MAGE: Discovering Mixture-based Areas of Interest over Geolocated Entities

Kostas Patroumpas
Athena Research Center
Athens, Greece
kpatro@athenarc.gr

Dimitrios Skoutas
Athena Research Center
Athens, Greece
dskoutas@athenarc.gr

Dimitris Sacharidis
Université Libre de Bruxelles
Brussels, Belgium
dimitris.sacharidis@ulb.be

ABSTRACT

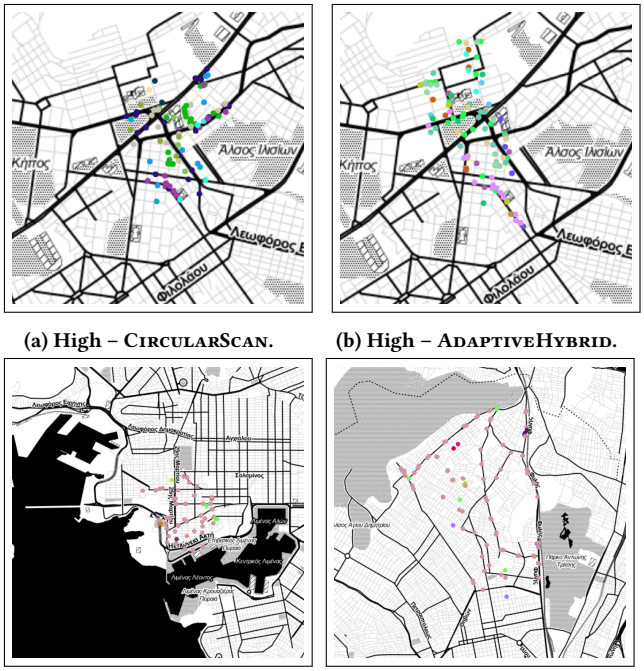
We demonstrate MAGE, an open-source tool for mixture-based best region search over geolocated entities of different types, like Points of Interest, geotagged posts or photos. MAGE detects the top-k areas of arbitrary shapes exhibiting high or low mixture patterns. Through a graphical interface, we show how users can specify their preferences, execute the selected algorithm, and visually inspect the results on map to unveil interesting patterns.

1 INTRODUCTION

Geolocated entities are associated with a *location* and can be of different *types*. Examples include Points of Interest (POIs), such as shops, restaurants, museums, or geotagged posts, such as photos or tweets described by keywords. In the former, the type is the category. In the latter, we can use a technique such as LDA [2] to extract a set of topics, and then associate each entity with a vector indicating a distribution over these topics. Given a collection of geolocated entities, detecting *Areas of Interest* (AOIs) can reveal interesting insights for many applications, including geomarketing, logistics, and urban planning. Hence, many methods to detect AOIs based on various criteria exist (e.g., [3–5]).

In this work, we present MAGE, an open-source tool for detecting AOIs that exhibit interesting (i.e., high or low) *spatial mixture patterns* with respect to the types of entities located in them. In particular, *high-mixture* AOIs include a wide variety of amenities and services, as denoted by the different colors of the POIs shown in Figure 1a and Figure 1b. For instance, such AOIs may attract customers seeking to purchase a new residence. In contrast, *low-mixture* AOIs (Figures 1c, 1d) are dominated by particular type(s) and reveal regions that are mostly dedicated to specific functions like a business district or an industrial zone. Previous works for detecting such patterns are limited to regions of fixed shape (e.g., circle or rectangle) [7]. However, fixed-shape regions may not accurately reflect the actual AOIs occurring in the real world, which often have arbitrary shapes due to natural barriers (e.g., lakes, rivers) or man-made structures (e.g., pedestrian streets).

To overcome this limitation, MAGE employs our graph-based approach and anytime algorithms for detecting *mixture-based AOIs of irregular shapes* [6]. It extends these algorithms to enable discovery of diversified top-k regions, offering results that are more suitable for exploration. MAGE includes a graphical interface to select the input dataset, choose the algorithm to be used, specify input parameters, perform the computation, and visualize the results on a map. By choosing different algorithms or parameters, users may obtain regions with differing characteristics (i.e., in shape or size), thus being able to unveil and explore different spatial mixture patterns in the data.



(a) High – CircularScan. (b) High – AdaptiveHybrid.

(c) Low – CircularScan. (d) Low – ExpandBest.

Figure 1: High- and low-mixture best AOI in Athens detected by different methods on POIs from OpenStreetMap.

2 MIXTURE-BASED AOIS

Assume a collection $\mathcal{D}$ of point entities located in a 2-dimensional space with each entity $p \in \mathcal{D}$ belonging to one or more classes $\tau \in \mathcal{T}$. An area (or region) of interest R is a subset of points in $\mathcal{D}$. We define the *interestingess score* of R as $\sigma(R) = \phi(R) \cdot (|R| \,/\, M)^{\gamma}$, where $\phi(R)$ measures the diversity of the types of entities in R, M is the maximum allowed region size, and γ is the weight of the size factor. Intuitively, we would like to favor larger regions compared to smaller ones, since mixture patterns involving just a few entities may occur merely by chance. Like previous work [7], we define function ϕ as Shannon's entropy.

To discover AOIs of arbitrary shapes, rather than fixed shapes such as rectangles or circles [7], we have proposed a graph-based approach along with certain properties to ensure that detected AOIs are meaningful [6]. Given a user-defined distance threshold ϵ, we first construct a *spatial connectivity graph* $G = (V, E)$ over the collection $\mathcal{D}$, where V is the set of entities and E is the set of edges such that $(u, v) \in E$ if $d(u, v) \leq \epsilon$. Without loss of generality, we assume d to be the Euclidean distance; however, other functions may be employed, such as road network distance.

A region R is a subgraph $G_R = (V_{R_C} \cup V_{R_B}, E_R)$ of G, where V_{R_C} and V_{R_B} are *core* and *border* points, respectively, adhering to three conditions. To ensure *cohesiveness* of R: (C1) the core points must form a connected subgraph; and (C2) each border point must be

connected to at least one core point. To ensure *completeness*, (C3) all neighbors of a core point must be included in region R. This prevents trivially 'hand-picking' subsets of points to form AOIs that artificially exhibit a high or low mixture pattern.

Given a spatial connectivity graph G representing a collection $\mathcal{D}$ of entities belonging to different types $\mathcal{T}$, and a maximum region size M, the goal is to find the region R with the maximum score $\sigma(R)$. A key observation is that the score of a region does not exhibit monotonicity with respect to its size, since its entropy may either increase or decrease after each expansion depending on how the types of the newly enclosed entities compare to those of the already contained ones. Thus, algorithms proposed for the best region search problem [5] or Apriori-like algorithms [1] cannot be applied. Moreover, exhaustively enumerating all subgraphs of G to identify and evaluate candidate regions is clearly infeasible for large, real-world datasets. Consequently, to address this problem, we have designed anytime algorithms, i.e., algorithms that aim at detecting regions of as high score as possible under any given time budget T.

3 SEARCH ALGORITHMS

Next, we outline the basic aspects of the algorithms for detecting mixture-based AOIs; full details and an extensive evaluation can be found in [6]. In order to run in an anytime manner, all algorithms employ a priority queue Q where each generated candidate AOI is pushed according to its score. Thus, instead of fully expanding all possible candidate regions around a given point (node in G) before moving on to the next one, each algorithm picks the next candidate AOI to expand in decreasing order of their current score. Despite the extra overhead of queue maintenance, this approach increases the likelihood of detecting top scoring AOIs as early as possible. For clarity, we first present how each method discovers the mixture-based best (i.e., top-1) region Then, we discuss how to tackle searching for the top-k AOIs.

3.1 Fixed-shape Scan

Our starting point is a baseline algorithm that enumerates candidate regions relying on a regular geometric shape. Specifically, we employ circles for such a fixed-shape scan of the search space, so we call this method CircularScan. As in [7], candidate regions are generated by exhaustively enumerating circles of increasing radius between a point $p \in \mathcal{D}$ picked as their center and another point $p' \in \mathcal{D}$ on their circumference. If the number of points in $\mathcal{D}$ is large, it is possible to use only a subset $S \subseteq \mathcal{D}$ of them as centers (seeds), e.g., via uniform sampling.

We further enhance this baseline with two adaptations. First, as already explained, a priority queue Q maintains the currently active candidate AOIs to check until expiration of time budget T. Initially, each AOI in Q contains just its center and its nearest neighbor point from $\mathcal{D}$. After the top-scoring candidate region R^* is probed for expansion with the next neighboring point from $\mathcal{D}$, it may be pushed back to Q with an updated score if it currently contains less than M points. The second adaptation concerns validation of candidate AOIs. To satisfy *cohesiveness*, when enumerating circles around a center p, we only consider neighboring points within distance ϵ from at least one of the points in the current candidate R around p. If the next neighbor point is farther than ϵ from the last neighbor inserted in R, this candidate region R cannot be further expanded with other points; the accumulated points constitute the set of core points of R. To satisfy *completeness*, any neighbor of a core point that is not already part of R is retrieved; these points form the border of R. If R has higher score than the best region R^* found so far at a previous iteration, then R itself becomes the current best region R^*.

3.2 Graph Expansion Strategies

CircularScan produces AOIs of regular shape (i.e., circle-like regions). We next discuss two strategies based on subgraph expansion in the connectivity graph G to detect the best arbitrarily shaped region R^*. This graph expansion approach also relies on a priority queue Q to enumerate candidate regions and executes in two phases, namely initialization and expansion.

The initialization phase first selects a set S of seed points. Using uniform random sampling, we pick $\rho \cdot |V_G|$ nodes of the spatial connectivity graph G, where $\rho \in (0, 1]$ is a parameter of the algorithm. Each seed point is then visited to initialize its corresponding valid region R and assign a score to it. For each seed node v, its AOI R comprises v as its core point and all the neighbors of v in G as its border points. In a post-processing step, the border of R is refined: border points that are actually core points in R are removed from the border set of R to avoid unnecessary computations during the expansion phase. Finally, if R does not exceed the maximum allowed size M and the region border is not empty, then R is inserted to the queue.

Next, the expansion phase iterates over the candidate regions in Q until either there are no remaining entries in Q or the time budget T is exhausted. Once a region R is popped from Q, one or more border points in R are selected to be expanded, which respectively generate one or more new candidate regions, whose nodes are a superset of R. Depending on how many and which border points to expand, we employ two alternative strategies: ExpandAll selects all border points, whereas ExpandBest selects the border point which leads to the highest scoring region once expanded. In ExpandAll, the entire border is expanded in each iteration, hence the maximum region size M is reached much faster, i.e., the expansion for each given seed will terminate faster. Instead, in ExpandBest, candidate AOIs grow at a slower pace, but the algorithm can select to expand only favorable border points. Both expansion strategies generate exactly one new candidate region from the existing R. This new R becomes the best region R^* if it contains less than M points and scores higher than the currently known R^*. Also, if the border of R is not empty, R is pushed to Q to be further expanded in a future iteration.

3.3 Adaptive Strategies

All previous methods rely on a priori selection of a fixed set S of seed points. Yet, suitably placing those seeds is crucial in order to return a good solution within a limited time budget T. Picking more seeds could alleviate the problem but has the downside of incurring higher execution cost. Hence, the number of utilized seeds needs to be small, but their allocation should be more effective. To achieve this, we select seeds dynamically and adaptively during evaluation employing *seed areas*. Instead of picking all seeds at once from the entire study area, we take a few seeds as starting points to gain some initial insight about the dataset, and then dynamically adapt the selection of subsequent seeds accordingly. In [6], we suggested two adaptive search strategies, each defining its own kind of seed areas.

Strategy AdaptiveHybrid essentially combines ExpandAll and ExpandBest in two steps. First, it randomly selects a small number of starting seed points. For each such seed, its corresponding AOI is iteratively expanded using ExpandAll until either

the maximum size M is reached or no other neighbor nodes exist. For each initial seed, its best region is marked as a seed area and is pushed to queue Q. During the second step, seed areas are popped from Q according to their score. For each such area, one of its points is randomly selected as a new seed. Intuitively, seeds drawn from areas already having a relatively high score are more likely to yield even more high-scoring regions. The selected seed is then expanded by applying ExpandBest, but removed from the set of points in that seed area before pushing it back to Q with its updated score found during expansion. If the seed selected from that area has yielded a high-scoring region, additional seeds will be drawn from the same area to examine it more thoroughly; otherwise, its score will diminish, and seeds will be drawn from another, more promising area with a higher score.

In strategy AdaptiveGrid, seed areas are the cells of a uniform, coarse-granularity grid applied over the dataset $\mathcal{D}$. The idea is to locate seeds that are uniformly distributed and thus avoid oversampling/undersampling a dense/sparse area. In the first step, it takes a small number of seeds by picking grid cells uniformly at random, and then also randomly choosing a seed point within each cell. ExpandAll is invoked for each selected seed point, and the resulting score is assigned to its corresponding cell. The cells with the k highest scores are designated as seed areas for the second step, where a fine-granularity grid is constructed within each qualified seed area. A series of seed probes is performed until expiration of time budget T. Each such probe successively selects a seed area, then a cell within its fine-granularity grid, and finally a point within that cell; all selections are done uniformly at random. A seed selected in this phase is expanded more thoroughly using process ExpandBest.

3.4 Extensions

3.4.1 Finding top-k AOIs. The focus in the previously discussed algorithms was on detecting the best mixture-based AOI in the input data. As all algorithms employ a priority queue for maintaining the candidate regions, this approach can be extended to return the top-k AOIs for a user-defined k, i.e., the k regions with the highest scores. Still, it may frequently occur that the returned AOIs cover almost the same entities with small deviations in size and slightly different (yet high) scores due to inclusion of different border points nearby. To avoid this effect, given that such top-k results have small practical significance, we introduce an extra parameter concerning the maximum allowed *overlap* $o \in [0, 1]$ between a pair of AOIs. During processing, if the percentage of common entities between a new candidate AOI and another already in the priority queue exceeds this threshold o, we choose to keep in the queue the candidate with the higher score. As a result, the returned top-k AOIs may have common entities (always less than the specified o threshold), but are generally placed in different areas and are more diversified in shape.

3.4.2 Coarse-grained AOIs. Since all algorithms are anytime and need to find mixture-based AOIs within a time budget T, results may be poor in size and score if T is small, especially for larger datasets. Clearly, there is a trade-off between the time budget and the quality of results. Thus, we offer the option for coarser AOIs that can be discovered faster. In a preprocessing stage, entities are aggregated into cells of side ϵ with a uniform grid partitioning applied over the input dataset $\mathcal{D}$. Each grid cell abstracts the distribution of entity types therein and graph G represents those cells instead of the original entities; cells with common sides or vertices are connected in G, which is then used in subsequent processing. Of course, the returned AOIs are composed of such adjacent cells and are less detailed in shape. Nevertheless, this approach can quickly provide hints for interesting mixture patterns in certain areas; users may then drill down and run the discovery process on such smaller parts of the data to get finer results consisting of original entities.

4 USER INTERFACE

MAGE is implemented in Python, and the code is publicly available[1]. Through a graphical interface (Figure 2), users can easily load a dataset of geolocated entities, preprocess it, specify their preferences regarding the method, its parameters and time budget, execute the process to discover mixture-based AOIs, and inspect the results on the map. Next, we outline these steps.

Load. The user first specifies the input dataset. We currently support CSV files but extension to other file formats is straightforward. As shown in Figure 2a, the input parameters concern the path to the file, the column delimiter, the columns corresponding to the entity identifier, name, coordinates, and keywords, as well as the separator used in the keywords column. If needed, the user can apply data transformation between coordinate reference systems. Users may also optionally apply a spatial filter on the input entities by specifying a bounding box designating the study area.

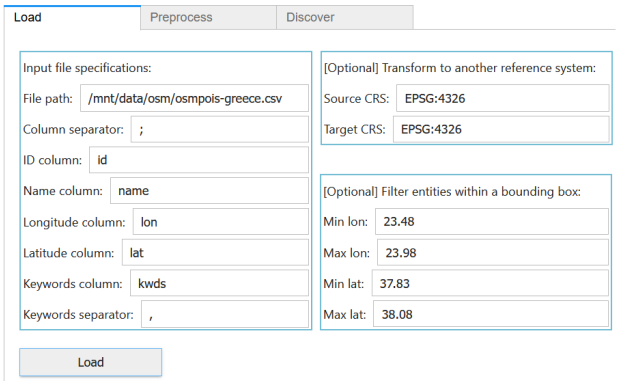
Preprocess. The second stage (Figure 2b) involves construction of the spatial connectivity graph G. Pairs of entities within the specified distance threshold ϵ are connected with an edge in this graph. Optionally, for faster discovery over large datasets, input entities may be aggregated into cells of width and height ϵ, which is enabled by selecting the "Apply grid partitioning" option. In that case, nodes in G correspond to cells, while edges indicate adjacent cells. Also, in case the input entities are not originally classified into categories but each is only tagged with one or more keywords, it is possible to apply LDA [2] in order to automatically assign a vector of topics per entity; once users check this latter option, they must also specify the number of topics. After triggering graph construction, statistics on its properties are shown once preprocessing is complete.

Discover. Once the graph is constructed, it can be reused under any user-specified parameter settings for any search algorithm (Figure 2c). First, the maximum size (i.e., number of points or cells) of an AOI is set. Moreover, a size weight is specified, denoting the relative importance of AOI size versus entropy in the score calculation. A time budget is indicated, dictating the maximum allowed execution time for the algorithm. The entropy mode (high or low) must be chosen, as well as the search algorithm to be applied. Other parameters include the percentage of the initial entities to be used as *initial seeds* for the discovery; the maximum allowed degree of *overlap* between two regions expressed as percentage of common entities; and the number k of AOIs with the highest scores to return. Once the discovery process is triggered, the best AOIs are returned and visualized on map with statistics per region (rank, score, actual size).

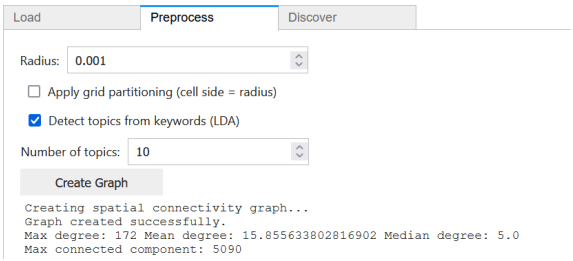
5 DEMONSTRATION

We have used MAGE to discover mixture-based AOIs over several open and proprietary real-world datasets. We will demonstrate MAGE on open data, including: (i) POIs extracted from OpenStreetMap[2]; from this large collection of 21 million POIs worldwide belonging to 15 thematic categories (e.g., transport, shop,

[1] https://github.com/smartdatalake/mbrs
[2] http://download.slipo.eu/results/osm-to-csv/



(a) Loading the input dataset.



(b) Preprocessing data.

(c) Discovering top-k mixture-based AoIs.

Figure 2: User interface.

food, education, etc.), we will focus on metropolitan areas (e.g., Athens, London, New York). (ii) Crime data available at monthly updates from the UK police[3]; we will use crime incidents in London, characterized by 16 different crime types (e.g., anti-social behavior, burglary, theft, robbery, etc.). (iii) Geotagged photos extracted from Flickr[4] for the greater London area. Entities in this dataset are not classified in categories, but are labelled with a variety of keywords, so LDA will be used to derive the topics.

For each dataset, we will first show how to construct the spatial connectivity graph with a user-specified distance threshold ϵ. We will also exemplify how to speed up the discovery process for larger datasets by first aggregating individual points into cells of a uniform grid. In that case, the graph is computed over the underlying grid, resulting in more coarse-grained AOIs that consist of adjacent cells rather than individual points.

We will demonstrate that the resulting AOIs may differ depending on the chosen discovery algorithm and its parameterization. In general, the fixed-shape CircularScan tends to detect more compact regions, smaller both in size (i.e., number of points) and spatial extent. For example, Figure 1a shows the best high-mixture AOI, which consists of various types of POIs from OpenStreetMap in Athens. Note that this AOI has an almost circular shape, since it has been expanded with POIs around a randomly chosen initial seed. The best low-mixture AOI identified by CircularScan also looks circle-like (Figure 1c); clearly, most of the POIs in this region belong to a certain category (transport) with only a few points from other categories. In contrast, the graph expansion methods typically yield AOIs of higher scores and more varying shapes. Note that the best high-mixture AOI found by AdaptiveHybrid (Figure 1b) is also located in the same neighborhood as the one detected by CircularScan (Figure 1a), but it is more extended and has an elongated shape with many POIs across major roads. Figure 1d shows the best low-mixture AOI identified by ExpandBest. This non-convex region covers a much larger area compared to the one identified by CircularScan in Figure 1c (both maps are in the same scale). With graph expansion methods, a region can adapt its shape to better capture the underlying mixture pattern, thus detecting larger AOIs of irregular shape while still satisfying the conditions for cohesiveness and completeness. Moreover, the adaptive methods can return more enlarged AOIs with shapes that better capture the underlying high/low mixture pattern, usually non-convex and sometimes rather elongated shapes (e.g., POIs along a main road and smaller streets nearby).

Finally, we will show the impact of the various parameters, such as the maximum region size M and the weight γ of the region size, to the performance of the algorithms and the quality of the results. We will also point out how the methods employing adaptive seed prioritization tend to be more robust with respect to the inherent randomness in the choice of initial seeds.

[3]https://data.police.uk/
[4]https://code.flickr.net/category/geo/

ACKNOWLEDGMENTS

This work was supported by the EU H2020 project SmartDataLake (825041).

REFERENCES

[1] Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast Algorithms for Mining Association Rules in Large Databases. In *VLDB*. 487–499.
[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3 (2003), 993–1022.
[3] Xin Cao, Gao Cong, Christian S. Jensen, and Man Lung Yiu. 2014. Retrieving Regions of Interest for User Exploration. *Proc. VLDB Endow.* 7, 9 (2014), 733–744.
[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *SIGKDD*. 226–231.
[5] Dimitrios Skoutas, Dimitris Sacharidis, and Kostas Patroumpas. 2018. Efficient progressive and diversified top-k best region search. In *SIGSPATIAL*. 299–308.
[6] Dimitrios Skoutas, Dimitris Sacharidis, and Kostas Patroumpas. 2021. Discovering Mixture-Based Best Regions of Arbitrary Shapes. In *SIGSPATIAL*. 468–479.
[7] Yiqun Xie, Han Bao, Yan Li, and Shashi Shekhar. 2020. Discovering Spatial Mixture Patterns of Interest. In *SIGSPATIAL*. 608–617.