mathcal{L}_{p^*})$ 
4 while time_limit not exceeded do
5   for  $p_{\text{current}} \in \mathcal{P}_k$  do
6     if  $\text{diversity\_improved}(\mathcal{P}_k, p_{\text{next}}, p_{\text{current}})$  then
7        $\mathcal{P}_k \leftarrow \text{replace}(\mathcal{P}_k, p_{\text{next}}, p_{\text{current}})$ 
8       break
9     end
10   end
11    $p_{\text{next}} \leftarrow \text{get\_next}(\mathcal{L}_{p^*})$ 
12 end
13 return  $\mathcal{P}_k$ 

```

---

who are all students of Computer Science. Half of them are “novice” users who don’t know the location, and the other half are “experts”. Participants should fulfill a task in two different frameworks: GEOPOLY and TABLEAU. We choose TABLEAU as our competitor as it is the most advanced off-the-shelf visualization product for analyzing spatial data. For each participant, we report a variant of time-to-insight measure, i.e., the number of times that participants interact with the frameworks before fulfilling the task. Evidently, less number of interactions is preferred as it means that the participant can reach insights faster.

**Table 3: Nb. interactions for “novice”**

	T1/I1	T2/I1	T1/I2	T2/I2
GEOPOLY	14.5	10.1	16.5	11.3
TABLEAU	61.3	57.6	67.2	55.7

**Table 4: Nb. interactions for “expert”**

	T1/I1	T2/I1	T1/I2	T2/I2
GEOPOLY	10.1	6.3	9.3	7.1
TABLEAU	56.7	49.1	54.6	43.4

On the Airbnb dataset of Paris with XXX points, we define two different tasks: *T1*: “finding a point in a requested location” (e.g., find a home-stay in the Trocadero area of Paris), and *T2*: “finding a point with a requested profile” (e.g., find a cheap home-stay with balcony.) Participants may also begin their navigation either from *I1*: “close to the goal” or *I2*: “far from the goal”.

In TABLEAU, participants employ filtering and querying tools to reach their goals. In GEOPOLY, participants benefit from information highlighting based on their implicit feedback. Tables 3 and 4 report the number of interactions for novice and expert participants, respectively. We observe that on average 10.65 interactions are needed to reach a defined goal in GEOPOLY, which is which is 5.23 times smaller than the average number of interactions for TABLEAU’s. This shows that implicit feedback capturing is an effective mechanism which helps analysts to reach their goals faster.

In GEOPOLY, expert participants need 4.9 fewer interactions on average. Interestingly, starting points, i.e., *I1* and *I2*, do not have a huge impact on number of steps. It is potentially due to the diversity component which provides distinct options and can quickly guide analyst towards their region of interest. We also observe that the task *T2* is an easier task than *T1*. This is potentially due to the similarity component where the analyst can request options similar to what she has already observed and greedily move to her preferred regions.

In the second part of our experiments, we briefly discuss the performance of GEOPOLY. We will provide details of our performance study in an extended version of this paper. We execute our algorithm on 10,000 points of Airbnb dataset and  $k = 3$ . We report execution time as the average of 20 runs. In general, GEOPOLY needs XXX seconds to deliver  $k$  highlights. Increasing  $k$  has a linear effect on the execution time. We also observe that increasing the number of points influences the overall time by a XXX slope. The bottle neck of GEOPOLY is XXX:WHICHSTEP. Limiting time in Algorithm 3 shows that even with restricted time limit to 200ms, the algorithm often reaches more than 85% of diversity.

## 6. RELATED WORK

To the best of our knowledge, the problem of spatial information highlighting using implicit feedback has been never addressed in the literature. However, our work relates to few others in their semantics.

**Information Highlighting.** The literature contains few instances of information highlighting approaches [19, 25, 30, 29]. However, all these methods are objective, i.e., they assume that analyst’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 analyst’s feedback. In GEOPOLY, the feedback vector gets updated in time based on the implicit feedback of the analyst.

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 few approaches focus on the spatial dimension [5, 18], they still lack the evolutionary feedback capturing. Moreover, most recommendation methods miss “result diversification”, i.e., highlights may not be effective due to overlaps.

**Feedback Capturing.** Several approaches are proposed in the state of the art for capturing different forms of feedback [8, 31, 12, 17, 23, 10]. The common approach is a top- $k$  processing methodology in order to prune the search space based on the explicit feedback of the analyst and recommend a small subset of interesting results of size  $k$ . A clear distinction of GEOPOLY is that it doesn’t aim for pruning, but leveraging the actual data with potential interesting results (i.e., highlights) that the analyst may miss due to the huge volume of spatial data. Moreover, in a typical top- $k$  processing algorithm, analyst’s choices are limited to  $k$ . On the contrary, GEOPOLY enables a freedom of choice where highlights get seamlessly updated with new analyst choices.

Few works formulate fusing approaches of explicit and implicit feedbacks to better capture user preferences [1, 4, 21].

GEOPOLY functions purely on implicit feedback and does not require any sort of explicit signal from the analyst.

**Region Discovery.** GEOPOLY finds interesting dense regions (IDRs) in order derive analyst’s implicit preferences. There exist several approaches to infer a spatial region for a given set of points [7, 13, 15, 2, 16, 6]. The common approach is to cluster points in form of concave and convex polygons. In [7], an algorithm is proposed to verify if a given point  $p$  on the surface of a sphere is located inside, outside, or along the border of an arbitrary spherical polygon. In [13, 15], a non-convex polygon is constructed from a set of input points on a plane. In [2, 16], imprecise regions are delineated into a convex or concave polygon. In GEOPOLY, it is important to discover regions in a way to only capture mouse move points. In case a concave polygon is constructed, the “dents” of such a polygon may entail points which are not necessarily in  $\mathcal{M}$ . In GEOPOLY, however, we adapt Quickhull [6], due its simplicity, efficiency and it’s natural implementation of convex polygons.

## 7. CONCLUSION

In this paper, we address the problem of spatial information highlighting using implicit feedback. We formulate our problem in form of a constrained optimization problem and propose GEOPOLY, a novel polygon-based feedback capturing approach which returns few highlights in line with analyst’s implicit preferences. GEOPOLY derives the implicit feedback by finding interesting dense regions (IDRs) from mouse movements of the analyst over the geographical map.

We consider a few directions of improvement for this work. We are interested to incorporate an “explainability” component into GEOPOLY which can describe causalities behind preferences. For instance, we are interested to find seasonal patterns to see why the preferences of analysts change from place to place during various seasons of the year. Another direction is to incorporate “Query by Visualization” approaches into GEOPOLY where analysts can specify their intents alongside their implicit preferences, directly on the map [27].

## 8. REFERENCES

- [1] E. M. Aoidh, M. Bertolotto, and D. 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.
- [2] A. Arampatzis, M. van Kreveld, I. Reinbacher, C. B. Jones, S. Vaid, P. Clough, H. Joho, and M. Sanderson. Web-based delineation of imprecise regions. *Computers, Environment and Urban Systems*, 30(4):436 – 459, 2006. Geographic Information Retrieval (GIR).
- [3] I. Arapakis, M. Lalmas, and G. 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.
- [4] A. Ballatore and M. Bertolotto. Semantically enriching vgi in support of implicit feedback analysis.

- In K. Tanaka, P. Fröhlich, and K.-S. Kim, editors, *Web and Wireless Geographical Information Systems*, pages 78–93, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
- [5] J. Bao, Y. Zheng, D. Wilkie, and M. Mokbel. Recommendations in location-based social networks: a survey. *GeoInformatica*, 19(3):525–565, 2015.
  - [6] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, Dec. 1996.
  - [7] M. Bevis and J.-L. Chatelain. Locating a point on a spherical surface relative to a spherical polygon of arbitrary shape. *Mathematical Geology*, 21(8):811–828, Oct 1989.
  - [8] M. Bhuiyan, S. Mukhopadhyay, and M. A. 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.
  - [9] D. Birant and A. Kut. St-dbscan: An algorithm for clustering spatial-temporal data. *Data Knowl. Eng.*, 60(1):208–221, Jan. 2007.
  - [10] M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. 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.
  - [11] M. C. Chen, J. R. Anderson, and M. H. 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.
  - [12] K. Dimitriadou, O. Papaemmanoil, and Y. Diao. Aide: an active learning-based approach for interactive data exploration. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2842–2856, 2016.
  - [13] M. Duckham, L. Kulik, M. Worboys, and A. 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.
  - [14] M. Ester, H.-P. Kriegel, J. Sander, and X. 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.
  - [15] M. Fadili, M. Melkemi, and A. ElMoataz. Non-convex onion-peeling using a shape hull algorithm. *Pattern Recognition Letters*, 25(14):1577 – 1585, 2004.
  - [16] A. Galton and M. Duckham. What is the region occupied by a set of points? In M. Raubal, H. J. Miller, A. U. Frank, and M. F. Goodchild, editors, *Geographic Information Science*, pages 81–98, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
  - [17] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi. Distributed and interactive cube exploration. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, pages 472–483. IEEE, 2014.
  - [18] J. J. Levandoski, M. Sarwat, A. Eldawy, and M. F. Mokbel. Lars: A location-aware recommender system. In *ICDE*, pages 450–461, 2012.
  - [19] J. Liang and M. L. Huang. Highlighting in information visualization: A survey. In *2010 14th International Conference Information Visualisation*, July 2010.
  - [20] L. Lins, J. T. Klosowski, and C. Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.
  - [21] N. N. Liu, E. W. Xiang, M. Zhao, and Q. 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.
  - [22] B. Omidvar-Tehrani, S. Amer-Yahia, and A. Termier. Interactive user group analysis. In *CIKM*, pages 403–412. ACM, 2015.
  - [23] B. Omidvar-Tehrani, S. Amer-Yahia, and A. Termier. Interactive user group analysis. In *CIKM*, pages 403–412. ACM, 2015.
  - [24] B. Omidvar-Tehrani, P. A. S. Neto, F. M. F. Pontes, and F. 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.
  - [25] A. C. Robinson. Highlighting in geovisualization. *Cartography and Geographic Information Science*, 38(4):373–383, 2011.
  - [26] J. F. Roddick, M. J. Egenhofer, E. G. Hoel, D. Papadias, and B. Salzberg. Spatial, temporal and spatio-temporal databases - hot issues and directions for phd research. *SIGMOD Record*, 33(2):126–131, 2004.
  - [27] T. Siddiqui, A. Kim, J. Lee, K. Karahalios, and A. Parameswaran. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment*, 10(4):457–468, 2016.
  - [28] A. Telang, D. Padmanabhan, and P. 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.
  - [29] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1129–1136, 2007.
  - [30] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *TVCG*, 22(1), 2016.
  - [31] D. Xin, X. Shen, Q. Mei, and J. 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.