SPATIAL AND SPATIO-TEMPORAL CLUSTERING

A Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Sujing Wang

May 2014

SPATIAL AND SPATIO-TEMPORAL CLUSTERING

Sujing Wang

APPROVED:

Dr. Christoph F. Eick, Chairman
Dept. of Computer Science

Dr. Guoning Chen
Dept. of Computer Science

Dr. Ernst Leiss
Dept. of Computer Science

Dr. Ricardo Vilalta
Dept. of Computer Science

Dr. Klaus Kaiser
Dept. of Mathematics

Dean, College of Natural Sciences and Mathematics

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Dr. Christoph Eick, for his continuous support of my Ph.D. study and research, and for his patient guidance, encouragement, and useful reviews. My sincere thanks also goes to Dr. Ernst Leiss, Dr. Ricardo Vilalta, Dr. Guoning Chen, and Dr. Klaus Kaiser for their encouragement and insightful comments for my Ph.D. research. Last but not least, I would like to thank my family. They are always supporting me and encouraging me with their best wishes.

SPATIAL AND SPATIO-TEMPORAL CLUSTERING

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Sujing Wang

May 2014

Abstract

Due to the advances in technology, such as smart phones, general mobile devices, remote sensors, and sensor networks, different types of spatial data become increasingly available. These data can also integrate multiple other types of information, such as temporal information, social information, and scientific measurements, which provide a tremendous potential for discovering new useful knowledge, as well as new research challenges. In this research, we focus on clustering and analyzing spatial and spatio-temporal data. We have addressed several important sub-problems in polygon-based spatial and spatio-temporal clustering and post-processing analysis techniques. We have developed (1) two distance functions that measure the distances between polygons, especially overlapping polygons; (2) a density-based spatial clustering algorithm for polygons; (3) two post-processing analysis techniques to extract interesting patterns and useful knowledge from spatial clusters; (4) two density-based spatio-temporal clustering algorithms for polygons; (5) a box plot based post-processing analysis technique to identify interesting spatio-temporal clusters of polygons; (6) a change-pattern-discovery algorithm to detect and analyze patterns of dynamic changes within spatio-temporal clusters of polygons; and (7) a formal definition of the task of finding uniform regions in spatial data and an algorithm to identify such uniform regions. Our algorithms and techniques are demonstrated and evaluated in challenging real-world case studies involving ozone pollution events in the Houston-Galveston-Brazoria area and the building data of Strasbourg, France. The results show that our algorithms are effective in finding compact clusters in spatial and spatio-temporal domains and in extracting interesting patterns and useful information from spatial and spatio-temporal data.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The complete spatial data-mining process is interactive and iterative, involving many tasks, such as data extraction and cleaning, feature selection, algorithm design, and post-processing analysis of the output when an algorithm is applied to the data. The goal of spatial data mining is to automate the discoveries of interesting patterns and potentially useful information, which can be examined by domain experts for further validation and verification. Spatial clustering is one of the most fundamental tasks in spatial data mining. It is the process of grouping objects into different groups known as clusters such that the objects within the same group are similar to each other but dissimilar from those in other groups.Clusters are generated on the basis of a "similarity" criterion which is used to determine the relationship between each pair of objects in the datasets. Post-processing analysis techniques deal with both spatial and non-spatial characteristics of spatial objects.

A spatial dataset contains objects characterized by a spatial location as well as

non-spatial attributes. The notion of spatial auto-correlation, namely that similar objects tend to cluster in geographic space, is central to spatial data mining. The principles of independence that are assumed in traditional data mining algorithms no longer apply in spatial data mining; therefore, spatial clustering is more difficult than traditional clustering due to the complexity of spatial data types, spatial relationships, and spatial auto-correlation.

This dissertation focuses on the design of clustering algorithms and post-processing analysis techniques for mining spatial and spatio-temporal datasets. In particular, we have developed a spatial clustering algorithm, two spatio-temporal clustering algorithms, and several post-processing analysis techniques for mining complex types of spatial objects, mostly centering on polygons.

1.1 Motivation of research

Due to the advances in remote sensors and sensor networks, different types of spatial data become increasingly available. The spatial data contain not only non-spatial attributes such as traditional numeric and categorical data, but also spatial attributes, e.g., longitude and latitude. Mining spatial data is a very challenging task. The separation of spatial and non-spatial attributes in spatial datasets poses new challenges because patterns and summaries have to be obtained frequently in the spatial subspace whereas the non-spatial attributes play key roles in determining the interestingness of the obtained patterns and summaries. Spatial data can also integrate multiple other types of information, such as temporal information, social information,

and scientific measurements, which provide a tremendous potential for discovering useful knowledge, as well as new research challenges.

Traditional clustering techniques are inefficient in mining spatial or spatial-temporal data because they do not incorporate the idiosyncrasies of the spatial and temporal domains; therefore, new techniques are needed to address these challenges and to provide effective solutions to analyze and mine large spatial or spatio-temporal data. As expected, extracting interesting patterns from such data is very important for many applications, such as geographic information systems, weather forecasting, medical imaging, environment protection, and urban computing.

Several types of spatial data are available in real applications, i.e., points, trajectories, and polygons. Polygons are more complex than points because polygons have topological properties that are not relevant to points. Traditional clustering algorithms for points and trajectories do not work directly for polygons. Moreover, polygons provide natural representations for many types of geo-referenced objects, such as countries, buildings, and pollution hotspots, which can be used by a broad set of applications, such as air pollution prevention, health-care study, and urban planning; therefore, it is very important to develop data mining techniques focusing on mining polygon-based spatial and spatio-temporal data, such as clustering and post-processing analysis techniques.

1.2 Research contributions

In this research, we have made several significant contributions to the state of the art in spatial and spatio-temporal clustering. They are briefly summarized below.

1. Distance functions for polygons - We have introduced two distance functions that can efficiently measure the spatial distances between polygons, especially overlapping polygons and polygons with holes.

2. Density-based spatial clustering algorithm - We have developed a density-based spatial clustering algorithm for polygons that extends the density-concepts of the Shared Nearest Neighbor clustering algorithm from points to polygons.

3. Domain-driven final clustering generation method - We have investigated an algorithm which generates the final clusterings by selecting at most one polygon from the meta-clusters to maximize a reward function that captures domain experts' notions of interestingness.

4. Post-processing analysis techniques for spatial data - We have developed algorithms with several plug-in interestingness functions to automatically identify spatial clusters whose member distribution with respect to a non-spatial variable deviates significantly from the distribution in the whole dataset.

5. Density-based spatio-temporal clustering algorithm - We have introduced two density-based spatio-temporal clustering algorithms for polygons by taking into account both the spatial and temporal dimensions of polygons.

6. Change-pattern-discovery algorithm - We have developed a change-pattern-discovery algorithm to detect and analyze dynamic patterns of changes within spatio-temporal clusters of polygons.

7. Post-processing analysis technique for spatio-temporal data - We have implemented a box plot based post-processing analysis technique for spatio-temporal data to identify interesting spatio-temporal clusters of polygons.

8. Identifying uniform regions in spatial data - We have formally defined the task of finding uniform regions in spatial data as solving a maximization problem for a plug-in measure of uniformity and have designed a spatial clustering framework to identify uniform regions.

9. Popular signatures - We have proposed popular signatures, which are distribution signatures frequently occurring in the subspaces of a spatial area of interest. A novel approach which summarizes the composition of a spatial dataset by annotating regions with popular signatures is presented as well.

1.3 Dissertation organization

The remainder of the dissertation is organized as follows: Chapter 2 discusses the literature review. Chapter 3 describes the details of our polygon-based clustering and analysis framework for mining spatial datasets, which includes two distance functions for polygons, a spatial clustering algorithm, and several post-processing

analysis techniques. We also present the results obtained by applying our algorithms on challenging real-world case studies involving ozone pollution events in the Houston-Galveston-Brazoria (HGB) area. Chapter 4 presents two density-based spatio-temporal clustering algorithms for polygons. We also explain the application of these two algorithms on ozone pollution datasets in order to detect the density-connected clusters of polygons in both spatial and temporal domains. A change-pattern-discovery algorithm that detects and analyzes dynamic patterns of changes within spatio-temporal clusters of polygons, and a post-processing analysis technique to extract interesting patterns and useful knowledge from spatio-temporal clusters of polygons for domain experts, are presented as well. Chapter 5 discusses the formal definition of the task of finding uniform regions in spatial data and a prototype-based clustering algorithm to find such regions. This chapters also shows the results of the application of our algorithm for analyzing the composition of the city of Strasbourg in France. Finally, in Chapter 6 we present a summary of our work, along with directions for future research.

Chapter 2

Literature Review

2.1 Distance functions and spatial clustering algorithms for polygons

Distance functions are the key for spatial clustering algorithms. In [1], Joshi et al. introduce a dissimilarity function for clustering polygons that takes into account different characteristics of the polygon separated in different groups: non-spatial attributes, intrinsic spatial attributes, and extrinsic spatial attributes. It is a weighted sum of a number of distance functions each pertaining to a different type attributes of a polygon. Buchin et al. [2] propose a polygonal time algorithm to compute the Fréchet distance between two polygons. Several papers [3, 4] propose algorithms to compute the Hausdorff distance between polygons; however, none of them can cope

with overlapping polygons. Overlapping polygons play a very important role in analyzing large spatial datasets in many application domains. Failing to measure the degree of overlap will result in inadequate clustering results.

Spatial clustering has been a highly active topic in data mining research. Many clustering methods for points and trajectories have been developed. Han et al. [5] classify spatial clustering algorithms into four categories: partitioning methods, hierarchical methods, density-based methods, and grid-based methods. Representative algorithms from each category are introduced as well; however, spatial clustering techniques for polygons are still rarely seen. In [6], Joshi et al. propose a DBSCAN-style spatial clustering algorithm for polygons. The algorithm works by replacing points in the original DBSCAN algorithm with polygons; however, the algorithm in [6] does not cope with overlapping polygons, and the identified spatial clusters of polygons are frequently not contiguous.

2.2 Spatio-temporal clustering algorithms

Spatio-temporal clustering techniques for points and trajectories have been heavily studied in past work as well. Kulldorff [7] introduces basic spatial scan statistics to search spatio-temporal cylinders representing areas where the points occur consistently for a significant amount of time; spatio-temporal cylinders are circular regions occurring within a certain time interval. Iyengar [8] extends the basic spatial scan statistics [7] using flexible square pyramid shapes instead of cylinders for spatio-temporal clusters that can either grow or shrink over time and that can also move

over time. Wang et al. [9] propose two spatio-temporal clustering algorithms, i.e., ST-GRID and ST-DBSCAN. ST-DBSCAN is an extension of the DBSCAN algorithm to perform spatio-temporal clustering by introducing the second parameter of temporal neighborhood radius in addition to the spatial neighborhood radius. ST-GRID is a grid-based clustering approach which maps the spatial and temporal dimensions into cells. Birant et al. [10] also improve DBSCAN for spatio-temporal clustering and apply it to discover spatio-temporal distributions of physical seawater characteristics in Turkish seas. A density factor is assigned to each cluster for detecting some noise points when clusters of different densities exist. The density factor of a cluster captures the degree of the density of the cluster. Rinzivillo et al. [11] propose a progressive clustering approach to analyze the trajectories of moving objects supported by visualization and interaction techniques. It progressively applies different distance functions for spatio-temporal data in each step to optimize the outcome of the algorithm. Li et al. [12] introduced the concept of moving micro-cluster to catch some regularities of a moving object. The micro-clusters are kept geographically small at any time.

Joshi et al. [13] propose a spatio-temporal polygonal clustering algorithm, STPC. STPC extends the DBSCAN algorithm to cluster spatio-temporal polygons by redefining the neighborhood of a polygon as the union of its spatial neighborhood and temporal neighborhood. The temporal aspect is constant or reduced to a fixed interval or time instance when calculating spatial neighbors of a polygon. Moreover, the spatial dimension is instead held to a constant space when calculating temporal neighbors of a polygon; therefore, STPC only clusters polygons that do not change

their locations, sizes, and shapes over time. Only the non-spatial attributes or properties might change with time.

2.3 Finding uniform regions using spatial clustering

Work in [14, 15] proposes a region discovery framework based on a fitness function to be maximized. The framework adapts four representative clustering algorithms, exemplifying grid-based, prototype-based, density-based, and agglomerative clustering algorithms to optimize the fitness function. The fitness function is defined according to the application, and the goal is to model the interestingness of a region. Other work seeks to find uniform regions for spatial regression [16, 17] using quite different methods. Both approaches partition the space into regions, associating different regression functions with different regions. Sheng et al. [18] introduce a search algorithm which finds the top-k regions with a similar distribution of Point of Interest (POI) on a spatial map.

Applegate et al. state that "signatures are compact representation ... that capture important characteristics of massive datasets"[19]; then, they investigate a special family of signatures for multidimensional distributions that represent the distribution of probability mass over a manifold; they also introduce a novel distance function for such signatures. Cortes et al. [20] discuss the use of signatures for mining massive telecommunications data to find communities of interest, and for

fraud detection. Wong et al. [21] demonstrate the benefits of using data signatures to guide the visualization of complex scientific datasets. The use of topic discovery approaches [22, 23] to annotate spatial regions has gained some popularity recently.

Chapter 3

A clustering and analysis framework for mining spatial data

3.1 Introduction

Tools that visualize and analyze geo-referenced datasets have gained importance in the last decade, as is evident from the increased popularity of products, such as Google Earth, Microsoft Virtual Earth, and ArcGIS. Polygons play an important role in the analysis of geo-referenced data as they provide a natural representation of geographical objects, such as countries, buildings, and pollution hotspots. Polygons can also serve as models for spatial clusters, and can model nested and overlapping clusters. Moreover, polygons have been studied thoroughly in geometry and they are mathematically well understood. Furthermore, powerful software libraries are available to manipulate, analyze, and quantify relationships between polygons.

Spatial extensions of popular database systems, such as ORACLE, PostGIS, and Microsoft SQL Server, support polygon search and polygon manipulation in extended versions of SQL; however, past and current data-mining research has mostly ignored the capabilities that polygon analysis can offer.

In general, polygon analysis is particularly useful to mine relationships among multiple spatial datasets, as it provides a useful tool to analyze discrepancies, progression, change, and emergent events. Our work focus on clustering and analyzing polygons that have been generated from multiple spatial point datasets. In particular, the scope of a spatial cluster is described by a polygon; points inside a polygon belong to the same spatial cluster, while points outside of a polygon do not. Our framework provides computational methods to create such spatial clusters from multiple spatial point datasets. Multiple related spatial datasets contain a lot of overlapping polygons. Traditional distance functions and clustering algorithms for points would not work directly for polygons. New distance functions and spatial clustering algorithm are proposed to cluster spatial polygons. We use the term meta-cluster to refer to a cluster containing sets of polygons. As there are usually a lot of meta-clusters containing multiple polygons, it is desirable to have automated screening procedures to help domain experts to select clusters and meta-clusters that they are interested in based on their domain-driven notion of "interestingness"; therefore, our framework provides post-processing techniques which mine the obtained meta-clusters to extract interesting patterns and summarized knowledge based on a domain expert's notion of interestingness. The architecture of our framework is introduced in Section 3.2.

The rest of the chapter is organized as follows. Section 3.2 introduces the architecture of our framework. Section 3.3 explains our DCONTOUR algorithm that can be used to generate polygons from point datasets. Section 3.4 discusses the distance functions for polygons and the spatial clustering algorithm for overlapping polygons. Section 3.5 presents two post-processing analysis techniques for finding interesting clusters. Section 3.6 evaluates our work with challenging real-world case studies involving ozone pollution events in the Houston-Galveston-Brazoria (HGB) area. Section 3.7 gives our conclusion.

3.2 Framework architecture

In our framework, we first generate spatial clusters represented by polygons from multiple spatial point datasets. Both spatial clustering algorithms which directly derive polygons from point datasets and approaches that initially obtain spatial clusters as sets of objects and wrap a polygon around those objects can be used to obtain such spatial clusters. As the first type of algorithm is not very common, an algorithm called DCONTOUR is introduced for this purpose. In the second step, we introduce two distance functions called Overlay distance and Hybrid distance to access the distance between overlapping polygons. The Shared Nearest Neighbor algorithm (SNN) [24] is extended to cluster polygons by redefining the density concepts for polygons. In the third step, post-processing analysis techniques are provided to extract interesting patterns and to provide summaries from the meta-clusters based on a domain expert's notion of "interestingness". A spatial cluster will be characterized by two

things in our work: a polygon which describes the scope of a spatial cluster and a statistical summary based on all the objects belonging to the same clusters; the statistical summary usually contains mean values and standard deviations of various non-spatial variables for the objects in the same spatial clusters. Two particular post-processing techniques are proposed: First, a greedy algorithm is developed to automatically select a set of interesting polygons from meta-clusters to obtain a final clustering. Second, a screening procedure which uses plug-in reward functions is introduced to automatically identify interesting meta-clusters which have unexpected member distributions with respect to a continuous non-spatial variable.

In summary, our framework is an integration of clustering algorithms, post-processing analysis techniques, and visualization. The architecture of our framework is summarized in Figure 3.1 It consists of three steps:

Step 1: Apply the DCONTOUR algorithm to generate polygons which describe spatial clusters from multiple spatial point datasets.

Step 2: Apply the Poly-SNN clustering algorithm to create meta-clusters from the polygons that were generated in Step 1.

Step 3: Extract interesting patterns and create summaries from the meta-clusters using post-processing analysis techniques.

We use multiple ozone concentration datasets downloaded from the website of Texas Commission on Environmental Quality (TCEQ) [25] as an example to further explain the three steps in our framework. The TCEQ uses a network of 44 ozone-monitoring stations in the HGB area which covers the geographical region within

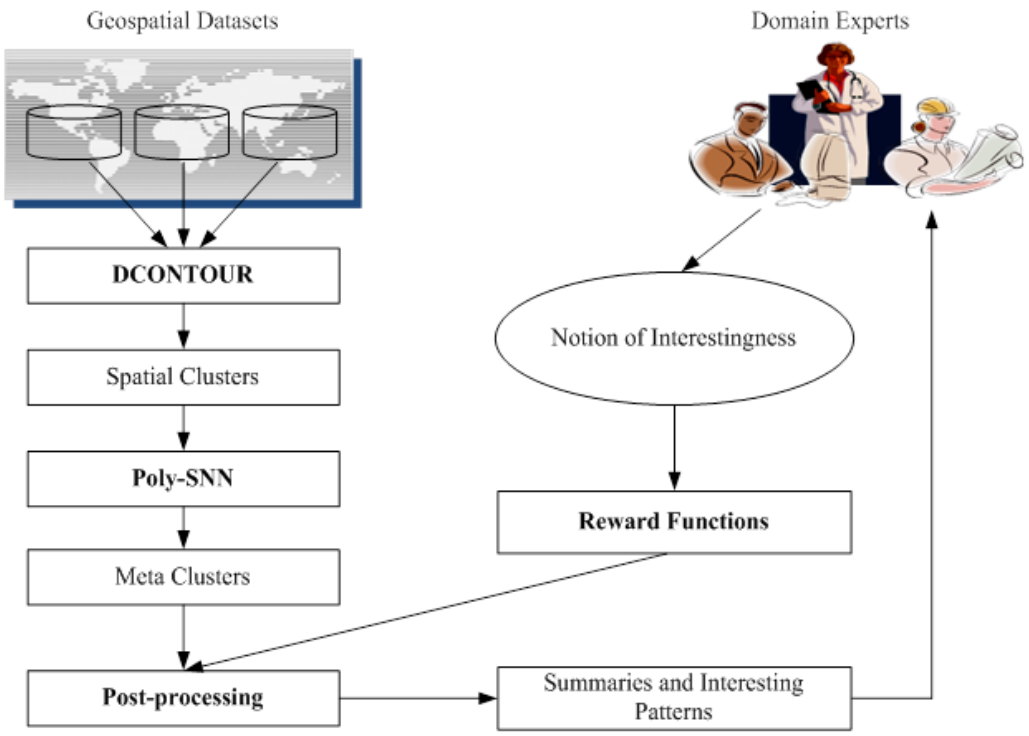

Figure 3.1: The clustering and analysis framework for mining spatial data

[-95.8070, -94.7870] longitude and [29.0108, 30.7440] latitude. It collects hourly ozone concentration data from each monitoring station and publishes the data on its website. In Step 1, we first apply a standard Kriging interpolation method [26] to compute the ozone hourly concentrations on 20×27 grids that cover the HGB area. Next, we feed the interpolation function into the DCONTOUR algorithm with a defined threshold to create sets of polygons. Such polygons describe ozone pollution hotspots at each hour - areas whose hourly ozone concentrations are above the input threshold. In Step 2, we apply the Poly-SNN algorithm with the Hybrid distance function to cluster polygons and create meta-clusters. In Step 3, we propose several plug-in reward functions to capture a domain expert's notion of "interestingness" to

guide the extraction of knowledge from the meta-clusters. In particular, an algorithm to generate a final clustering from the meta-clusters is proposed. Such a final clustering could help domain experts to clearly capture the dominant ozone pollution hotspots and the possible maximum range of the ozone pollution events in the HGB area. Moreover, automated screening procedures to identify unusual meta-clusters are introduced.

3.3 DCONTOUR

DCONTOUR [27] is the first density-based clustering algorithm that uses contour lines to determine cluster boundaries. Objects that are inside a contour polygon belong to the same cluster. DCONTOUR operates on top of supervised density functions.

We assume that an object o in the dataset O has the form $((x, y), z)$ where (x, y) is the location of the object o, and z or $z(o)$ is the value of interestingness of object o. In general, density estimation techniques employ influence functions that measure the influence of a object o with respect to another object v. The overall influence of all objects $o_i \in O$ for $1 \leq i \leq n$ on a point v is measured by the density function $\varphi^o(v)$. The density estimation is called supervised because in addition to the density information based on the locations of all objects, we take the interestingness $z(o)$ into consideration when measuring the density. The density function $\varphi^o(v)$ is defined as follows:

$$\varphi^o(v) = \sum_{i=1}^{n} f_{influence}(v, o_i) = \sum_{i=1}^{n} z(o_i) e^{-\frac{d(v,o_i)^2}{2\sigma^2}} \tag{3.1}$$

The parameter σ determines how quickly the influence of a object o_i on v decreases as the distance between o_i and v increases. The pseudo-code of DCONTOUR is given in Algorithm 1.

Input: *Density function φ^o, density threshold d*
Output: *Density polygons for the density threshold d*
Algorithm:

1. *Subdivide the space into D grid cells.*
2. *Compute densities at grid intersection points by using density function φ^o.*
3. *Compute the contour intersection points b on the grid cell edges where $\varphi^o(b) = d$ using binary search and interpolation.*
4. *Compute contour polygons from the contour intersection points b.*

Algorithm 1: DCONTOUR algorithm pseudo-code

Figure 3.2 gives an illustration on how to construct contour intersection points based on the density threshold d being equal to 4.5. For instance, when the right edge of the lower left cell is considered, because 4.5 is between 4.1 and 5.5, a contour intersection point exists on this edge; by interpolating between 4.1 and 5.5, a point on this edge is sampled and its density is computed as 4.8. Because 4.8 is larger than d, we continue the binary search by sampling a point south of this point. The binary search terminates if the density difference between a sampled point and d is less than a threshold. All the blue points in Figure 3.2 are the contour intersection points b for density threshold d equal to 4.5. Finally, in step 4, we connect contour intersection points b found on cell edges and continue this process on its neighboring cells until

a closed polygon is formed or both ends of the polyline reach the grid boundary. An algorithm proposed by Cottafava and Moli [4] is used to compute contour polygons.

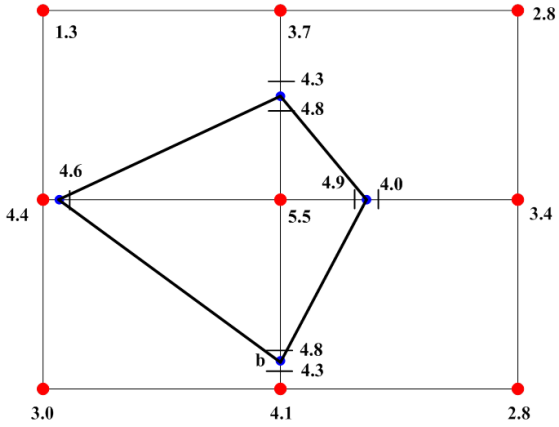

Figure 3.2: Contour construction for density equal to 4.5

3.4 Distance functions and clustering algorithm for polygons

3.4.1 Distance functions for polygons

One unique characteristic of our work is that we have to cope with overlapping polygons. We believe that considering polygon overlap is of critical importance for polygon-based clustering of large spatial datasets; therefore, in addition to the Hausdorff distance, we propose two novel distance functions: the Overlay distance and the Hybrid distance. We define a polygon A as a sequence of points $A = p_1, ..., p_n$, with p_1 being connected to p_n to close the polygon. Moreover, we assume that the boundary of a polygon does not cross itself and a polygon can have holes inside.

Throughout this chapter we use the term polygon to refer to such polygons.

The Hausdorff distance [3, 4] measures the distance between two sets of points A and B by using Formula 3.2.

$$D_{\text{Hausdorff}}(A, B) = \max_{a \in A}\{\min_{b \in B}\{d(a, b)\}\} \tag{3.2}$$

where a and b are points of sets A and B, and $d(a, b)$ is the distance between two points a and b. Based on this defined formula, the distance between two polygons is determined by the maximum distance of a point in A to its nearest point in B.

In order to use the Hausdorff distance for polygons, we have to determine how to associate a set of points with a polygon. One straight-forward solution is to define the set of points as the points that lie on the boundary of a polygon; however, computing the distance between a set of points that consist of unlimited number of points is considerably expensive. An algorithm that solves this problem for trajectories has been proposed [8] and the same technique can be applied to polygons.

The Overlay distance measures the distance between two polygons based on their degree of overlap. The overlay distance between two polygons A and B is defined as:

$$D_{\text{Overlay}}(A, B) = 1 - \frac{area(\text{Intersection}(A, B))}{area(\text{Union}(A, B))} \tag{3.3}$$

where the function $area(X)$ returns the area a polygon X covers. The overlay distance subtracts the ratio of the size of the intersection over size of the union of two polygons from 1. The overlay distance is 1 between two non-overlapping polygons.

The Hybrid distance function uses a linear combination of the Hausdorff distance and the Overlay distance. Because the Overlay distance between two non-overlap

polygons is always 1, regardless of the actual location in space, using the Hausdorff distance can provide more precise approximations of the distance between two non-overlap polygons. The Hybrid distance function is defined as:

$$D_{\text{Hybrid}}(A, B) = w \times D_{\text{Overlay}}(A, B) + (1 - w) \times D_{\text{Hausdorff}}(A, B) \tag{3.4}$$

where w is the weight factor associated with the Overlay distance function ($0 \leq w \leq 1$).

There are several distance functions [1]-[4] proposed in the literature for polygons; however, none of them can cope with overlapping spatial polygons. Overlapping spatial polygons play a very important role in analyzing multiple related spatial datasets in many application domains. Failing to measure the degree of overlap will result in inadequate clustering results.

3.4.2 Spatial clustering algorithm for polygons

The Shared Nearest Neighbors (SNN) clustering algorithm [24] is a density-based algorithm. SNN clusters data as DBSCAN does, except that the number of the nearest neighbors that two points share is used to assess the similarity instead of the number of points being within the radius ϵ of a particular point. In SNN, the similarity between two points p_1 and p_2 is the number of points they share among their k nearest neighbors as follows:

$$similarity(p_1, p_2) = size\_of(NN(p_1) \cap NN(p_2)) \tag{3.5}$$

where $NN(p_i)$ is the set of k nearest neighbors of a point p_i.

The SNN density of a point p is defined as the sum of the similarities between p and its k nearest neighbors as follows:

$$density(p) = \sum_{i=1}^{k} similarity(p, p_i) \tag{3.6}$$

where p_i is the i^{th} nearest neighbor of p.

After assessing the density of each point, SNN finds all core points; all points in the dataset D that have the SNN density of at least $MinPs$ and form the clusters around the core points like DBSCAN.

$$coreP(D) = \{p \in D | density(p) \geq MinPs\} \tag{3.7}$$

Our spatial clustering algorithm for polygons (Poly-SNN) extends the SNN algorithm from points to polygons. The key component of Poly-SNN is the calculation of the distance between two polygons. We use the Hybrid distance function discussed in Section 3.4.1. Next, we identify the k nearest neighbors for each polygon. Poly-SNN calculates the SNN density of each polygon using its k nearest neighbors, and clusters the polygons around the core polygons as described above. The pseudo-code of Poly-SNN is given in Algorithm 2.

There are several advantages of using the SNN algorithm as our reference algorithm. First, it has the ability to find clusters in the presence of outliers. Second, the SNN algorithm is capable of finding clusters of different shapes, sizes, and densities. Third, it works well for high dimensional data. The experimental results and detailed discussions in [24] show that SNN performs better than traditional methods, such as K-means, DBSCAN, and CURE, on a variety of datasets.

Input: *Contour polygons, number of nearest neighbor k*
Output: *Clusters of polygons (Meta-clusters)*
Algorithm:

1. *Compute the similarity matrix of the input contour polygons.*
2. *For each polygon p, find its k nearest neighbors.*
3. *For each polygon p, compute its SNN density.*
4. *Identify the core polygons.*
5. *Form clusters from the core polygons.*
6. *Mark all noise polygons.*

Algorithm 2: Poly-SNN algorithm pseudo-code

3.5 Post-processing analysis techniques

3.5.1 Domain-driven final clustering generation method

In general, domain experts seek clusters based on their domain-driven notion of "interestingness". Usually, a domain expert's interestingness is different from generic characteristics used by traditional clustering algorithms; moreover, for a given dataset there usually are many plausible clusterings whose value has to be determined by domain experts. Finally, even for the same domain expert, multiple clusterings are of value, e.g., clusterings at different levels of granularity. A key idea of this research is to collect a large number of frequently overlapping clusters organized in the form of meta-clusters; final clusterings and other summaries are then created from those meta-clusters based on a domain expert's notion of interestingness.

To reflect what was discussed above, we assume that our final clustering generation algorithms provide several plug-in reward functions that capture a domain

expert's notion of interestingness. The reward function will be maximized during the final clustering generation procedure. Our methodology provides an alternative approach to the traditional ensemble clustering by creating a more structured input for obtaining a final clustering, and reducing algorithm complexity by restricting choices. We propose algorithms that create a final clustering by selecting at most one cluster from each meta-cluster. Moreover, due to the fact that polygons that were generated from multiple spatial datasets usually overlap a lot, we provide an option for domain experts to restrict the cluster overlap in the final clustering. More specifically, we develop algorithms that create the final clustering from the input meta-clusters by solving the following optimization problem:

Inputs:

1. A meta-clustering $M = X_1, \ldots, X_k$ - at most one object will be selected from each meta-cluster $X_i (i = 1, \ldots, k)$.

2. The user provides the individual cluster reward function $Reward_U$ whose values are in $[0, \infty)$.

3. A reward threshold θ_U - clusters with low rewards are not included in the final clustering.

4. A cluster distance threshold θ_d, which expresses to what extent the user would like to tolerate cluster overlap.

5. A cluster distance function $dist$.

Find $Z \subseteq X_1 \cup ... \cup X_k$ that maximizes:

$$q(Z) = \sum_{c \in Z} reward_U(c) \tag{3.8}$$

subject to:

1. $x \in Z \ \forall x' \in Z(x \neq x' \Rightarrow Dist(x, x') > \theta_d)$
2. $x \in Z(Reward_U(x) > \theta_U)$
3. $x \in Z \ \forall x' \in Z((x \in X_i \wedge x' \in X_k \wedge x \neq x') \Rightarrow i \neq k)$

Our goal is to maximize the sum of the rewards of clusters represented by polygons that have been selected from the input meta-clusters. Constraint 1 prevents two clusters which are spatially too close to be included in the final clustering. Constraint 3 makes sure that at most one cluster from each meta-cluster is selected.

Assuming that we have n meta-clusters, and that each meta-cluster contains an average of m clusters (polygons), there are roughly $(m+1)^n$ final clusterings; for each meta-cluster, we can either select one cluster for inclusion or we might decide not to take any cluster due to the violations of Constraints 1 and 2. Constraint 2 is easy to handle by removing clusters below the reward threshold from the meta-clusters prior to running the final clustering generation algorithm.

Many different algorithms can be developed to solve this optimization problem. We developed a greedy algorithm that always selects the cluster with the highest reward from the unprocessed meta-clusters whose inclusion in the final clustering does not violate Constraints 1 and 2. If there is no such cluster left, no more cluster will be added from the remaining meta-clusters to the final clustering.

The greedy algorithm is very fast ($O(m \times n)$) but far from optimal; the backtracking algorithm explores the complete search space ($O(m^n)$) and - if not stopped earlier - finds the optimal solution if n and m are not very large; however, the anytime backtracking approach can be used for large values of m and n. Finally, the evolutionary computing algorithm covers a middle ground, providing acceptable solutions that are found in the medium run-time. The pseudo-code of the greedy algorithm is given in Algorithm 3

Input: *Meta-clusters* $M = \{X_1, \ldots, X_k\}$*, Reward functions* $Reward_U$*, Reward threshold* θ_U*, Cluster distance function dist, Distance threshold* θ_d
Output: *Final Clustering* $F = \{p_1, \ldots, p_n\}$
Algorithm:

1. *Mark all meta-clusters* X_i *unprocessed; initialize final clustering* F *to empty.*
2. *Compute the reward using* $Reward_U$ *for meta-clusters* M *and delete polygons whose rewards are less than the reward threshold* θ_U *from meta-clusters* M*.*
3. *Select the polygon* p *with the highest reward from the unprocessed meta-clusters.*
4. *Compute the distances* $dist_q$ *between* p *and every* $q \in F$*, if all* $dist_q \geq \theta_d$*, put* p *into* F*, mark* $X_i (p \in X_i)$ *processed; otherwise remove* p *from* X_i*; if* X_i *is empty, mark* X_i *as processed.*
5. *Stop if all* X_i *are marked processed, otherwise go back to Step 3.*
6. *Output the final clustering* F*.*

Algorithm 3: The greedy algorithm pseudo-code

As an example, we use the reciprocal of the area of a polygon as the reward function. This reward function can help domain experts to identify potential ozone pollution point sources and to analyze patterns at different levels of granularity when different parameters are selected. First, all input meta-clusters generated by Poly-SNN are marked "unprocessed". The final clustering F is initialized to empty.

The reward values for all polygons (the reciprocal of the areas of all polygons) are computed. The user inputs a reward threshold and a distance threshold, e.g., the reward threshold $Reward_U$ equal to 10, and the distance threshold θ_d equal to 0.5. Next, a polygon p with the highest reward from the unprocessed meta-clusters M is selected; compute the distances between p and all polygons in F, if all distances are greater than the distance threshold 0.5, put the polygon p into F, and flag the meta-cluster X_i that the polygon p belongs to as "processed". Otherwise, remove the polygon p from the meta-cluster X_i. The algorithm repeats until all meta-clusters are flagged as "processed". The output is the final clustering F containing all selected polygons.

3.5.2 Finding interesting meta-clusters with respect to a continuous variable v

Our second post-processing analysis technique allows automatic screening of the obtained meta-clusters for unexpected distributions. The main idea is to provide interestingness functions that automatically identify the meta-clusters whose member distribution with respect to a non-spatial variable v deviates significantly from its distribution in the whole dataset. We introduce such an interestingness function that measures the interestingness of a meta-cluster based on its mean value and standard deviation of a non-spatial variable v.

We assume that a dataset $D = (a_1, \ldots, a_n, v)$ and a meta-clustering $M = \{X_1, \ldots, X_k\}$

is given, where v is a continuous variable which has been normalized using the z-scores. Our goal is to find the contiguous clusters in $A = \{a_1, \ldots, a_n\}$ space which maximize the following interestingness function:

Let $X_i \in 2^A$ be a cluster in the A-space

Let σ be the variance of v with respect in the dataset D

Let $\sigma(X_i)$ be the variance of variable v in a cluster X_i

Let $mv(X_i)$ be the mean value of variable v in a cluster X_i

Let $t_1 \geq 0$ be a mean value reward threshold and $t_2 \geq 1$ be a variance reward threshold

We suggest using the following interestingness function φ to calculate the reward for each cluster:

$$\varphi(X_i) = max(0, |mv(X_i)| - t_1) \times max(0, \sigma - \sigma(X_i) \times t_2) \tag{3.9}$$

In general, only clusters which satisfy $|mv(X)| > t_1$ and $\sigma(X) < \sigma/t_2$ will receive a reward value; e.g., for $t_1 = 0.2$ and $t_2 = 2$, only clusters whose mean-value is below -0.2 or above 0.2 and whose variance is less than or equal to half of σ will receive a reward. In general clusters whose mean values are significantly different from 0 and variances are low will receive high rewards. We rank all clusters based on their rewards and only report those whose rewards are higher than the reward threshold specified by domain experts.

The proposed interestingness function is just an example for identifying unusual

clusters with respect to a continuous non-spatial variable v - other useful interestingness functions can be proposed as well. Similar interestingness functions can also be proposed for categorical variables.

3.6 Experimental evaluation

3.6.1 The ozone datasets

The HGB area is currently classified as a severe nonattainment area for ozone. In order to closely monitor the air quality in this area, an air quality monitoring network has been established to continuously monitor the ground level of ozone concentration in this area, which provide large amounts of dynamic data associated with ozone pollution events in this area, such as ozone concentration, nitrogen oxides (NOx) concentration, and various meteorological data. Time-series measurements for such data constitute large-scale spatial datasets. Data-mining techniques are needed to facilitate the automatic extraction and analysis of interesting patterns from such large-scale spatial data; thus, we consider an application of our spatial clustering algorithm and post-processing analysis techniques to ozone pollution spatial data in the HGB area. In particular, we downloaded the raw data for the timeframe from 1 am on April 1, 2009 to 11 pm on November 30, 2009 from the website of the Texas Commission on Environmental Quality (TCEQ) [25], which include ground level ozone concentration, outdoor temperature, wind direction, wind speed, and solar radiation. The spatial scope covers the geographical region within the

longitude domain of $[-96.1, -94.5]$ and the latitude domain of $[29.0, 30.4]$. The standard Kriging interpolation method [26] and the DCONTOUR [27] algorithm along with the user defined density threshold, i.e., 90 *ppb* (parts per billion), are adopted to generate polygons. Those polygons represent the area in which the hourly ground level ozone concentration is higher than 90 *ppb*. Polygons are represented by contour lines. The shape and area of a polygon reflect the impact location and scope of an ozone pollution event, which are controlled by multiple factors of the emission precursors and the meteorological conditions. We use the dataset consisting of 162 polygons generated by the DCONTOUR algorithm for the density threshold 90 *ppb* in the following case studies in this chapter.

Ozone formation is a complicated chemical reaction. There are several control factors involved:

- Sunlight, measured by solar radiation, is needed to produce ozone.
- High outdoor temperature causes ozone formation reactions to speed up.
- Wind transports ozone pollution from the point sources.
- Time of Day: ozone levels can continue to rise all day long on a clear day, and then decrease after sunset.

Solar radiation is measured in langleys per minute. A langley is a unit of energy per unit area (1 gram-calorie/cm2) commonly employed in radiation measurements. Outdoor temperature is measured in Fahrenheit. Wind speed is measured in miles per hour.

Basically, we generate polygons from the original point datasets to capture ozone hotspots for particular time slots in the HGB area. 255 polygons are created by using DCONTOUR with density threshold 180 (ozone concentration 80 *ppb*). 162 polygons are generated by using DCONTOUR with density threshold 200 (ozone concentration 90 *ppb*). The current Environmental Protection Agency (EPA) ozone standard is based on eight-hour average measurements. In order to meet the standard, the eight-hour average ozone concentration has to be less than 75 *ppb*; therefore, we can consider these two sets of polygons represent areas where the ozone level exceeds the EPA standard.

We evaluate our method in three case studies. The goal of the first case study is to verify that our new distance functions and spatial clustering algorithm for polygons can effectively cluster overlapping polygons. By analyzing additional meteorological variables associated with polygons, such as outdoor temperature, solar radiation, wind speed, and time of day, we can characterize each cluster of polygons and identify interesting patterns associated with these ozone pollution hotspots represented by polygons.

In the second case study, we are interested in generating final clusterings that capture a domain expert's notions of interestingness by plugging in different reward functions. For example, domain experts may be interested in finding typical ozone pollution hotspots that occurred when the outdoor temperatures were extremely high. In order to summarize the final clusterings, the statistical results of three ozone pollution control variables are provided as well.

In the third case study, we try to find interesting clusters with unexpected distributions with respect to a continuous variable. A screening procedure and several interestingness functions are proposed for this task. Meta-clusters are evaluated with respect to different continuous variables, such as solar radiation, wind speed, and outdoor temperature, respectively.

3.6.2 Case study for spatial clustering

An ozone polygon is a hotspot that has an ozone concentration above a certain threshold. In order to generate polygons representing ozone pollution hotspots where ozone concentration is above 90 *ppb* from the original point datasets downloaded from TCEQ's website [25], we divide the HGB area into a 20×27 grid. The density function discussed in Section3.3 is used to compute the ozone concentration at each grid intersection point. Next, we compute the contour intersection points on the grid cell edges where the density is equal to 90 *ppb* using binary search and interpolation. Finally, we compute the contour polygons by connecting the contour intersection points.

In this case study, we select the dataset with 162 polygons generated by the DCONTOUR algorithm with the density threshold equal to 200. These polygons represent areas with a one hour ozone concentration higher than 90 *ppb*. We then apply the Poly-SNN clustering algorithm to find clusters of ozone hotspots represented by polygons called meta-clusters. Figure 3.3 displays the result of 30 meta-clusters generated by Poly-SNN with the number of nearest neighbor k equal to 5 and the

Hybrid distance function. Out of 162 polygons, 30 percent are considered outliers by Poly-SNN. Polygons marked by the same color belong to the same meta-clusters.

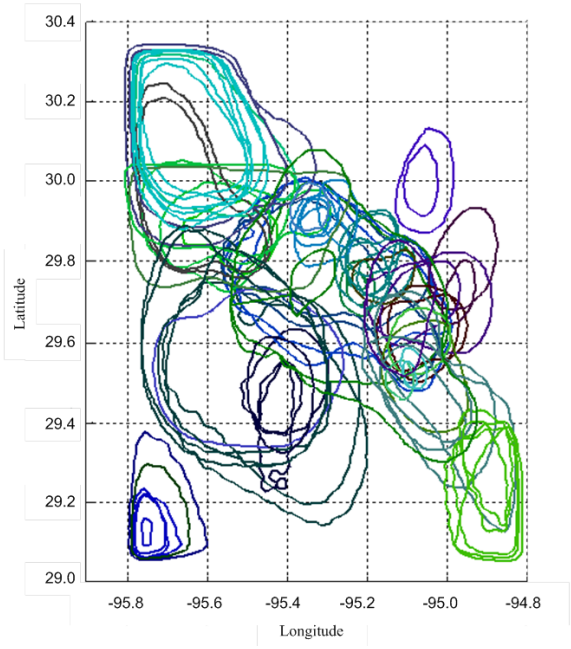

Figure 3.3: All meta-clusters generated by Poly-SNN

In general, by analyzing the meteorological characteristics of polygons, domain experts may find some interesting phenomena that could lead to further scientific investigation; therefore, we also compute the statistics of four meteorological variables involved in ozone pollution events. Table 3.1 lists the statistical results of four meteorological variables associated with the meta-clusters displayed in Figure 3.3.

Table 3.1: The statistical results for all meta-clusters

	Mean	Deviation	Max	Min
Temperature (°F)	90.6	5.3	102.8	78.6
Solar Radiation (Langleys per minute)	0.8	0.4	1.4	0.03
Wind Speed (Miles per hour)	6.1	1.9	15.7	0.3
Time of Day	2 : 30 pm	1.8	8 pm	10 am

As expected, meta-clusters shown in Figure 3.3 are characterized by high outdoor temperature (average of 90.6 °F and standard deviation of 5.3) and strong solar radiation (average of 0.8 langleys per minute and standard deviation of 0.4), which usually happens between 1 pm and 4 pm. Since the standard deviation of the wind speed (1.9) compared with the average wind speed (6.1 miles per hour) is nontrivial, the variance of the sizes of all polygons in Figure 3.3 is significant.

It is hard to visualize all meta-clusters in a single picture when clusters overlap a lot. Figures 3.4 and 3.5 display eight out of 30 meta-clusters shown in Figure 3.3. As expected, the Hybrid distance function that employs both the Overlay distance and the Hausdorff distance creates clusters of polygons that are similar in terms of shape, size and location. Particularly, since we give more weights to the overlay distance function, the meta-clusters in Figures 3.4 and 3.5 overlap significantly. This case study proves that our Poly-SNN algorithm in conjunction with the Hybrid distance function can effectively find clusters of overlapping polygons with similar sizes, shapes, and locations.

Tables 3.2, 3.4, 3.3, and 3.5 list the mean and standard deviation of outdoor temperature, solar radiation, wind speed, and time of day associated with eight meta-clusters shown in Figures 3.4 and 3.5. The solar radiation information related to clusters 2 and 4 are not available from the TCEQ's website. Certainly, ozone formation is more complicated than only considering those four control factors; however, our polygon-based method does have the capability of handling more variables.

Based on Tables 3.2 and 3.3, we can see that polygons in the meta-clusters 11 and 12 are characterized by high outdoor temperatures (98.8 °F and 99.1 °F) compared

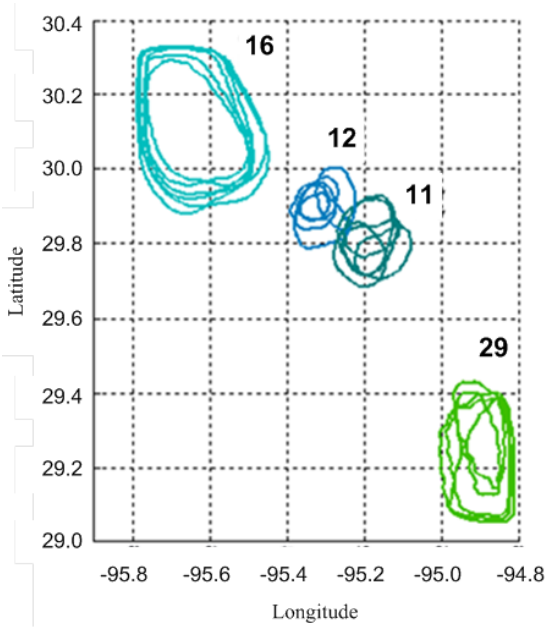

Figure 3.4: Visualization of four meta-clusters (ID: 11, 12, 16, and 29)

Table 3.2: The mean values of the meta-clusters 11,12,16, and 29

Meta-cluster ID	11	12	16	29
Temperature (°F)	98.8	99.1	91.0	85.5
Solar Radiation (Langleys per minute)	0.9	0.9	0.7	0.7
Wind Speed (Miles per hour)	5.2	4.9	5.9	8.3
Time of Day	2 pm	2 pm	3 pm	12 pm

with the entire dataset (90.6 °F) and strong solar radiations (0.9 langleys per minute and 0.9 langleys per minute) compared with the entire dataset (0.8 langleys per minute). The wind speeds of the meta-clusters 11 and 12 (5.2 miles per hour and 4.9 miles per hour) are low compared with the mean value of entire dataset (6.1 miles per hour) so that the average size of the polygons in the meta-clusters 11 and 12 are relatively small compared with other polygons shown in Figure 3.3. Also, meta-clusters 11 and 12 are captured around 2 pm. The statistical results associated with the meta-cluster 16 are very close to the mean values of the entire dataset listed in

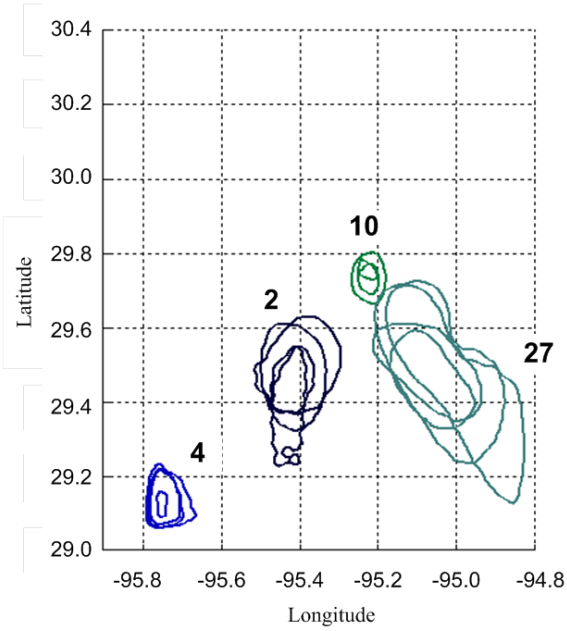

Figure 3.5: Visualization of four meta-clusters (ID: 2, 4, 10, and 27)

Table 3.3: The variances of the meta-clusters 11,12,16, and 29

Meta-cluster ID	11	12	16	29
Temperature (°F)	1.1	2.9	4.3	1.0
Solar Radiation (Langleys per minute)	0.3	0.3	0.3	0.5
Wind Speed (Miles per hour)	0.5	1.0	0.9	2.6
Time of Day	0.9	1.6	1.6	1.9

Table 3.1.

Based on Tables 3.4 and 3.5, the meta-cluster 10 has lower outdoor temperature (86.0 °F), lower solar radiation (0.7 langleys per minute), and lower wind speed (4.8 miles per hour) compared with the mean values of the entire dataset listed in Table 3.1. The average time of day for the meta-cluster 4 is about 4 pm. Those four lower meteorological values contribute to the smaller sizes of polygons inside the meta-cluster 4 shown in Figure 3.5.

Table 3.4: The mean values of the meta-clusters 2,4,10,and 27

	Mean	Deviation	Max	Min
Temperature (°F)	83.4	88.5	86.0	92.3
Solar Radiation (Langleys per minute)	N/A	N/A	0.7	0.6
Wind Speed (Miles per hour)	6.8	6.2	4.8	6.5
Time of Day	2 pm	1 pm	4 pm	3 pm

Table 3.5: The variances of the meta-clusters 2,4,10, and 27

Meta-cluster ID	11	12	16	29
Temperature (°F)	3.6	1.6	2.1	2.9
Solar Radiation (Langleys per minute)	N/A	N/A	0.0	0.3
Wind Speed (Miles per hour)	1.0	0.5	0.8	0.5
Time of Day	1.7	0.9	0.8	0.8

3.6.3 Case study for domain-driven final clustering generation

In this case study, the greedy algorithm is implemented to generate the domain-driven final clustering from the meta-clusters shown in Figure 3.3. We use several reward functions to capture the domain experts' notions of interestingness. The final clustering with statistical results of the corresponding meteorological variables can be used to summarize what characteristics the ozone hotspots in the same meta-clusters share.

The range of an ozone pollution event represented by the area of a polygon is selected as the first reward function, which will help domain experts recognize the possible maximal range of ozone pollution events in the HGB area. By selecting different reward threshold values and distance threshold values, several final clusterings could be generated. Figure 3.6 shows one such final clustering using reward threshold 0.04 and the Hybrid distance threshold 0.5. There are five polygons. Table 3.6 shows the statistical results of the corresponding meteorological variables. Since the

standard deviations of the meteorological variables are relatively small, we will not discuss the standard deviations in this case study. Polygons 21, 80, and 150 cover larger areas with high outdoor temperature, high wind speed, and strong solar radiation compared with polygons 13 and 125. Polygon 150 is interesting because it has a hole inside. Our method can handle polygons with holes. Further analysis could be done to help understand the formation of holes inside polygons.

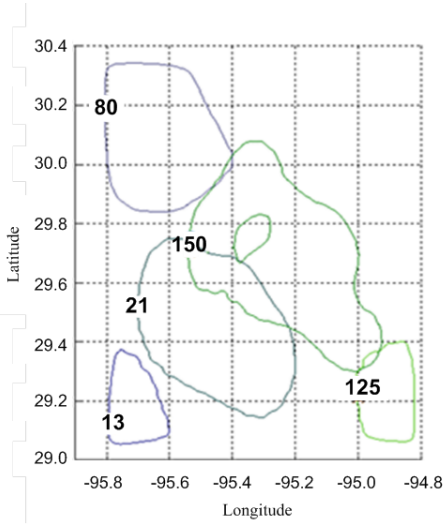

Figure 3.6: Final clustering for the area of polygon reward threshold 0.04 and the Hybrid distance threshold 0.5

Table 3.6: The mean of the meteorological variables for the final clustering of the area reward function

Polygon ID	13	21	80	125	150
Temperature (°F)	79.0	86.4	89.1	84.1	88.9
Solar Radiation (Langleys per minute)	N/A	1.3	1.2	0.1	1.1
Wind Speed (Miles per hour)	4.5	6.1	6.2	4.9	5.4
Time of Day	6 pm	1 pm	2 pm	2 pm	12 pm

The reciprocal of the area of a polygon is selected as the second reward function for smaller granularity, which may be useful to identify the ozone pollution point sources and enable the domain experts to analyze patterns at different levels of

granularity. By decreasing either the reward threshold or the distance threshold, we are able to get several final clusterings. Figure 3.7 shows the final clustering with the reward threshold equal to 10 and the distance threshold equal to 0.45. There are 14 polygons. Table 3.7 lists the mean values of four corresponding meteorological variables. Some of the values are not available in the original datasets downloaded from the TCEQ's website. All of those 14 polygons with relatively smaller sizes occur either before 1 pm or after 4 pm. According to Table 3.1, the average of the time of day for the entire dataset is 2:30 pm with a standard deviation of 1.8. The time slot from 1 pm to 4 pm is definitely a major time period for ozone formation which could change the range and the concentration density of an ozone pollution event significantly. More analysis should be done especially for the time slot between 1 pm and 4 pm.

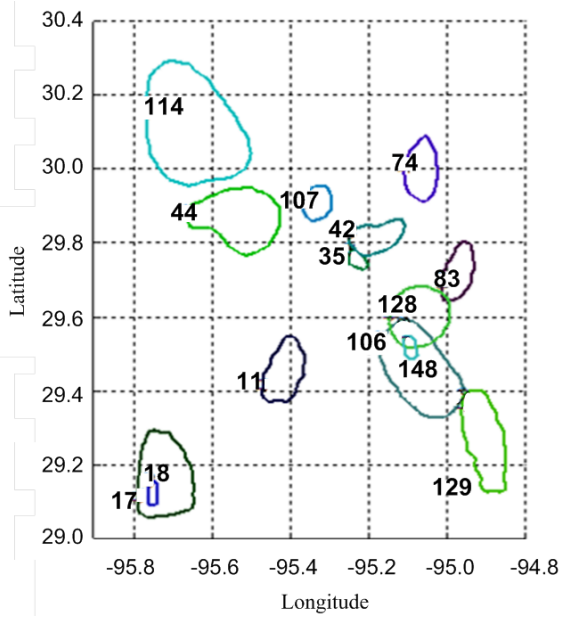

Figure 3.7: Final clustering for the reciprocal of the area reward threshold 10 and the Hybrid distance threshold 0.45

Table 3.7: The mean of the meteorological variables for the final clustering of the reciprocal of the area reward function

Polygon ID	Temperature (°F)	Solar Radiation (Langleys per minute)	Wind Speed (Miles per hour)	Time of Day
11	81.4	N/A	6.3	4 pm
17	88.2	N/A	6.0	3 pm
18	N/A	N/A	N/A	4 pm
35	86.3	N/A	6.2	5 pm
42	N/A	N/A	N/A	1 pm
44	N/A	N/A	N/A	3 pm
74	N/A	N/A	N/A	4 pm
83	N/A	N/A	5.9	10 am
106	93.5	0.12	5.9	4 pm
107	94.4	1.2	4.6	11 am
114	94.6	0.6	5.8	4 pm
128	86.4	0.1	5.4	5 pm
129	86.2	1.1	8.8	10 am
148	N/A	N/A	N/A	N/A

Outdoor temperature, wind speed, and solar radiation also play very important roles in ozone formation. We use average outdoor temperature as the third reward function. Figure 3.8 shows one final clustering with the average outdoor temperature threshold equal to 90 °F and the Hybrid distance threshold equal to 0.55. The mean values of the corresponding meteorological variables are summarized in Table 3.8. Obviously, all the polygons with high temperatures occur during 2 pm to 4 pm. The lower the wind speed is, the smaller the area of a polygon is. For example, polygon 67 has the lowest wind speed of 4.1 miles per hour compared with all other polygons in Figure 3.8, along with a relative high outdoor temperature and a strong solar radiation; the area of polygon 67 is still smaller than the other polygons shown in Figure 3.8.

The solar radiation associated with each polygon is selected as next reward function. Figure 3.9 shows the final clustering for the solar radiation threshold equal to

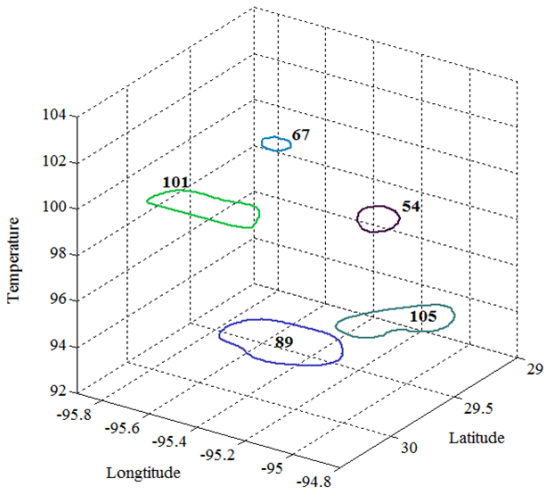

Figure 3.8: Final clustering for the average temperature reward threshold 90 and the Hybrid distance threshold 0.55

Table 3.8: The mean of the meteorological variables for the final clustering of the average temperature reward function

Polygon ID	54	67	89	101	105
Temperature (°F)	100.3	102.8	92.4	99.4	94.5
Solar Radiation (Langleys per minute)	N/A	1.0	0.9	0.7	0.7
Wind Speed (Miles per hour)	6.0	4.1	8.5	8.2	6.0
Time of Day	2 pm	3 pm	3 pm	4 pm	3 pm

0.9 and the Hybrid distance threshold equal to 0.55. Table 3.9 lists the corresponding mean values of four meteorological variables. Based on Table 3.9, strong solar radiation happens between 11 am and 1 pm. During that time period, the outdoor temperature is not relative high compared to the entire dataset (90.6). Polygon 107 is the smallest due to the smallest wind speed (4.6) even though it has the highest outdoor temperature (94.4) and strong solar radiation (1.2). Polygon 21 has relative strong solar radiation (1.3), high wind speed (6.1), and relative low outdoor temperature (86.4) compared with the other four polygons shown in Figure 3.9; however, the area of polygon 21 is the largest one.

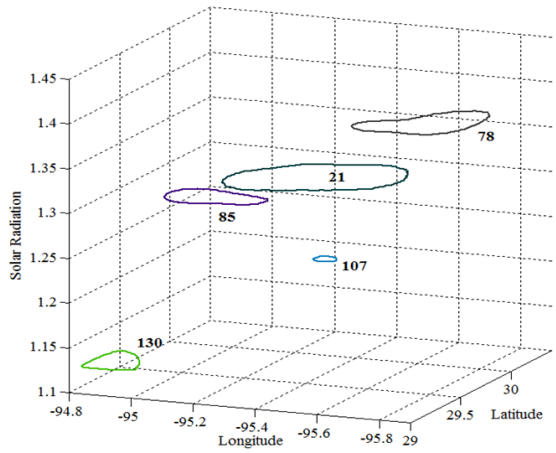

Figure 3.9: Final clustering for solar radiation threshold 0.9 and the Hybrid distance threshold 0.55

Table 3.9: The mean of the meteorological variables for the final clustering of the solar radiation reward function

Polygon ID	21	78	85	107	130
Temperature (°F)	86.4	86.1	92.4	94.4	86.9
Solar Radiation (Langleys per minute)	1.3	1.4	1.3	1.2	1.1
Wind Speed (Miles per hour)	6.1	5.5	4.7	4.6	12.3
Time of Day	1 pm	11 am	11 am	11 am	12 pm

3.6.4 Case study for identifying interesting meta-clusters

For this case study, we use the dataset with 255 polygons generated by DCONTOUR with density threshold 180. 21 meta-clusters are created by using Poly-SNN with the Hybrid distance function and the number of nearest neighbor k equal to 4. 20% of those polygons are considered outliers by Poly-SNN. We evaluate these meta-clusters with respect to several continuous meteorological variables, such as solar radiation, wind speed, and outdoor temperature, respectively. The meta-clusters with high rewards based on the interestingness function φ are identified.

The statistical summaries for three meteorological variables for the entire dataset

are listed in Table 3.10. The average temperature is 89 °F. The average solar radiation is 0.8 langleys per minute. The average wind speed is 5.9 miles per hour. Table 3.11 shows the statistical results after the Z-score normalization. All mean values become 0; all variances become 1.

Table 3.10: The statistical results for the dataset of 255 polygons

	Mean	Variance	Max	Min
Temperature (°F)	89.0	35.5	102.8	71.6
Solar Radiation (Langleys per minute)	0.8	0.1	1.4	0
Wind Speed (Miles per hour)	5.9	2.8	12.3	2.5

Table 3.11: The statistical results of the dataset of 255 polygons after the Z-score normalization

	Mean	Variance	Max	Min
Temperature (°F)	0	1	2.3	−2.9
Solar Radiation (Langleys per minute)	0	1	1.7	−2.1
Wind Speed (Miles per hour)	0	1	3.8	−2.1

In this case study, the mean value threshold is set to 0.2, the variance threshold is set to 2, and the interestingness reward threshold is set to 0.4. We are interested in finding meta-clusters whose mean values are below -0.2 or above 0.2, whose variances are less than or equal to half of the variance of the entire dataset, and whose interestingness rewards are above 0.4.

We first select outdoor temperature. Three meta-clusters (3, 15, and 16) depicted in Figure 3.10 were selected by our post-processing analysis procedure. Table 3.12 lists the normalized outdoor temperatures associated with the meta-clusters shown in Figure 3.10.

Table 3.13 lists the detailed information of each polygon in the final meta-clusters. For example, meta-cluster 15 has five polygons; two out of five polygons were monitored at 1 pm and 2 pm on May 4, 2009, the other three were monitored at 10

Figure 3.10: Interesting meta-clusters with respect to outdoor temperature

Table 3.12: The statistical results for the meta-clusters using outdoor temperature

Meta-cluster ID	Mean	Variance	Number of Polygons
3	−2.2	0.01	2
15	−0.7	0.09	5
16	1.6	0.11	3

am, 11 am, and 12 pm, respectively on June 7, 2009. Further investigation of the meta-cluster 15 will help domain experts better understand how the ozone pollution events change over time.

In general, the highest level of ozone concentration appears a few hours after the maximum solar radiation. We pick solar radiation as our second continuous variable. Figure 3.11 shows three selected meta-clusters with respect to solar radiation. Table 3.14 lists the statistical results of the normalized solar radiation associated with each meta-cluster shown in Figure 3.11. Meta-cluster 5 was picked due to the

Table 3.13: The meteorological information of the selected meta-clusters

Meta-cluster ID	Polygon ID	Temperature (°F)	Solar Radiation (Langleys per minute)	Wind Speed (Miles per hour)	Date	Time of Day
2	3	71.6	N/A	4.0	11/13/2009	2 pm
2	4	71.9	N/A	2.5	11/13/2009	3 pm
2	115	84.7	N/A	4.5	6/6/2009	2 pm
2	117	85.8	N/A	4.9	6/6/2009	3 pm
2	118	86.3	N/A	5.2	6/6/2009	4 pm
3	5	75.1	1.0	4.8	11/7/2009	12 pm
3	6	76.0	0.9	6.6	11/7/2009	1 pm
5	9	76.8	0.4	6.8	11/7/2009	3 pm
5	70	86.1	0.7	8.7	5/4/2009	4 pm
5	127	88.2	0.4	6.3	6/7/2009	5 pm
5	166	93.2	0.3	7.9	8/15/2009	5 pm
5	245	85.2	0.3	7.5	9/4/2009	4 pm
12	38	88.2	N/A	5.7	5/29/2009	2 pm
15	67	85.8	1.1	6.1	5/4/2009	1 pm
15	68	86.6	1.1	6.5	5/4/2009	2 pm
15	120	82.2	1.0	4.6	6/7/2009	10 am
15	121	83.9	1.2	5.1	6/7/2009	11 am
15	122	85.8	1.4	5.6	6/7/2009	12 pm
16	79	96.7	1.3	5.9	6/24/2009	1 pm
16	102	98.3	1.1	5.4	6/24/2009	1 pm
16	112	100.5	0.9	6.2	6/3/2009	3 pm
19	105	98.8	1.0	2.7	6/27/2009	3 pm
19	113	102.8	0.5	4.2	6/3/2009	4 pm
19	184	91.1	0.8	4.6	8/28/2009	3 pm
19	185	91.6	0.5	6.7	8/28/2009	4 pm
19	186	93.2	0.2	3.8	8/28/2009	5 pm
21	172	88.3	1.2	10.0	8/28/2009	1 pm
21	197	85.6	1.1	8.9	8/31/2009	10 am
21	198	86.9	1.1	12.3	8/13/2009	11 am

very low value of solar radiation. It contains five polygons monitored between 3 pm and 5 pm on five different dates ($5/4/2009, 5/29/2009, 6/7/2009, 8/15/2009$, and $9/4/2009$). Meta-cluster 15, however, is picked up again in this case study due to its high values of the solar radiation.

The higher levels of ozone concentrations are associated with greater magnitudes of wind velocity. Figure 3.12 shows two final meta-clusters (2, 5) when wind speed

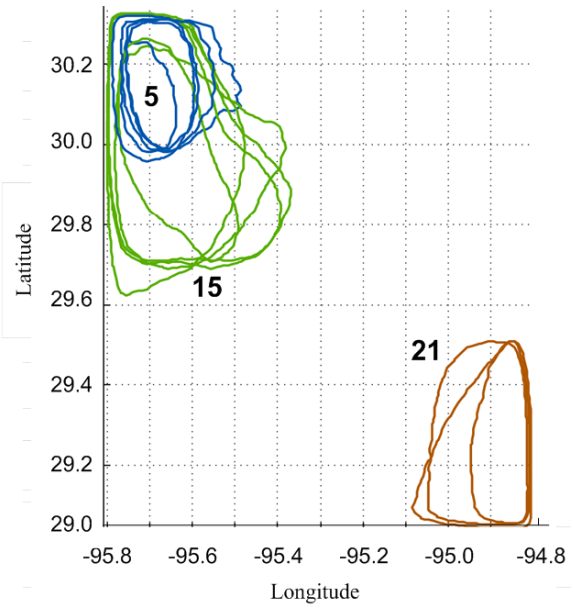

Figure 3.11: The interesting meta-clusters with respect to solar radiation

Table 3.14: The statistical results for the meta-clusters using solar radiation

Meta-cluster ID	Mean	Variance	Number of Polygons
5	−0.9	0.20	5
15	1.1	0.13	5
21	1.0	0.04	3

is selected as the continuous variable in calculating the interestingness reward. Table 3.15 lists the statistical results of the normalized wind speed of each meta-cluster shown in Figure 3.12. There are five polygons in meta-cluster 2; two out of five polygons were monitored at 2 pm and 3 pm on November 13, 2009; the other three were monitored at 2 pm, 3 pm, and 4 pm on June 6, 2009. Figure 3.12 clearly shows the progressions of the ozone pollution events on those two days. Meta-cluster 5 is selected again due to the high values of the wind speed.

Table 3.13 lists the summarized information of the meteorological variables for

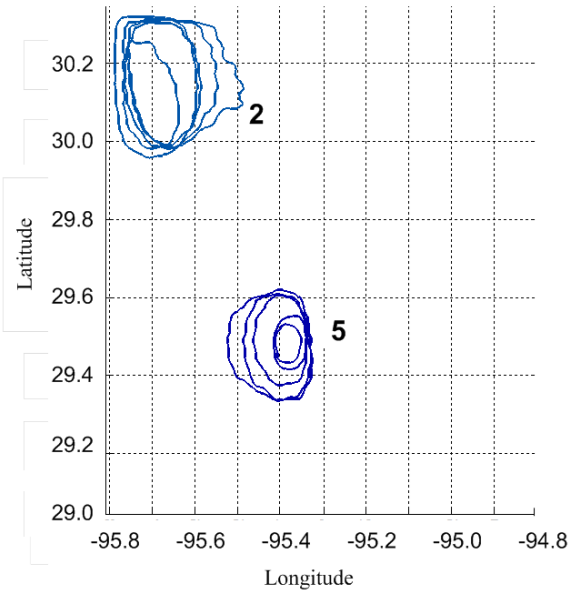

Figure 3.12: Interesting meta-clusters with respect to wind speed

Table 3.15: The statistical results for the meta-clusters using wind speed

Meta-cluster ID	Mean	Variance	Number of Polygons
2	−1.0	0.41	5
15	0.9	0.32	5

all polygons in the final meta-clusters which were red flagged by our post-clustering analysis procedure in this case study. Some meteorological data are not available in the original dataset from TCEQ noted as N/A in Table 3.13. Both meta-cluster 5 and meta-cluster 15 are reported twice in this case study with respect to different meteorology variables. Meta-cluster 5 has a very low mean value of solar radiation and a relative high wind speed. Meta-cluster 15 has a very high mean value of solar radiation and a relative low temperature. They are located in the same area. Under different meteorological conditions, the sizes of the ozone hotspots in meta-clusters 5 and 15 are different. Further analysis of the polygons in meta-clusters 5 and 15 may

help domain experts better understand how ozone pollution hotspots change under different weather patterns over time.

3.7 Conclusion

Polygons are very useful to mine spatial datasets as they provide a natural representation for particular types of spatial objects and provide a useful tool to analyze discrepancies, progression, changes, and emergent events. In this chapter, we proposed a novel polygon-based clustering and analysis framework for mining spatial data. We introduced a density-based contouring algorithm called DCONTOUR to generate polygons from multiple spatial point datasets. Two distance functions were proposed to assess the distance between a pair of overlapping polygons. A density-based polygonal clustering algorithm called Poly-SNN was developed to cluster polygons. Two post-processing analysis techniques were implemented, which employ different plug-in reward functions capturing domain experts' notion of interestingness to extract interesting patterns and summaries from the meta-clusters.

Experiments on real-world spatial data involving ozone pollution events in the HGB area show that our method is effective and can reveal interesting relationships between different ozone hotspots represented by polygons. Our framework can also identify interesting hidden relations between ozone hotspots and several meteorological variables, such as outdoor temperatures, solar radiation, and wind speed. Moreover, our framework has the capability for supporting various applications, such as water pollution and urban evolution.

In general, our work has the capability to cluster overlapping polygons, and polygons with holes inside. In today's society, we are faced with analyzing an ever growing and changing amount of data. It should be highlighted that our framework tries to turn the information overload to our advantage by providing automated screening procedures. It allows for high level views of the data to facilitate data analysis. One key idea of our work is to use different plug-in reward functions to guide the knowledge extraction process, focusing on the extraction of the interesting patterns and summaries with respect to a domain expert's notion of interestingness. To the best of our knowledge, this is the first research that proposes a comprehensive method that relies on polygon clustering and post-processing analysis techniques to mine multiple related spatial datasets.

Chapter 4

New spatio-temporal clustering algorithms and post-processing analysis techniques

4.1 Introduction

Due to the advances in remote sensors and sensor networks, different types of dynamic and geographically distributed spatio-temporal data become increasingly available. Extracting spatial and temporal patterns from such data are very important for many applications, such as geographic information systems, weather forecasting, medical imaging, and environment protection. Traditional clustering techniques are inefficient in clustering spatio-temporal data because they do not incorporate the idiosyncrasies of the spatial and temporal domains; therefore, new techniques are

needed to address these challenges and to provide effective solutions to analyze large spatio-temporal data. Spatio-temporal clustering is particularly useful in analyzing large amounts of data since it allows domain experts to consider groups of objects rather than individual objects and to focus on a higher level representation of the data. Moreover, spatio-temporal clustering can help reveal interesting distribution patterns and serve as the foundation for other data mining and analysis techniques.

Several types of spatio-temporal data are available in real applications, i.e., points, trajectories, and polygons. Different spatio-temporal clustering algorithms have been developed in the literature; however, most of them focus on clustering points and trajectories. Spatio-temporal clustering techniques for polygons are still rarely reported. Moreover, polygons serve an important role in the analysis of geo-referenced data as they provide a nature representation for certain types of objects, such as cities blocks, city neighborhoods, and pollution hotspots. Polygons can serve as models for clusters as well, and can be used by a broad set of applications, such as air pollution prevention, health-care study, and urban planning.

Geographic dynamics refer to the changes that occur across both the spatial and temporal dimensions. Different change analysis techniques for spatio-temporal data have been developed; however, most of them focus on points and trajectories. The current state-of-art is still lacking techniques for analyzing the changes that may occur within the polygon-based spatio-temporal clusters across their spatial and temporal dimensions simultaneously, as well as formalizing their properties. New change detection algorithms need be developed to automate the identification, representation, and computation of geographic dynamics for polygons.

In this chapter we propose two novel spatio-temporal clustering algorithms, i.e., Spatial-Temporal Shared Nearest Neighbor (ST-SNN), and Spatial-Temporal Separated Shared Nearest Neighbor (ST-SEP-SNN), by extending the Shared Nearest Neighbor clustering algorithm (SNN) [24]. Advantages of the SNN algorithm include its capability to find clusters of different shapes, sizes, and densities in high dimensional data and its tolerance to noise. In addition, SNN does not require the number of clusters to be determined in advance. We redefine the spatio-temporal similarity between a pair of polygons and the density concepts of the SNN algorithm. Furthermore, both ST-SNN and ST-SEP-SNN can be easily modified to cluster spatio-temporal point datasets.

We demonstrate how ST-SNN and ST-SEP-SNN can be used to extract knowledge from spatio-temporal data involving ozone pollution events in the HGB area and help answer different analytic questions from domain experts. Ozone has been the main air-quality concern in the HGB area for years. Regional meteorological conditions combined with the variety of emissions from industry and transportation make the city a prime media for ground level ozone formation [25]. The rapid growth of the city causes regional emissions to continue increasing. Our approaches can help analyst find interesting spatio-temporal patterns from ozone pollution events and make preliminary predictions for the future. For example, our algorithms can find hourly patterns of high ozone concentrations that occurred in similar areas. For people who are physically active outdoors or have respiratory problems, this knowledge can help them to plan.

Post-processing analysis techniques are important tasks in data mining as well,

which can help domain experts to explore the data and the clustering results from a variety of viewpoints. Our post-processing analysis technique allows automatic screening of the clusters of polygons for interesting ones, whose members have attribute values that deviate significantly from those of the entire population. For example, we can find clusters of polygons with attribute values that are much smaller or larger than others. Such anomalous clusters are exceptional in some sense, and are often of unusual importance.

The rest of the chapter is organized as follows. Section 4.2 introduces two spatio-temporal clustering algorithms called ST-SNN and ST-SEP-SNN in detail. Functions to assess the similarity between a pair of spatio-temporal polygons are formulated as well. Section 4.3 presents our change pattern discovery algorithm and post-processing analysis technique for spatio-temporal data mining. Section 4.4 evaluates our spatio-temporal clustering algorithms with the case studies on ozone pollution events in the HGB area. Section 4.5 presents the experimental results of applying our change-patter-discovery algorithm and post-processing analysis technique for the same spatio-temporal datasets. Section 4.6 gives our conclusion.

4.2 New spatio-temporal clustering algorithms

We propose two spatio-temporal clustering algorithms, ST-SNN and ST-SEP-SNN by extending the SSN algorithm [24] from points to polygons. We redefine the spatio-temporal similarity between two polygons taking into account both the spatial and temporal similarities. Each spatio-temporal polygon p is associated with a time t

when it occurs, and a set of non-spatial attributes. Henceforth, all polygons discussed in this chapter are spatio-temporal polygons unless specified otherwise. A spatio-temporal cluster of polygons is a group of polygons that lie in close proximity in both space and time.

In ST-SEP-SNN, any function that can compute the distance between a pair of polygons, such as the Hausdoff distance [4], the Fréchet Distance [2], the PDF [1], the Overlay distance [28], the Hybrid distance [29], can be used. Any function that can compute the temporal distance between a pair polygons can be adopted as well. The k-nearest spatial neighbor list and k-nearest temporal neighbor list for each polygon p, denoted by k-SPN-List(p) and k-TN-List(p), are generated by keeping its k-nearest spatial neighbors and k-nearest temporal neighbors only. Then the nearest spatio-temporal neighbor list of a polygon p, denoted by $NN(p)$, is calculated as the intersection of the k-nearest spatial neighbor list and the k-nearest temporal neighbor list of p:

$$NN(p) = \text{k-SPN-List}(p) \cap \text{k-TN-List}(p) \tag{4.1}$$

The similarity between two polygons p and q, denoted by $similarity(p, q)$, is the number of the nearest spatio-temporal neighbors that they share:

$$similarity(p, q) = size\_of(NN(p) \cap NN(q)) \tag{4.2}$$

where $NN(p)$ is the set of k nearest neighbors of the polygon p.

The SNN density of a polygon p is defined as the sum of the similarities between

the polygon p and its k nearest neighbors as follows:

$$density(p) = \sum_{i=1}^{k} similarity(p, p_i) \tag{4.3}$$

where p_i is the i^{th} nearest neighbor of a polygon p.

The core polygons are identified by using a user specified parameter $MinPs$, namely all polygons in the dataset D that have the SNN density of at least $MinPs$:

$$CoreP(D) = \{p \in D | density(p) \geq MinPs\} \tag{4.4}$$

Clusters are then formed by computing the transitive closure of the polygons that can be reached from an unprocessed core polygon using their respective nearest neighbor lists; this process continues until all core polygons have been assigned to a cluster. The remaining polygons that are not within a radius of Eps of any core polygon are classified as outliers; they are not included in any clusters.

In contrast with ST-SEP-SNN, ST-SNN uses a weighted sum of the spatial distance and the temporal distance between two polygons p and q to calculate the spatio-temporal distance between polygons p and q, denoted by $dist_{st}(p, q)$:

$$dist_{st}(p, q) = w \times dist_s(p, q) + (1 - w) \times dist_t(p, q) \tag{4.5}$$

where w is the weight factor associated with the spatial distance ($0 \leq w \leq 1$); $dist_s$ is any functions that can compute the spatial distance between two polygons p and q; $dist_t$ is any functions that can compute the temporal distance between two polygons p and q. Then we rank the obtained spatio-temporal distance matrix to get the k-nearest spatio-temporal neighbor list for each polygon. In this case, the sizes of

the nearest neighbor lists for all polygons in the dataset are the same, i.e., k. The SNN-based concepts for spatio-temporal polygons are identical in both ST-SNN and ST-SEP-SNN. The pseudo-code of ST-SNN and ST-SEP-SNN is given in Algorithm 4.

Input: *polygon dataset* $D = \{p_1, p_2, \ldots, p_n\}$*, size of nearest neighbor list* k*, the core polygon threshold* $MinPs$
Output: *set of clusters* C_i *of polygons*
Algorithm:

1. *for every polygon* p *in* D *,compute k-nearest neighbor list:* $NN(p)$ *,and mark* p *unprocessed.*
2. *for every pair of polygons* p *and* q *in* D*, compute* $similarity(p, q)$*.*
3. *for every polygon* p *in* D*, compute* $density(p)$*, identify core polygons whose density is greater than* $MinPs$*.*
4. *for every core polygon* p *in* D*, if* p *is marked unprocessed, form a cluster* C_i *of polygons that can be reached from* p *following the entries of the respective NN-lists of core points. mark all polygons in* C_i *as processed.*
5. *return set of generated clusters* C_i *of polygons*

Algorithm 4: Pseudo-code of ST-SNN and ST-SEP-SNN

In general, ST-SEP-SNN uses separate k-nearest spatial neighbor list and k-nearest temporal neighbor list, and does not try to integrate the spatial distance and the temporal distance into a single spatio-temporal distance. The nearest neighbor lists of all polygons have different cardinalities m ($m \leq k$). This property distinguishes ST-SEP-SNN from ST-SNN.

Both ST-SNN and ST-SEP-SNN require several user-defined parameters that have significant impact on clustering results. These user-defined parameters need to be changed and adapted according to the dataset being clustered:

- k: the size of the nearest neighbor list. It is the most important parameter as it determines the granularity of the clusters. In general, if k is too small, both ST-SNN and ST-SEP-SNN will tend to find many small clusters and a lot of outliers. On the other hand, if k is too large, both ST-SNN and ST-SEP-SNN will tend to find only a few large clusters.

- $MinPs$: the core polygon threshold. It is the minimum number of the shared nearest neighbors required for core polygons. It allows the user to control how many polygons are needed to qualify a polygon as a core polygon. $MinPs$ should be smaller than k.

Both ST-SNN and ST-SEP-SNN follow the structure of SNN. The time complexity of ST-SNN and ST-SEP-SNN are the same as that of SNN which is $O(n^2)$ without the use of an indexing structure, where n is the number of polygons in the datasets. If an indexing structure such as a k-d tree or an R* are used, the time complexity will be reduced to $O(n \times log(n))$. The space complexity is $O(k \times n)$ since only the k-nearest neighbor need to be stored, while the k-nearest neighbor can be computed once and used repeatedly for different runs of the algorithms with different parameter values.

4.3 Change-pattern-discovery and post-processing analysis techniques

In this section we introduce the polygon-based change-pattern-discovery algorithm and the post-processing analysis techniques for spatio-temporal clusters in detail. The polygon-based change-pattern-discovery algorithm allows domain experts to observe the dynamic changes of the spatio-temporal clusters across both spatial and temporal dimensions. Our post-processing analysis techniques help domain experts identify interesting clusters based on their notion of interestingness.

4.3.1 Polygon-based change-pattern-discovery algorithm

In this section we introduce a method for discovery and analysis of changes that may occur within a spatio-temporal cluster of polygons. Individual changes are categorized according to the spatial relationship between polygons on two snapshots. The following primitive change patterns are considered:

- Formation: when the number of polygons at time t_i increases from time t_{i-1}.
- Expansion: the overall areas covered by all polygons that occurred at time t_i is increased compared to time t_{i-1}.
- Dissipation: the overall areas covered by all polygons that occurred at time t_i is decreased compared to time t_{i-1}.
- Disappear: the number of polygons that occurred at time t_i is changes to zero

from non-zero at time t_{i-1}.

These individual changes can also be linked into sequences to describe the evolution of a spatio-temporal cluster over time. Continuation measures the duration started from formation until disappear within a spatio-temporal cluster. The pseudo-code of the polygon-based change-pattern-discovery algorithm (Poly-CD) is given in Algorithm 5.

Input: *set of spatio-temporal clusters* C_i
Output: *a vector of change pattern of cluster* C_i*, continuation of* C_i
Algorithm:

1. *Initialize vector changes = null for every spatio-temporal cluster* C_i *in the dataset* D*.*
2. *For each time* t *in* C_i*, compute the number of polygons* N_t *that occurred at time* t*.*
3. *Compute the change patterns for cluster* C_i *based on every two continues time* t_i *and* t_{i-1}*:*
   if $N_t \geq 1$ *AND* $N_{t-1} = 0$ *then changes.add (formation); continuation = 1;*
   if $N_t \geq 1$ *AND* $N_{t-1} \neq 0$ *AND* $area_t > area_{t-1}$ *then changes.add (expansion); continuation++;*
   if $N_t \geq 1$ *AND* $N_{t-1} \neq 0$ *AND* $area_t < area_{t-1}$ *then changes.add (dissipation); continuation++;*
   if $N_t == 0$ *AND* $N_{t-1} \geq 1$ *then change.add (disappear); continuation++;*
4. *return the change pattern vector and the continuation of cluster* C_i

Algorithm 5: Pseudo-code of Poly-CD

Poly-CD also measures the spatial properties of multiple polygons, such as overlap and distance. The Hausdorff distance function [3, 4] is applied to compute the distance between a pair of polygons. The Overlay distance function [28] is utilized to calculate the overlap between a pair of polygons.

4.3.2 Post-processing analysis technique for spatio-temporal data

Our post-processing analysis technique for spatio-temporal data allows automatic screening of the spatio-temporal clusters of polygons for interesting clusters, whose members have attribute values that deviate significantly from those of the entire population. For example, we try to find clusters of polygons with attribute values that are much smaller or larger than others.

It is desirable to have some assessment of the degree to which the attribute values of a cluster are anomalous. Since box plots are commonly used for showing the distribution of values of a single numerical attribute, and for comparing how attribute values vary among different clusters of objects, our post-processing analysis technique is developed based on box plots. We assume a dataset D with n attributes, and a set of clusters in D identified by a spatio-temporal clustering algorithm, e.g., ST-SEP-SNN. Let $(a_{i,j}, b_{i,j})$ be the interquartile range (IQR) for attribute j of a cluster, C_i, with $a_{i,j} > b_{i,j}$, and let (a'_j, b'_j) be the IQR for attribute j of the dataset D with $a'_j > b'_j$; we compute the degree of deviation for attribute j in the cluster C_i compared with the dataset D as follows:

$$R_{i,j} = 1 + \frac{max(a'_j - a_{i,j}, 0) + max(b'_j - a_{i,j}, 0) - max(a'_j - b_{i,j}, 0) - max(b'_j - b_{i,j}, 0)}{a_{i,j} - b_{i,j}} \tag{4.6}$$

$R_{i,j}$ could be any number between -1 and 1. We explain the equation described above with the help of several examples shown in Figure 4.1, which displays the box plots for attribute j of several selected clusters and the dataset D from the ozone pollution

datasets that we use for our case studies.

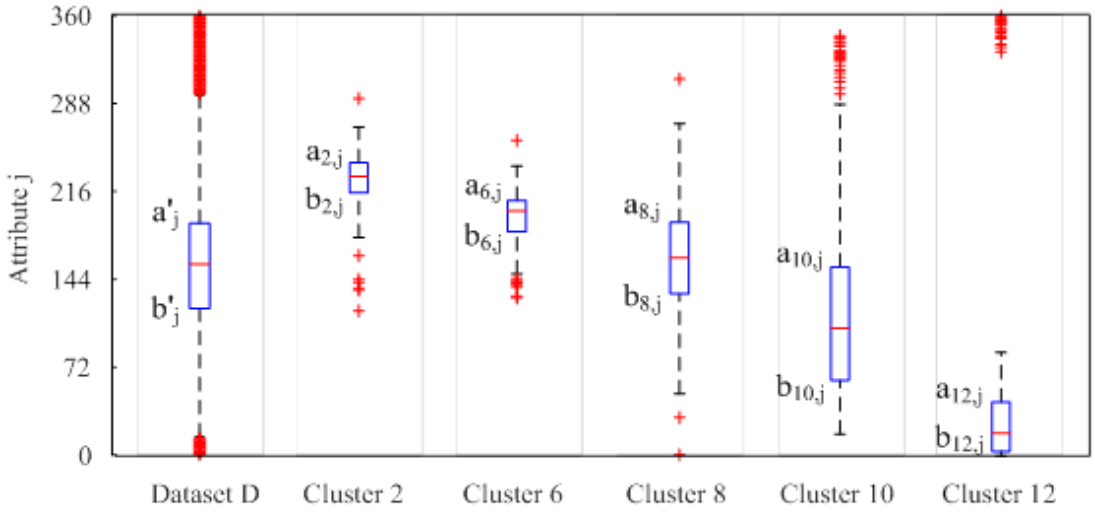

Figure 4.1: Box plots for attribute j

For example, the deviation degree of attribute j for cluster 2, $R_{2,j}$ can be calculated as:

$$R_{2,j} = 1 + \frac{0 + 0 - 0 - 0}{a_{2,j} - b_{2,j}} = 1 \tag{4.7}$$

where $a_{2,j} > b_{2,j} > a'_j > b'_j$ holds as shown in Figure 4.1.

Consider cluster 6, its box plot overlaps with the box plot for the dataset D and $a_{6,j} > a'_j > b_{6,j} > b'_j$. $R_{6,j}$ can be calculated as:

$$R_{6,j} = 1 + \frac{0 + 0 - (a'_j - b_{6,j}) - 0}{a_{6,j} - b_{6,j}} = \frac{a_{6,j} - a'_j}{a_{6,j} - b_{6,j}} \tag{4.8}$$

Apparently, $0 < R_{6,j} < 1$; therefore, for all box plots that overlap with the box plot for the dataset D above its 50^{th} percentile line, the deviation degrees will be greater than 0 but less than 1.

For cluster 8, its box plot is covered completely by the box plot for the dataset D, and $a'_j > a_{8,j} > b_{8,j} > b'_j$:

$$R_{8,j} = 1 + \frac{(a'_j - a_{8,j}) + 0 - (a'_j - b_{8,j}) - 0}{a_{8,j} - b_{8,j}} = 0 \tag{4.9}$$

Consider cluster 10, its box plot overlaps with the box plot for the dataset D below its 50^{th} percentile line, and $a'_j > a_{10,j} > b'_j > b_{10,j}$:

$$R_{10,j} = 1 + \frac{(a'_j - a_{10,j}) + 0 - (a'_j - b_{10,j}) - (b'_j - b_{10,j})}{a_{10,j} - b_{10,j}} = \frac{b_{i,j} - b'_j}{a_{10,j} - b_{10,j}} \tag{4.10}$$

Obviously, $-1 < R_{10,j} < 0$. For cluster 12, its box plot is displayed below the box plot for the dataset D, and $a'_j > b'_j > a_{12,j} > b_{12,j}$:

$$R_{12,j} = \frac{-a_{12,j} + b_{i,j}}{a_{12,j} - b_{12,j}} \tag{4.11}$$

In general, $R_{i,j}$ is interesting if $R_{i,j}$ is equal to 1 or -1, which means that a cluster C_i has significant different values of attribute j compared to the dataset D. The interestingness score of a cluster, C_i, is calculated based on the values of all $R_{i,j}$ associated with C_i. Let $O_i = \{r_{i,1}, r_{i,2}, \ldots, r_{i,n}\}$ be the set of deviation degrees of n attributes in C_i; in general, the interestingness score of a cluster C_i is a function of O_i:

$$I(C_i) = f(O_i) \tag{4.12}$$

Different interestingness functions may be adopted for different analysis tasks. Domain knowledge with respect to what patterns are interesting is crucial in determining interestingness functions. In our case, we plan to mine and analyze multi-source spatial air quality data involving in ozone pollution events in the HGB area.

Ozone formation involves complicated chemical reactions. There are several control factors involved, such as solar radiation, outdoor temperature, wind speed, wind direction, and NOx concentration; therefore, in addition to the ozone concentration, we also collected several corresponding meteorological data and the NOx concentrations reported from different sensor networks. An interestingness function for ozone pollution data is proposed as follows based on the domain experts' notion of interestingness. For a cluster, C_i, we first calculate the Ozone Formation Potential Index (OFPI) defined by the following equation:

$$OFPI_i = \frac{1}{3}R_{i,no_x} + \frac{1}{3}R_{i,T} + \frac{1}{3}R_{i,SR} \tag{4.13}$$

where R_{i,NO_x} is the degree of deviation of NOx concentrations in cluster $C_i, R_{i,T}$is the degree of deviation of outdoor temperatures in cluster $C_i, R_{i,SR}$ is the degree of deviation of solar radiations in cluster C_i. Note that the OFPI function is a linear function of these three variables, i.e., $R_{i,NO_x}, R_{i,T}$, and $R_{i,SR}$, because NOx concentration, outdoor temperature, and solar radiation are key control factors in ozone formation. The negative $R_{i,j}$values will contribute negatively to the OFPI because lower temperature, lower NOx concentration, and lower solar radiation will slow down the ozone formation process; however, the other two attributes, i.e., wind speed, and wind direction, contribute to the ozone pollution dispersion process; therefore, the ozone Dispersion Index (DI) for a cluster, C_i, is calculated as follows:

$$DI_i = exp(1 - 0.4 \times |R_{i,WD}| + 0.6 \times R_{i,WS}) \tag{4.14}$$

where $R_{i,WD}$ is the degree of deviation of wind direction in cluster $C_i, R_{i,WS}$ is the degree of deviation of wind speed in cluster C_i. Note that the dispersion index is

an exponential function due to the fact that the dispersion distribution satisfies the Gaussian air pollutant dispersion equation. We use the absolute value of $R_{i,WD}$ because the lower degree of wind direction and the high degree of wind direction contribute equally to the impact scope of an ozone dispersion process. We compute the interestingness score for a cluster, C_i, as follows:

$$IS_i = OFPI_i \times DI_i \tag{4.15}$$

The proposed interestingness function is just an example for identifying interesting clusters related to the ozone pollution impact scope. Our goal is to find clusters with either relative high or relative low interestingness scores, i.e., clusters with either relative large or small ozone pollution impact scopes under unusual environmental conditions and NOx concentrations. Those are the anomalous clusters that the domain experts are interested in and want to further analyze. Other interestingness functions for different analysis tasks can be developed as well.

4.4 Case study for spatio-temporal clustering algorithms

In this section we focus on demonstrating the effectiveness of both ST-SNN and ST-SEP-SNN in finding interesting spatio-temporal patterns from ozone pollution events in the HGB area. A spatio-temporal cluster of polygons is a group of polygons representing high ozone concentration hotspots that are in close proximity in both

space and time, and possibly share other attributes. Statistics and cluster post-processing analyses are also presented to interpret the discovered patterns. The Hybrid distance function [29] is applied to compute the spatial distance between a pair of polygons due to its capability of handling overlapping polygons.

Any functions that can compute the temporal distance between a pair of polygons can be used in this case study. Moreover, different temporal distance functions may be adopted for different analysis tasks.

4.4.1 ST-SNN clustering and analysis

The task of this case study is to apply ST-SNN to find interesting spatio-temporal patterns from the ozone pollution data in the HGB area. In particular, we are interested in finding hourly patterns of ozone pollution events that occurred in similar areas. This type of study can help domain experts identify not only similar impact scopes of the ozone pollution events in space but also their corresponding time instants or time intervals, which can help domain experts to gain knowledge from the past. Unlike the time slicing approaches, which perform snapshot clustering at each time stamp or time interval, ST-SNN takes into account both spatial and temporal distances between polygons and is able to detect clusters of polygons that are similar in both spatial and temporal dimensions simultaneously. The following temporal distance function is proposed for this case study:

$$dist_t(p,q) = \begin{cases} h(p) - h(q), & abs(h(p) - h(q)) < 12 \\ 24 - abs(h(p) - h(q)), & \text{otherwise} \end{cases} \tag{4.16}$$

where p and q are two polygons, the $h(p)$ function returns the hour information associated with polygon p, and the function $abs()$ returns the absolute value. The input parameters for ST-SNN are $k = 5, MinPs = 3$, and $w = 0.5$. We use a relative smaller value of k, i.e., 5, to find smaller compact clusters. There are 17 clusters found by ST-SNN. Figures 4.2, 4.3, 4.4, and 4.5 visualize four clusters, i.e., clusters 9, 11, 14, and 16, respectively.

As expected, the ST-SNN algorithm is able to find clusters of polygons that are very similar in space and time; for example, ST-SNN could successfully identify 17 clusters of polygons that are very close spatially and occurred at the same time on different dates. We also identify the locations of the emission sources and monitor stations represented by the blue points and red points, respectively on each figure. Cluster 9 and cluster 11 lie closely in space, but are identified as two different clusters due to different temporal similarities, i.e., all five polygons in cluster 9 occurred at 1 pm, whereas all four polygons in cluster 11 occurred at 3 pm. Cluster 14 and cluster 16 are also detected due to their high spatial and temporal similarities. All five polygons in cluster 14 occurred at 2 pm along highway interstate 10 east. All six polygons in cluster 16 occurred at 3 pm along highway interstate 45 north. Clusters 6 and 14 are formed due to the emissions from highway traffic vehicles and strong solar radiation which usually happen between 2 pm and 3 pm each day.

Further we closely inspect cluster 11. Polygons 37, 157, 112, and 80 in cluster 11 are numbered in ascending order according to their impact scopes. Their extending directions are to the east. The box plots shown in Figure 4.6 provide the statistical comparisons of NOx concentration, solar radiation, outdoor temperature, wind

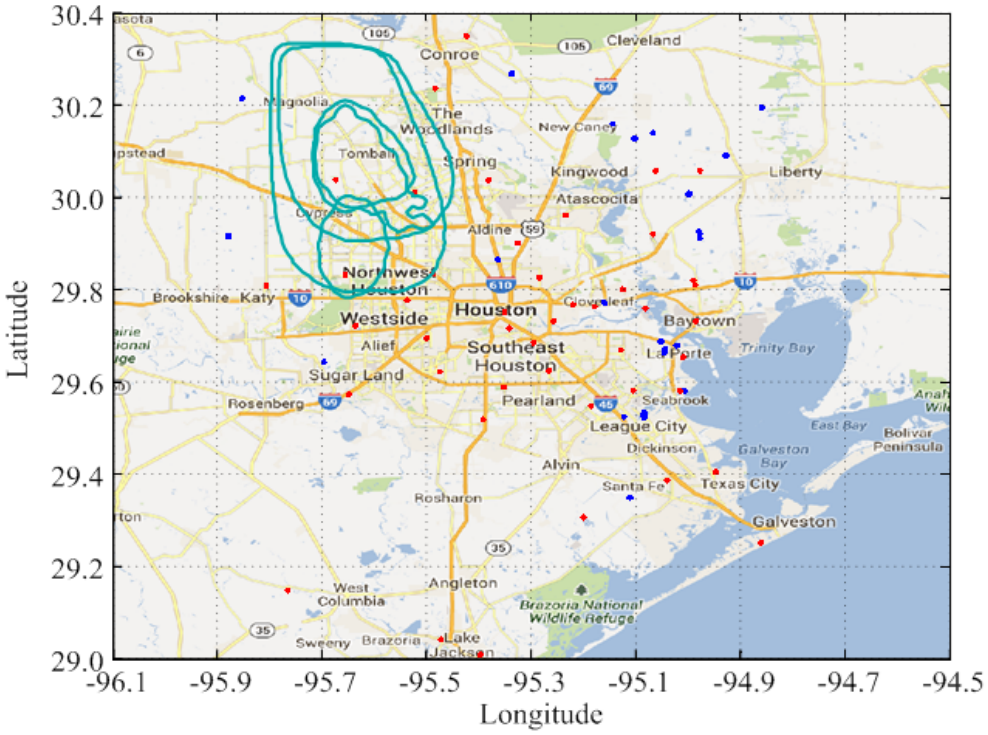

Figure 4.2: Visualization of clusters 9 identified by ST-SNN

speed, wind direction, and relative humidity of these four polygons, respectively. It can be observed that the majority of wind directions associated with these polygons distribute in the domain between 60 and 180, which means that the wind comes from the Gulf of Mexico as the result of a typical sea breeze encompassing east-southeast to south on the shore winds in the HGB area [30]. It transports the emissions from both point sources and highway traffic vehicles that locate at the upwind direction to this area, creating a suitable condition for ozone formation in this area. Polygon 80 has a relatively stronger solar radiation which facilitates the ozone formation, a higher wind speed which speeds up the emission transportation; therefore, polygon 80 has the largest impact scope in cluster 11. Polygon 37 has the lowest wind

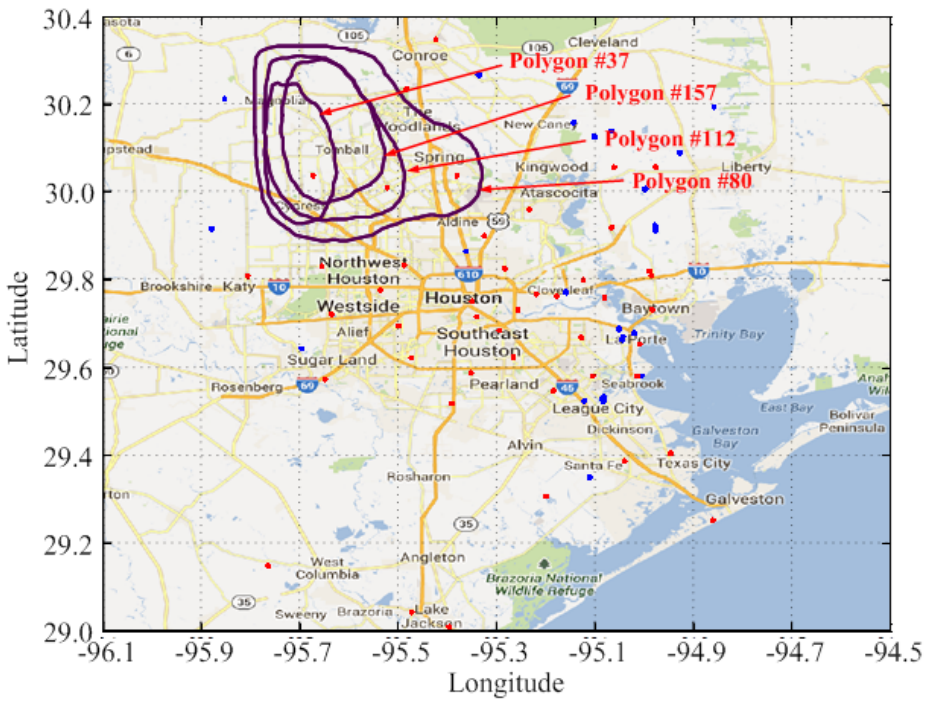

Figure 4.3: Visualization of clusters 11 identified by ST-SNN

speed, the largest distribution of wind directions, and low solar radiation, which cause polygon 37 to have the smallest impact scope in cluster 11, even though it has higher NOx concentration compared with other polygons in cluster 11. All the information suggests that the wind speed, wind direction, and solar radiation may be major contributive factors for the impact scopes of the ozone pollutant events in this area, while NOx concentration, relative humidity, and outdoor temperature may not. It also demonstrates that the land and sea breeze can affect ozone formation by transferring the emissions from the emission sources.

If large values of k, such as 8, or 10, are used, ST-SNN could find fewer clusters. If we increase the weight associated with spatial distance w, clusters that are closer in

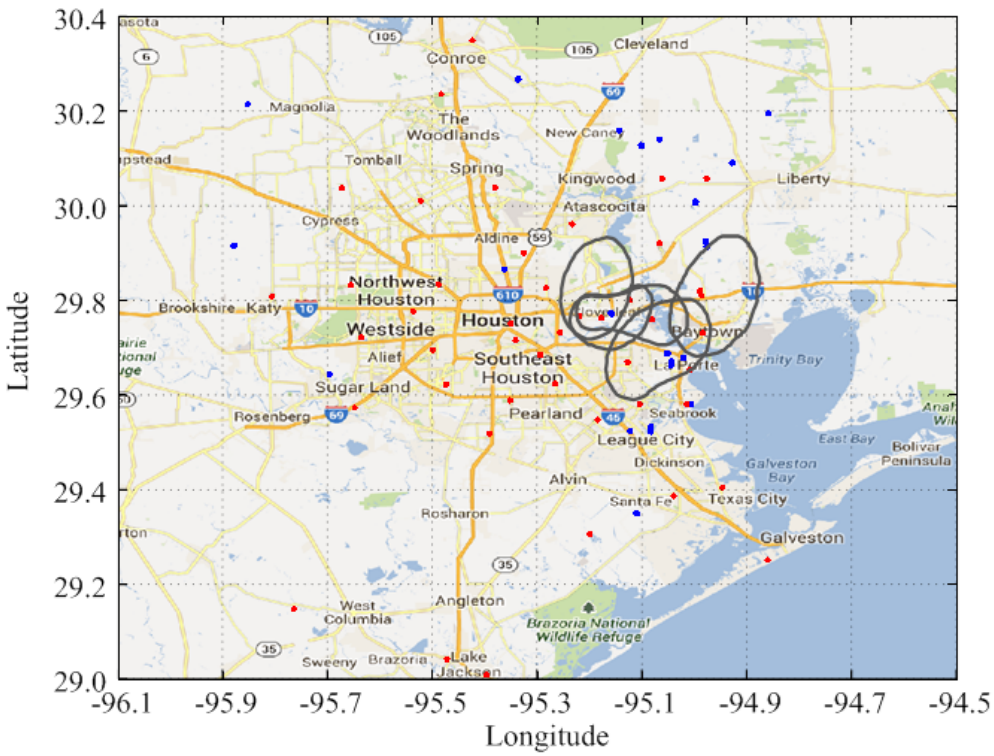

Figure 4.4: Visualization of clusters 14 identified by ST-SNN

space and less close in time will be identified, such as clusters of polygons that occur within several hour intervals, while lying very closely in space. If we use w equal to 1.0, ST-SNN becomes a spatial clustering algorithm. We can also adopt different temporal distance functions, for example, in order for ST-SNN to find groups of polygons that lie closely in space and occur within certain time intervals instead of at a particular time instant; the following temporal distance function could be adopted:

$$dist_t(p,q) = \begin{cases} 1, & abs(h(p) - h(q)) < t \\ 0, & \text{otherwise} \end{cases} \tag{4.17}$$

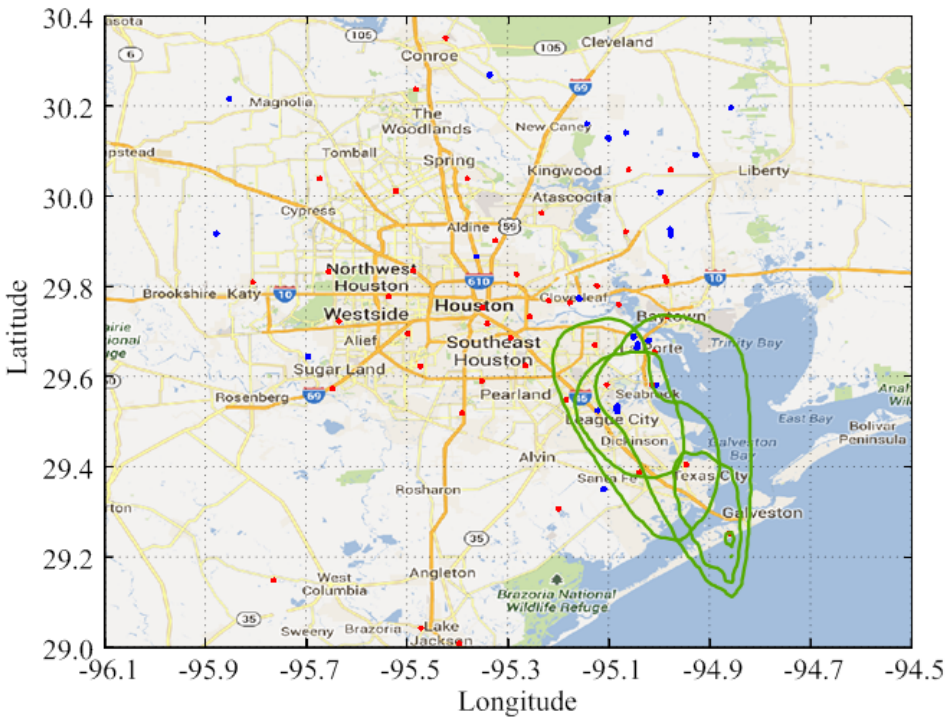

Figure 4.5: Visualization of clusters 16 identified by ST-SNN

where p and q are two polygons, $h(p)$ returns the value of hour information associated with polygon p, function $abs()$ return the absolute value, and t is the time interval threshold input by the users.

4.4.2 ST-SEP-SNN clustering and analysis

In this case study, we apply ST-SEP-SNN and the same temporal distance function discussed in Section 4.4.1 for comparison purpose. The input parameters for ST-SEP-SNN are $k = 5$ and $MinPs = 3$. There are 12 clusters identified by ST-SEP-SNN. Six of them contain polygons that occurred at the same time on different dates at similar locations, which are very similar to the clusters identified by ST-SNN; however, the

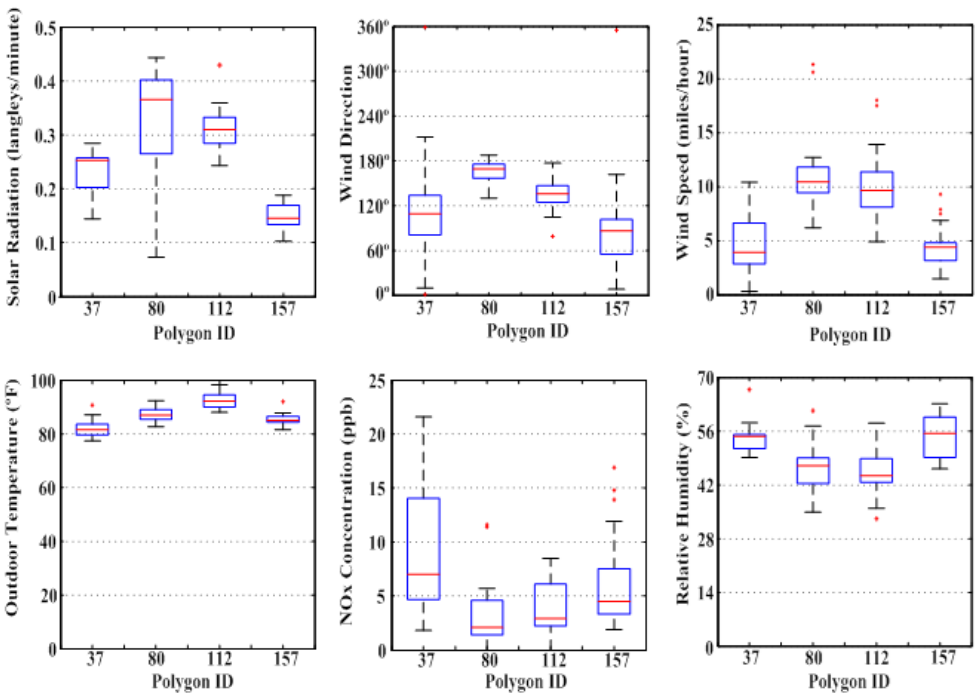

Figure 4.6: Boxp lots for cluster 11 identified by ST-SNN

other six clusters that contain polygons occur within certain time intervals, such as three or four hours, instead of at a particular time. Unlike ST-SNN, ST-SEP-SNN intends to group polygons that lie at similar locations but occur within certain time intervals into one large cluster instead of dividing them into two or more smaller compact clusters. This is because ST-SNN uses the weighted sum of spatial distance and temporal distance to compute the spatio-temporal distance between a pair of polygons; then it ranks the spatio-temporal distance matrix to get the k-nearest neighbor list for each polygon, whereas ST-SEP-SNN first ranks both the spatial distance matrix and the temporal distance matrix separately; next it computes the nearest neighbor list using the intersection of the k-nearest spatial neighbor list and

the k-nearest temporal neighbor list for each polygon; thus, the k-nearest temporal neighbor list may include some polygons that have larger temporal distances if they were ranked in the top k, which may be excluded in the k-nearest neighbor list by ST-SNN due to their large temporal distances. Figures 4.7 and 4.8 visualize two such clusters, namely clusters 4 and 9.

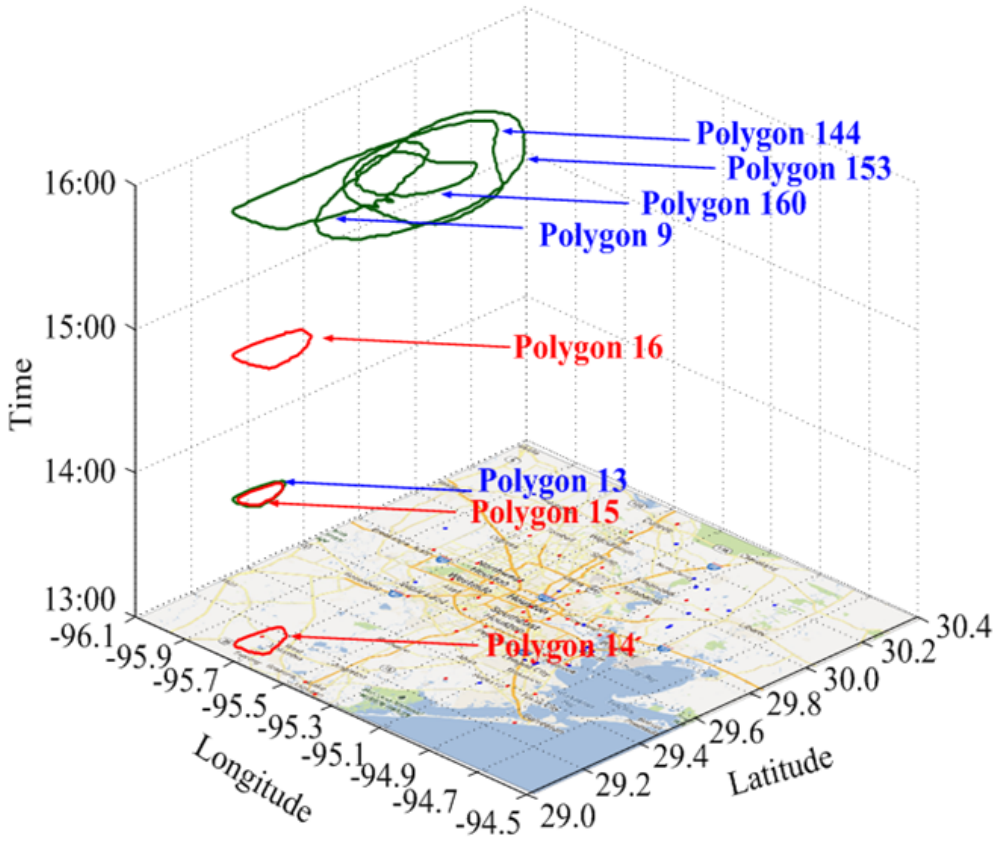

Figure 4.7: Visualization of cluster 4 identified by ST-SEP-SNN

Cluster 4 is an interesting cluster. It includes eight polygons, i.e., polygons 9, 13, 14, 15, 16, 144,153, and 160. Polygons 14, 15 and 16, shown in red in Figure 4.7, occurred in three continuous hours, i.e., 1 pm, 2 pm, and 3 pm on the same day.

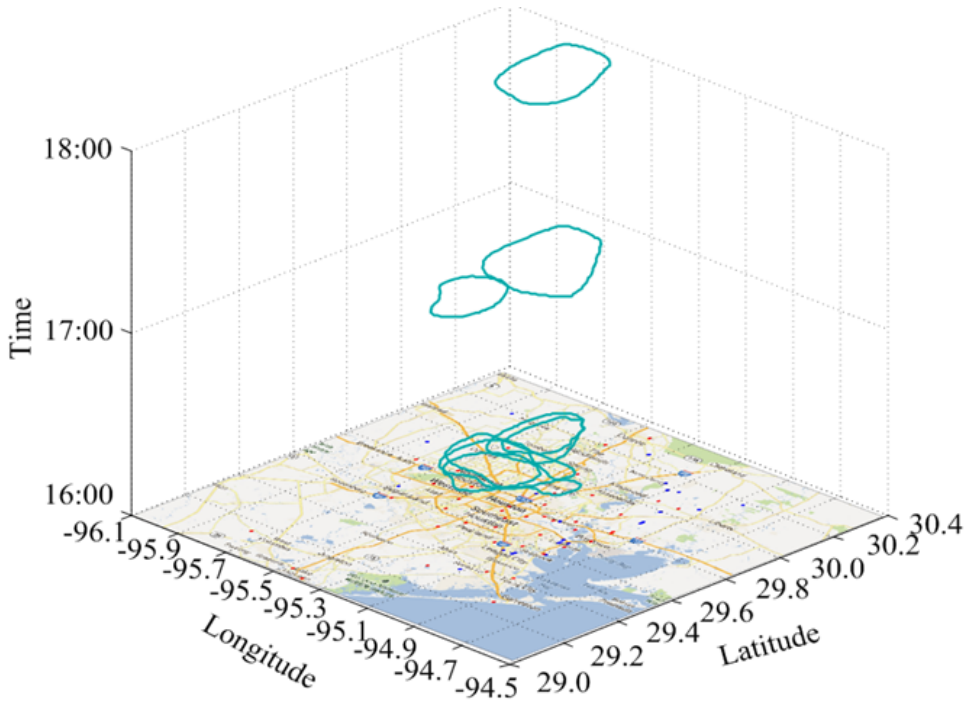

Figure 4.8: Visualization of cluster 9 identified by ST-SEP-SNN

They are in ascending order according to the area each polygon covered, and the scope extending direction is to the northeast. The box plots in Figure 4.9 provide the statistical comparisons of NOx concentration and related meteorological data.

Based on Figure 4.9, from 1 pm to 3 pm, the mean of the wind direction rotated clockwise and the wind speed increased gradually, which enhanced the dispersion effect; therefore the areas covered by polygons 14, 15, and 16 were enlarged; however, these enhancement factors were counterbalanced by the reduction of solar radiation from 1 pm to 3 pm, which resulted in the slow change of the areas covered by polygons 14, 15, and 16 over time. Compared with other polygons, these three polygons have large variances of solar radiation, wind direction, wind speed, and lower mean values of relative humidity, while their NOx concentrations and outdoor temperatures are

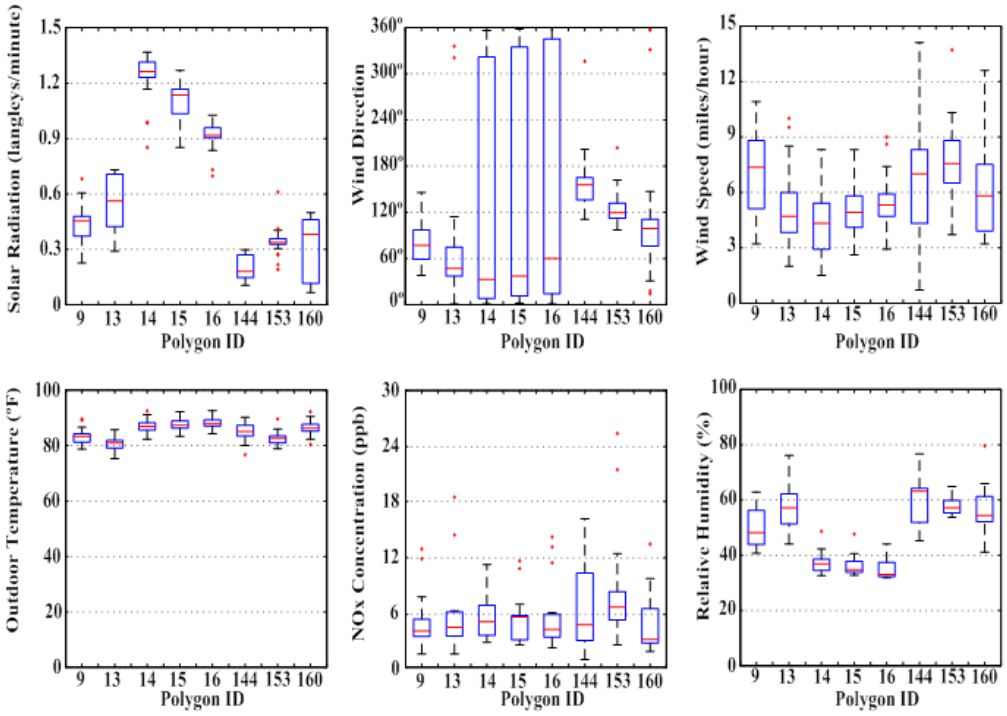

Figure 4.9: Box plots for cluster 4 identified by ST-SEP-SNN

quite similar to those of the other polygons. Further investigations of such clusters may help domain experts detect dynamic changes of the ozone pollution events in this area.

4.5 Case study for change-pattern-discovery and post-processing techniques

For this case study, we downloaded the ozone pollution data in the HGB area for the timeframe from 1 am on April 1, 2010 through 11 pm on November 30, 2010 from the website of the TCEQ [25]. The DCONTOUR algorithm [27] along with a

user defined density threshold, i.e., 80 *ppb* (parts per billion), is adopted to generate polygons. Each polygon represents the area with hourly ozone concentration higher than the threshold value of 80 *ppb*. The shape and area of a polygon reflect the impact location and the scope of an ozone pollution event, which are determined by multiple factors of the ozone precursors and meteorological conditions. A total of 460 polygons are identified. Figure 4.10 provides the statistical distributions of five corresponding meteorological attributes and NOx concentration. The bottom and top of each box are the first and third quartiles, respectively. The band inside a box is the second quartile (the median). The ends of the whiskers represent the minimum and maximum of each data categories. Any data not included between the whiskers are plotted as outliers.

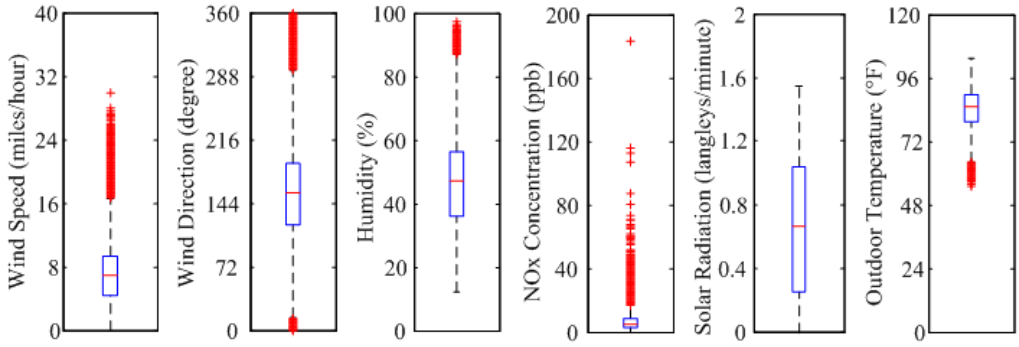

Figure 4.10: Box plots for ozone pollution data

It can be observed that the majority of the wind directions associated with all identified polygons are distributed between 120 and 200; the distribution of wind direction covers from 0 and 360, which means that the wind can come from any direction horizontally, e.g., the wind can blow from the Gulf of Mexico as the result of a typical sea breeze which encompass east-southeast to south on shore winds in

the HGB area [30]. It can help transport the emissions from point sources, area sources, and on-road mobile sources located at the upwind direction to different downwind regions, which cause ozone formation and transportation in these regions. The corresponding solar radiation is between 0.3 and 1.05 langleys/minute, which is relative strong and in favor of ozone formation. The higher wind speed between 4 and 9 miles/hour promotes the precursor transportation. The average NOx concentration is below 10 *ppb* and the peak concentration is 183 *ppb*. The outdoor temperatures are between 75 °F and 90 °F. In summary, the integrated condition of high outdoor temperature, high NOx concentration, and high solar radiation creates favorable conditions for high ozone concentration.

4.5.1 Change-pattern-discovery algorithm evaluation

In this case study, we use the Hybrid distance function [29] to compute the spatial distance between polygons due to its capability of handling overlapping polygons. The spatial distance threshold θ_s is set to the average distance of the datasets; the temporal distance threshold θ_t is set to 24 hours because domain experts are interested in the daily ozone pollution spatio-temporal patterns. Note that different temporal distance functions and thresholds can be adopted for different analysis tasks. We use a relative small number as core polygon threshold $MinPs$, i.e., 2. There are 45 clusters identified by ST-SNN. Figures 4.11, 4.12, and 4.13 visualize three clusters, i.e., clusters 10, 12, and 35, respectively. The centroid of each polygon is calculated and marked as a red dot.

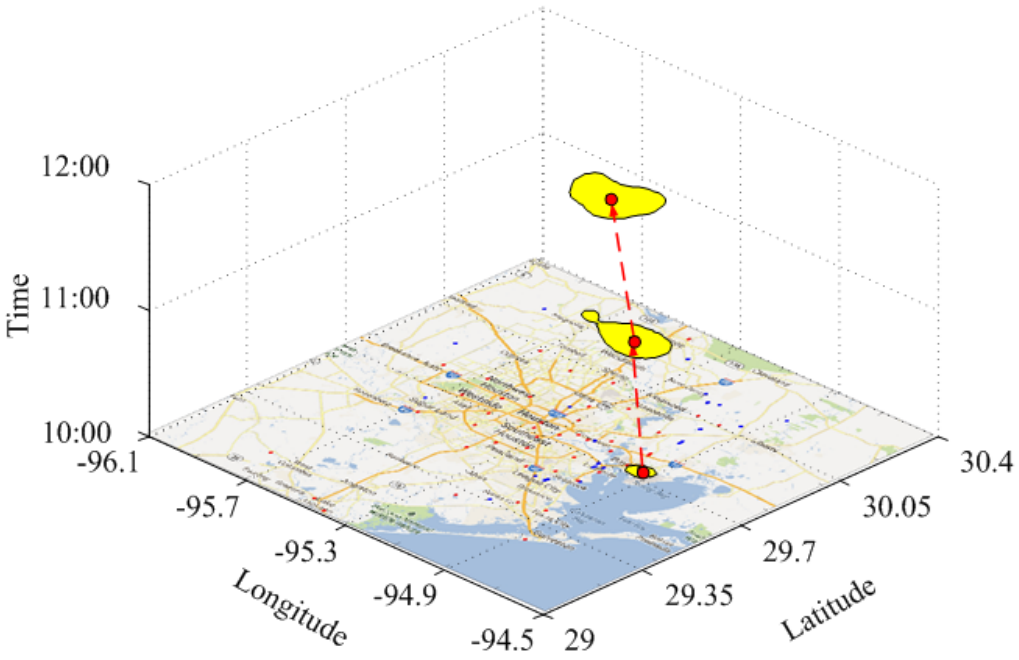

Figure 4.11: Visualization of cluster 10

The dynamic change patterns of the daily ozone pollution events in a specific region are very complex. The main concern is the expansion and duration of the impact region during a ozone pollution event. A long duration time means that the corresponding ozone pollution event is very serious, and may be caused by abnormal emissions from local industrial plants. For example, the continuation of cluster 35 is five hours. The impact regions of the ozone pollution events in cluster 35 have been largely expanded during five continuous hours because the wind is not fast enough to effectively transport the ozone pollution to the downwind direction. The change pattern for cluster 35 is $formation \rightarrow expansion \rightarrow expansion \rightarrow expansion \rightarrow expansion \rightarrow disappear$.

Further we closely inspect cluster 10, which is a typical change pattern of daily

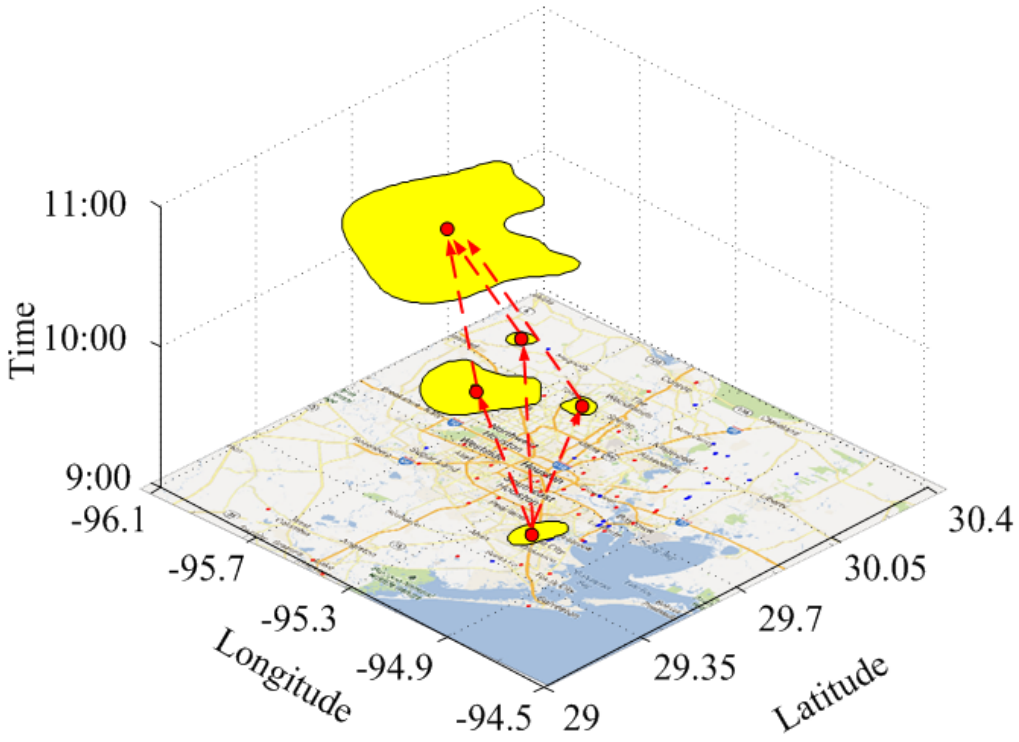

Figure 4.12: Visualization of cluster 12

ozone pollution events. The dynamic profiles of NOx concentration, solar radiation, and outdoor temperature for all polygons in cluster 10 are shown in Figures 4.14, 4.15, and 4.16, respectively. Figure 4.14 displays the dynamic changes of 3D NOx concentration profiles disclosed from 10 am to 12 pm. The peak NOx concentration occurred at 11 am. The maximum value of solar radiation occurred at 12 pm as shown in Figure 4.15. Note that an ozone pollution event was started at 10 am due to the impact of high NOx concentration, solar radiation, and outdoor temperature; in the following continuous two hours, the pollution hotspots were expanded because

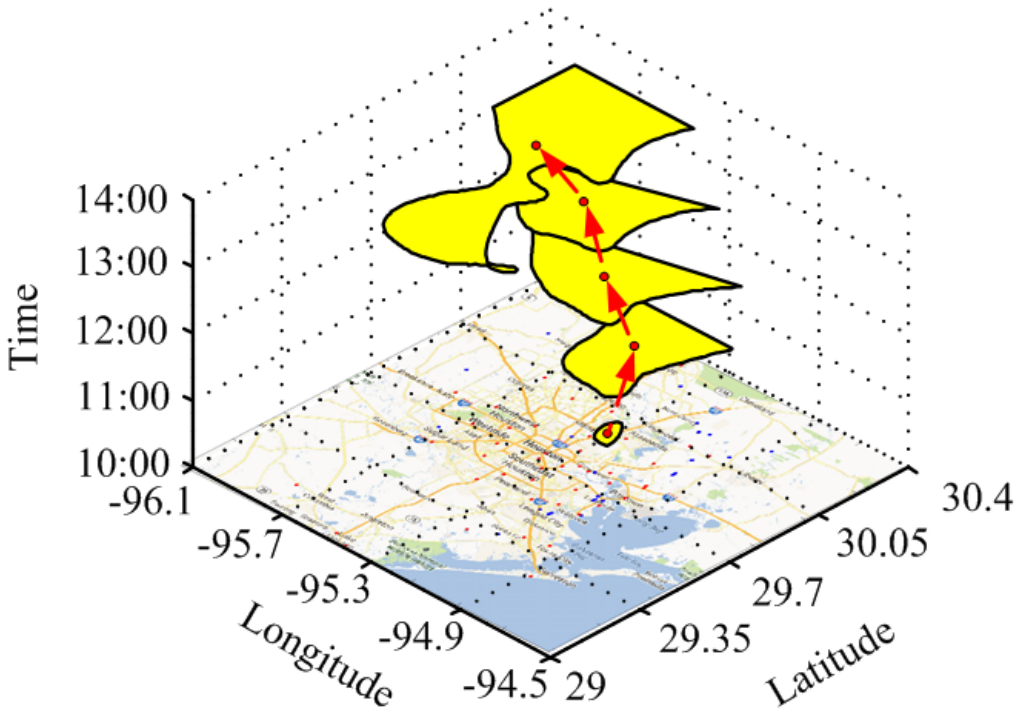

Figure 4.13: Visualization of cluster 35

the major impacts were enhanced; meanwhile, the pollution hotspots were continuously moving toward the northwest direction partially due to the impact of the wind flow.

4.5.2 Post-processing analysis technique evaluation

The goal of our post-processing analysis technique is to help domain experts identify interesting clusters that are unusual compared to all other clusters. Figures 4.17 and 4.18 visualize two such clusters identified by our post-processing analysis technique, i.e., clusters 23 and 26, respectively. It can be observed that cluster 23 has

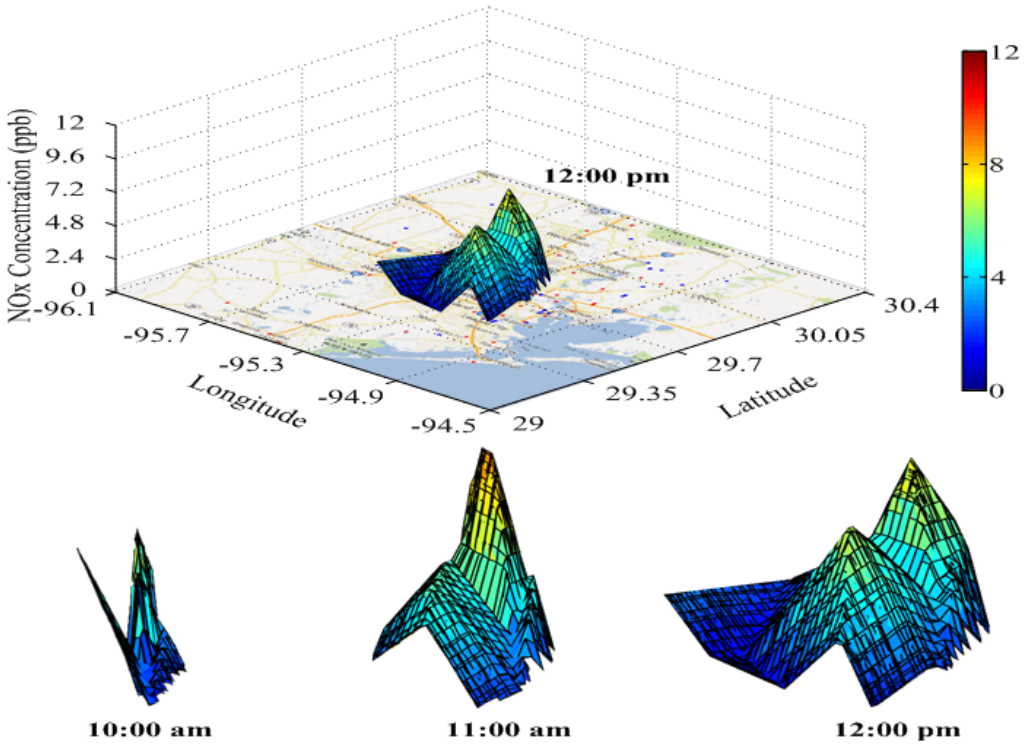

Figure 4.14: Dynamic changes of 3D NOx concentration profiles for cluster 10

a very small impact scope within a two hour duration, whereas cluster 26 has a relatively larger impact scope within a three hour duration. Table 4.1 summarizes the corresponding results of Ozone Formation Potential Index (OFPI), Dispersion Index (DI), and the Interestingness Scores (IS) for clusters 23 and 26. The deviation degrees of all attributes in clusters 23 and 26 are summarized in Table 4.2. The OFPI value of cluster 23 is 0.13 because cluster 23 has relative low values of solar radiation (deviation degree 0.39), and low NOx concentration (deviation degree -1) compared with the entire dataset. Furthermore, cluster 23 has the lowest DI value (1.00) among all clusters because it has the lowest wind speed and smallest range

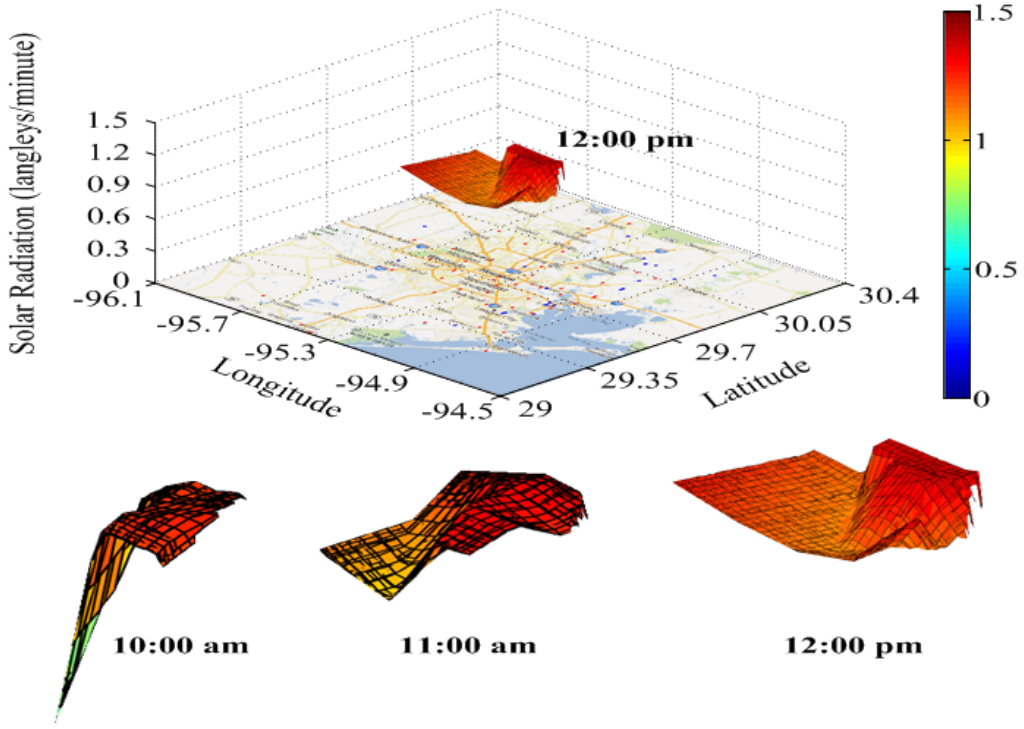

Figure 4.15: Dynamic changes of 3D solar radiation profiles for cluster 10

of wind direction since the deviation degrees are -1 for both attributes shown in Table 4.2. On the contrary, cluster 26 has larger values of OFPI (0.70) and DI (1.78) due to the high values of solar radiation (deviation degree 1) and high values of temperature (deviation degree 1); therefore, cluster 26 has a larger IS value (1.25).

Table 4.1: The interesting scores of clusters 23 and 26

Cluster ID	OFPI	DI	IS
23	0.13	1.00	0.13
26	0.70	1.78	1.25

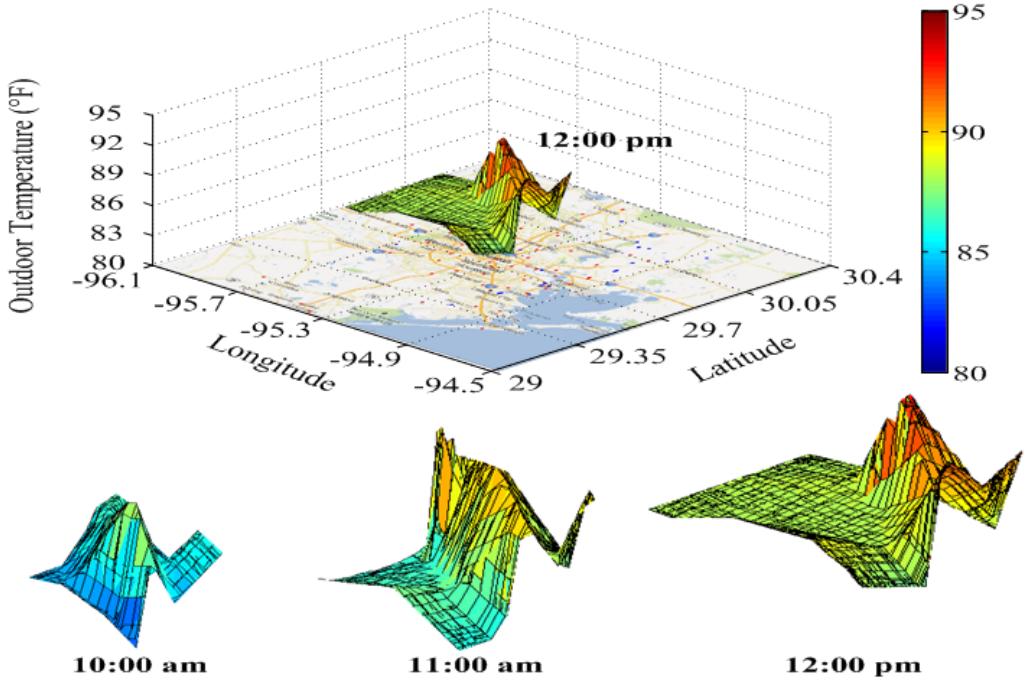

Figure 4.16: Dynamic changes of 3D outdoor temperature profiles for cluster 10

Table 4.2: The deviation degrees of clusters 23 and 26

Cluster ID	Temperature	Wind Direction	Wind Speed	NOx	Solar Radiation
23	1.00	−1.00	−1.00	−1.00	0.39
26	1.00	−0.19	−0.58	0.11	1.00

4.6 Conclusion

The main objective of this research was to develop novel spatio-temporal clustering algorithms for polygons and post-processing analysis techniques to analysis spatio-temporal clusters of polygons. Two density-based spatio-temporal clustering algorithms, called ST-SNN and ST-SEP-SNN, were developed by extending the generic Shared Nearest Neighbor clustering algorithm. We redefined the nearest

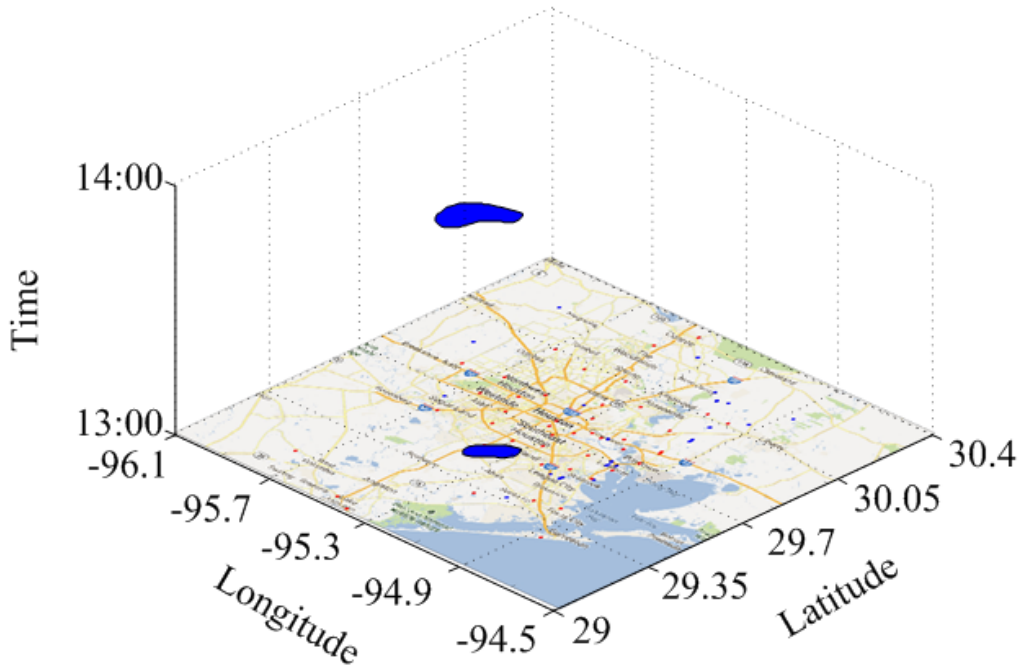

Figure 4.17: Visualization of cluster 23

spatio-temporal neighborhood of a polygon and the density-based concepts for polygons. Both ST-SNN and ST-SEP-SNN can find clusters of varying shapes, sizes, and densities in high dimensional data, even in the presence of outliers. We also proposed a change-pattern-discovery algorithm to atomically detect and analyze dynamic changes within spatio-temporal clusters of polygons and a post-processing analysis technique to identify interesting spatio-temporal clusters of polygons for domain experts. Experiments on multiple real spatio-temporal data involving ozone pollution events in the HGB area demonstrate that our method is effective and can discover interesting spatio-temporal patterns and change patterns of ozone pollution events from spatio-temporal datasets. Moreover, statistic and post-processing analysis techniques can help domain experts to identify interesting patterns of ozone

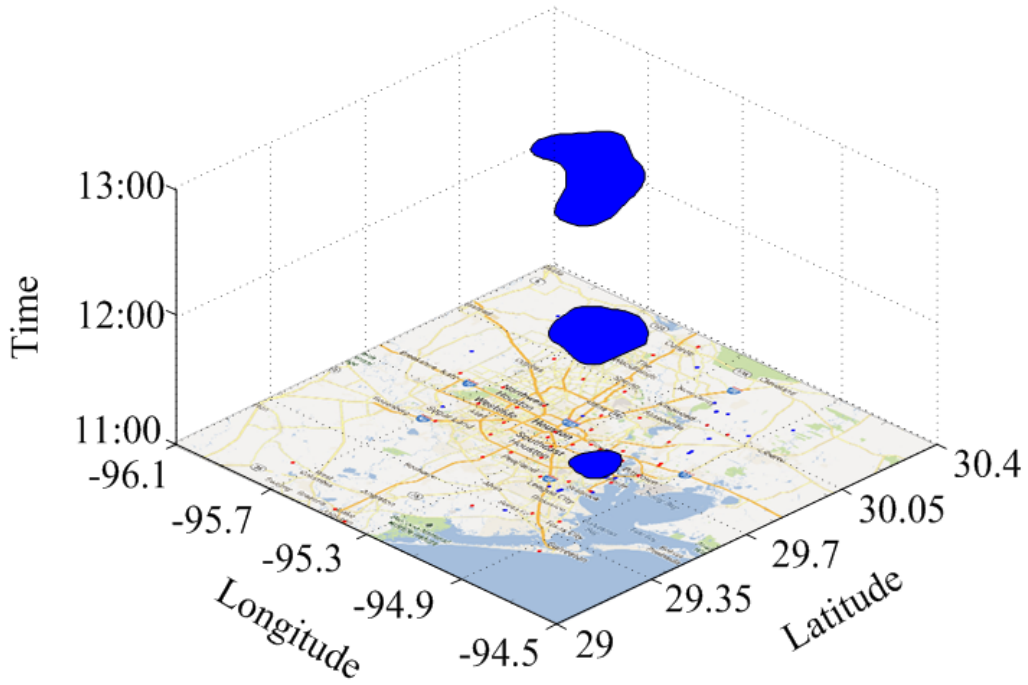

Figure 4.18: Visualization of cluster 26

pollution events in the HGB area, and to learn from the past and be better prepared for the future.

Chapter 5

Spatial clustering for analyzing the composition of cities

5.1 Introduction

"Urbanization is the physical growth of urban areas as a result of global change where increasing proportion of the total population becomes concentrated in towns. The United Nations reported that since 2008 more than half of the world's population is living in urban areas"[31]; thus, mastering urban evolution becomes a major challenge for all major cities in the world. Consequently, there is a growing need to develop urban computing and analysis tools to guide the orderly development of cities, as well as enhance their smooth and beneficiary evolution. The evolution of a city is a very dynamic activity; therefore, modeling the dynamics of urban evolution is a quite challenging task. Recently, data describing cities is widely available as

they are collected on a regular basis, offering a great opportunity to develop urban computing techniques, which can be used to analyze and model urban evolution. Understanding and monitoring urban evolution allows urban planners to make smarter decisions because they can provide deep insights into city changing dynamics. Moreover, it offers an opportunity to improve people's knowledge about the impacts from urbanization on the territory.

The step of urbanization leads to different functional regions in a city, called urban patches throughout the remainder of this chapter, such as residential areas, business districts, industrial and recreational areas. Different types of urban patches support different needs of people's lives and serve as a valuable organization technique for framing detailed knowledge of a metropolitan area [23]. Urban patches may be artificially created by urban planners, or may be the result of natural urban evolution; both could change functions and the territories with the development of a city.

Improvements in scanning devices, gps, and image processing lead to an abundance of geo-referenced data. For example, tracking devices are now available to capture the movement of human and animals in the form of trajectories [32]. Furthermore, more and more Point of Interest (POI) databases are created which annotate spatial objects with categories, e.g., buildings are identified as restaurants, and systems, such as Google Earth, already fully support the visualization of POI objects on maps. As more and more data become available for a spatial area, it is desirable to identify different functions and roles which different parts of this spatial area play; in particular, it is desirable to identify homogeneous regions in spatial data and to describe their characteristics, creating high-level summaries for spatial

datasets which are valuable for planners, scientists, and policy makers. For example, ecologists might be interested in partitioning a wetland area into uniform regions based on what animals and plants occupy this area and on other environmental characteristics [33]. Similarly, city planners might be interested in identifying uniform regions of a city with respect to the functions they serve for the people who live in or visit this part of a city [23].

More specifically in this research, we are interested in developing spatial clustering frameworks which are capable of creating summaries for an area of interest by identifying the spatial structure in spatial data and capturing its spatial heterogeneity. It should be stressed that traditional clustering algorithms are not suitable for this task - as they minimize distance-based objective functions or employ distance-based density estimation techniques - whereas assessing uniformity relies on non-distance based uniformity measures which operate on non-spatial attributes, such as purity, entropy or variance with respect to continuous non-spatial attributes. The focus of this chapter is the introduction of a method which identifies uniform regions in spatial data and provides analysis functions to create summaries for the identified uniform regions.

The rest of the chapter is organized as follows. Section 5.2 formally defines the problem of finding uniform regions in spatial data and introduces the spatial clustering approach for this task. Section 5.3 gives an experimental evaluation of the method. Section 5.4 concludes the chapter.

5.2 Using spatial clustering to discover uniform regions

5.2.1 Finding uniform regions in spatial data

Figure 5.1 gives an example of a spatial clustering result in which buildings of different types (e.g., schools and industrial buildings) of a city are clustered.

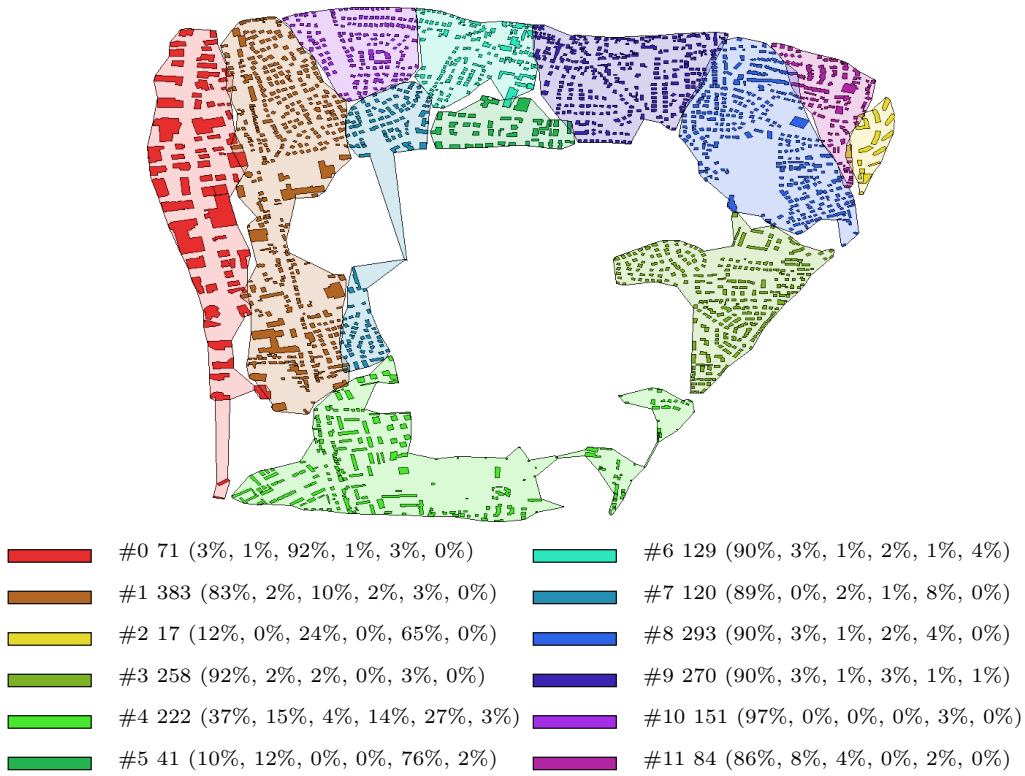

Figure 5.1: A spatial clustering of buildings belonging to different building types

The proposed method characterizes spatial clusters using their *scope* and *signature*. The scope of a spatial cluster captures the model of a cluster. In our approach,

we use polygons as models for spatial clusters as depicted in Figure 5.1; that is, if a spatial object is inside the polygon which describes the scope of a spatial cluster, it belongs to that spatial cluster. Secondly, the proposed method uses signatures to annotate spatial clusters. Signatures summarize the distribution of the objects that belong to a cluster. As the clusters in the example contain buildings belonging to different types, building-type histograms are used as signatures to annotate spatial clusters. There are six building types: single houses, garages, industrial buildings, light buildings, collective buildings, and schools. For example, the leftmost cluster is identified as cluster 0 containing 71 buildings, and its building-type signature is $(3\%, 1\%, 92\%, 1\%, 3\%, 0\%)$, indicating that 3% of the buildings in cluster 0 are single houses, 1% are garages, 92% are industrial buildings, 1% are light buildings, 3% are collective houses, and there are no schools in cluster 0.

So far we did not clearly discuss what distinguishes a uniform region of a city from one that is not uniform. In general, we assume that distribution signatures are used to characterize the objects that belong to an urban patch. Examples of such signatures include histogram-style building-type signatures which give the proportions of different building types that occur in an urban patch, such as 15% are commercial buildings and 85% are residential buildings. Moreover, the similarity between different building-type signatures can be easily assessed: for example, we could take the Euclidean distance between the vectors associated with different building-type signatures. More formally, we are interested in obtaining spatial clusters which are uniform with respect to their signatures using the following maximization procedure:

Input: *a dataset O containing spatial objects belonging to p classes*
Task: *Find a spatial clustering $X = \{C_1, \ldots, C_k\}$ of O such that*

(1) $C_i \subseteq O$ for $i = 1, \ldots, k$

(2) $C_p \cap C_q = \emptyset$ for $p \neq q$

which maximizes the following objective function $\varphi(X)$:

$$\varphi(X) = \sum_{C \in X \text{ and } C' \in X} \frac{d(s(C), s(C')}{b} \tag{5.1}$$

where b is the number of pairs of neighboring clusters in X, $s(C)$ denotes the signature of cluster C and $d()$ is a distance function which assesses the similarity of two signatures.

In summary, we are interested in obtaining a spatial clustering in which the average Euclidean distance between the signatures of neighboring clusters is as large as possible. It should be emphasized that only distances between neighboring clusters are considered in the definition of φ. In order to find uniform partitions, we can devise a search procedure which maximizes the disagreement of neighboring clusters with respect to their signatures; however, developing a spatial clustering algorithm which directly maximizes $\varphi(X)$ is quite challenging, as this would require to identify and to keep track of which spatial clusters are neighboring in order to compute $\varphi(X)$, which leads to quite significant clustering overhead, and to theoretical problems[1]. Consequently, we are using different heuristics to find uniform spatial clusters

[1]If prototype-based clustering algorithms, such as K-medoids or K-means are used, a Voronoi tessellation can be used to derive cluster models from the set of cluster prototype which are convex polygons; unfortunately, it is not computationally feasible to compute Voronoi cells in higher dimensional spaces, as the complexity of the algorithm is exponential with respect to the dimensionality of the dataset. Consequently, it is only feasible to compute the Voronoi tessellation in $1D$, $2D$, and

without having to deal with the question which clusters are neighboring, and rely on approaches which use simplified versions of $\varphi(X)$ instead; in particular:

1. We use prototype-based spatial clustering algorithms that are guaranteed to obtain contiguous spatial clusters without the necessity of knowing which clusters are neighboring. These algorithms maximize reward functions which encourage the merging of similar neighboring clusters and the splitting of non-homogeneous clusters if it leads to a significant increase in the total reward.

2. We reformulate the above optimization task in two ways:

   i. We make the problem supervised, by using interestingness functions which assess the quality of spatial clusters based on uniformity measures which capture a domain expert's notion of uniformity. Moreover, as we will see later, those uniformity measures assume that certain signatures are more desirable than others.

   ii. Instead of comparing the signatures of all neighboring clusters - as φ does - we employ an approach which identifies a set of popular[2] signatures and then uses those signatures to annotate clusters. In particular, this approach seeks for a spatial clustering which maximizes the match of a cluster's signature with the closest signature in the popular signature set, as will be explained in Section 5.3.3.

for small datasets in $3D$. For density-based clustering algorithm the situation is even worse; for example, we are not aware of any methods which are capable of producing cluster models from a DBSCAN clustering.

[2]Popular signatures are distribution characteristics which occur frequently in contiguous subspaces of a spatial dataset.

5.2.2 CLEVER

In order to employ these two approaches outlined in Section 5.2.1, we need a spatial clustering algorithm capable of finding contiguous spatial clusters by maximizing a plug-in reward function which captures a particular notion of uniformity. A spatial clustering algorithm named CLEVER [34, 35] will be adapted for this task. CLEVER is a prototype-based, k-medoid-style [36] spatial clustering algorithm which employs randomized hill climbing to maximize a plug-in reward function. Reward functions are assumed to have the following form when assessing the quality of a clustering $X = \{c_1, \ldots, c_k\}$:

$$q(X) = \sum_{c \in X} reward(c) = \sum_{c \in X} i(c) \times |c|^{\beta} \tag{5.2}$$

where $|c|$ denotes the number of objects in a cluster c, and $i(c)$ is an interestingness function which assesses how interesting the cluster c is. Two such interestingness functions will be introduced in Section 5.3. Moreover $\beta \geq 1$ is a parameter which determines how much reward is put on cluster size; β indirectly controls the number of the clusters in X, as cluster size is rewarded using a non-linear function. Usually fewer clusters are obtained when larger values for β are used. The reward function assesses the quality of a clustering as the sum of the rewards of all clusters; The pseudo-code of CLEVER is given in Algorithm 7.

CLEVER maintains a current set of representatives which are objects in the dataset and forms clusters by assigning the remaining objects in the dataset to the closest object in the representative set. It samples p solutions in the neighborhood of the current representative set by adding, deleting, and replacing representatives. This

Input: *Dataset O, distance function $d()$, $k^{'}$, $i(c)$, β, sampling rate p*
Output: *Clustering X, quality $q(X)$, rewards for clusters in X*
Algorithm:

1. *Randomly create a set of $k^{'}$ representatives from dataset O.*
2. *Sample p solutions in the neighborhood of the current representative set*
3. *If the best solution of the p solutions improves the clustering quality of the current solution, its representative set becomes the current set of representatives and search continues with Step 2; otherwise, terminate returning the current clustering.*

Algorithm 6: CLEVER algorithm pseudo-code

process continues as long as a better clustering with respect to $q(X)$ is found. The algorithm begins its search from a randomly created set of $k^{'}$ representatives, where $k^{'}$ is an input parameter of the algorithm. CLEVER has recently been generalized to cluster complex spatial objects, such as lines and polygons.

To give an example let us assume we cluster a dataset $O = \{o_1, \ldots, o_{200}\}$ with $k^{'}$ set to 3. In this case, the algorithm starts with a random representative set, let us say $\{o_3, o_9, o_{88}\}$, and forms clusters by assigning the remaining 197 objects to the closest representative which takes $O(k \times (n-k))$, where n is the number of objects in the dataset and k is current number of representatives. Next, the algorithm samples p new clusterings in the neighborhood of the current solution by inserting, deleting, or replacing representatives. For example, assuming p is 3, the algorithm might create clusterings for the representative sets $\{o_3, o_9, o_{88}, o_{92}\}$,$\{o_3, o_{88}\}$, and $\{o_3, o_{17}, o_{88}\}$, all of which have been obtained by a single insertion/deletion/replacement applied to the current representative set $\{o_3, o_9, o_{88}\}$. Next, the algorithm computes $q(X)$ for

these three clusterings, and if the best of these three clusterings improves the clustering quality, its representative set becomes the new current solution; otherwise, the algorithm terminates. In general, assuming that CLEVER runs for t iterations its complexity[3] is of the order of $O(t \cdot p \cdot k \cdot n)$ with t and k usually being much smaller than n.

5.2.3 Spatial homogeneity between neighboring clusters and within clusters

In Section 5.2.1 we stated that ideally signatures of two neighboring clusters should be significantly different from each other. In order to further discuss this issue, let us assume we have two neighboring clusters c_1 and c_2 containing a_1 and a_2 objects, respectively, which have exactly the same signature s, whose interestingness is $i(s)$. As we explained earlier, our reward framework employs a parameter $\beta > 1$ that puts a reward on the cluster size. We claim that in the discussed scenario our reward structure assigns a higher reward to a clustering which merges clusters c_1 and c_2 into a single cluster c as this clustering receives a higher reward, because of the following:

$$i(s) \times (a_1 + a_2)^\beta > i(s) \times (a_1^\beta + a_2^\beta) \tag{5.3}$$

For example, if we have two neighboring clusters with purity 90% that are dominated by objects belonging to the same class, merging these two clusters leads to a better clustering with respect to the function $q(X)$, introduced earlier. Moreover, merging

[3]The analysis of the complexity of CLEVER is further complicated by the fact that the number of representative/clusters change between iterations; that is, the algorithm might start with $k' = 100$ clusters but the final clustering might contain 83 or 113 clusters. That is, CLEVER seeks for "optimal" number of clusters with respect to the dataset O and the fitness function q.

clusters frequently leads to a drop in the interestingness/purity in the merged cluster; however, if the cluster size reward measured by $(clustersize)^{\beta}$ makes up for the loss of the interestingness with respect to the cluster signature $s(c)$, these two clusters should still be merged; therefore, the distribution of the objects belonging to a spatial cluster should be spatially homogeneous with respect to their associated signatures.

5.2.4 Determining the scope of a spatial cluster

In general, determining the scope of a spatial cluster is a challenging task. The goal is to create a spatial representation of a set of spatial objects in order to easily visualize it on the plane. One of the easiest approaches is to compute the convex hull of the spatial objects in the cluster; however, the obtained convex hull polygon is usually not very tight and frequently encloses empty spaces. This is especially the case when the spatial objects are spread out and exhibit a low spatial density. Alpha shapes [37] and the concave hull algorithm [38] generalize the convex hull algorithm, allowing for the generation of much tighter polygons which might contain holes. In our proposed method, we use the PostGIS Concave Hull algorithm [39] for computing the scope of a spatial cluster; we believe this approach is more effective than the convex hull algorithm, as it wraps a much tighter line around a set of spatial objects, resulting in less overlap with respect to the scope of neighboring clusters and less empty spaces in clusters, as can be seen in Figure 5.1.

5.3 Case study for identifying uniform regions in a city

Since understanding the evolution of a city is the key to intelligent urbanization, there is a growing need to develop urban planning and analysis tools to guide the orderly development of cities, as well as to enhance their smooth and beneficiary evolution; however, it is a big challenge for urban planners to come up with methods to analyze how cities are changing over time. Partitioning a city into uniform regions facilitates this task, as change can be analyzed based on higher levels of granularity instead on the raw data. In this section we present a set of experiments which use the method introduced in Section 5.2 to extract urban patches from a building dataset. In this context, metrics for evaluating the homogeneity of a group of buildings are very important as they impact how a city is partitioned into urban patches characterized by signatures. In particular, two such metrics, one based on purity and the other based on popular signatures, will be introduced in this section. In particular, we report the results of a series of experiments in which the CLEVER algorithm is used in conjunction with two uniformity metrics to obtain interesting, uniform regions for the city of Strasbourg, France. As part of the GeOpenSim project, a temporal topographic database of the city of Strasbourg, France has been acquired [40]. As buildings are represented as polygons, we use the Hausdorff distance [3, 4] to compute the distance between buildings in the experiments.

5.3.1 Building-type purity experiments

This section introduces a purity interestingness function which measures uniformity by the degree of the dominance of instances belonging to a single category and discusses the spatial clustering results obtained by using the purity interestingness function.

The purity interestingness function is used for analyzing the interestingness with respect to a categorical non-spatial attribute. The purity interestingness $i_{pur}(c)$ of a cluster c is computed using the following formula: Let $R = max_t p_t(c)$, $t \in cl(O)$

$$i_{pur}(c) = \begin{cases} 0, & R < th \\ (R - th)^{\eta}, & \text{otherwise} \end{cases} \tag{5.4}$$

where $cl(O)$ is the set of classes in the dataset O, $p_t()$ is a function that computes the proportions of the objects of the category t in the class c, $\eta > 0$ is the scaling factor, and $th > 0$ is the threshold. For example, assuming that $th = 0.4$, $\beta = 1$, and $s(c) = (0.6, 0, 0, 0, 0.4, 0)$ indicating that 60% of the objects belong to the first category, and 40% of the objects belong to the fourth category, we obtain: $i_{PUR}(c) = 0.6 - 0.4 = 0.2$ for the cluster c. In general when using the purity interestingness function, we are interested in obtaining clusters which are dominated by the objects of a single category.

There are six different building types in the dataset: single house, garage, commercial building, light building, collective house, and school. In year 2008, 78% of the buildings were single houses; commercial buildings were 7%; collective houses were 8%; 4% of the buildings were garages, and 3% of the buildings were light building;

finally, 1% of the buildings were schools. Building-type signatures describe the characteristics of each urban patch which can help domain experts to better understand the composition of a city.

Figure 5.1 visualizes and lists the building-type signatures of 12 clusters for the year 2008; they were generated by CLEVER using the purity interestingness function with $th = 0.5$, $\eta = 2$, and $\beta = 1.2$. Cluster 0 contains 92% commercial buildings; therefore, cluster 0 is labeled as a business urban patch. Cluster 10 is a residential area because 97% of the buildings in cluster 10 are single houses. There are 76% of collective houses in cluster 5, which indicates a living area with a lot of apartment complexes. Both garages and schools constitute very small percentages in the whole dataset, but garages and schools are more frequent in the collective housing areas in clusters 4 and 5. Surprisingly they are not present in cluster 2. Figure 5.1 verifies that our approach is able to identify contiguous urban patches dominated by buildings of a single type.

5.3.2 Using popular signatures to find uniform regions in a city

Many uniform regions are characterized by particular proportions of the class densities without having a dominating class; for example, collective houses usually have a lot of garages next to them. This is the motivation for the following alternative approach which seeks to find popular signatures which occur frequently in the contiguous subspaces of the area of interest and then uses these signatures to annotate

urban patches, as depicted in Figure 5.2.

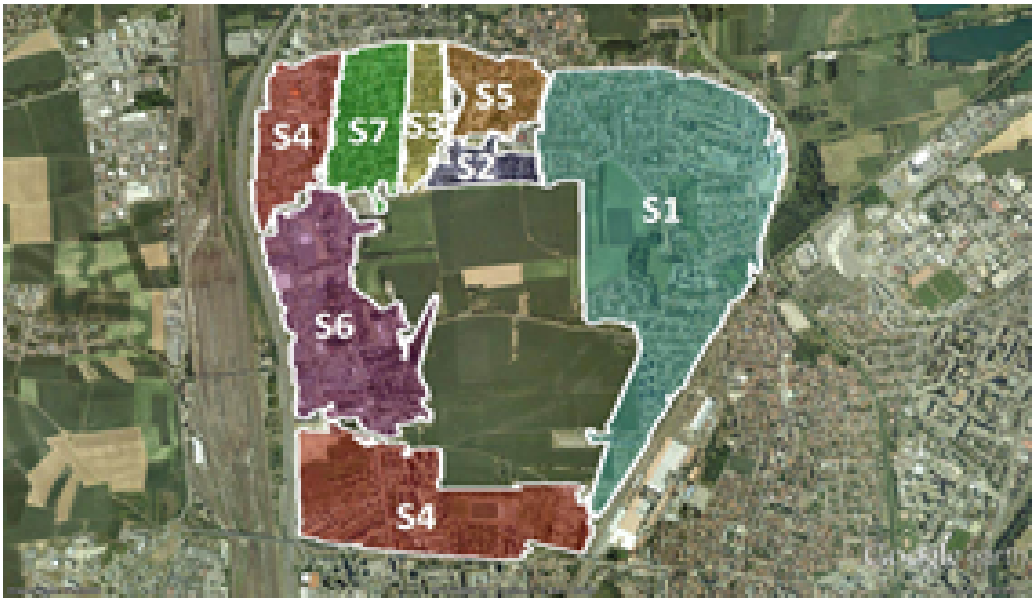

Figure 5.2: Example of a spatial clustering of buildings annotated by popular signatures

As we can see in Figure 5.2, the popular signature $S4$ is used to annotate regions in the northwest and southwest corner of the display. The challenge of generating such maps is that if we annotate a region by a popular signature, this makes sense only if the region's signature is close to the popular signature associated with it. To accomplish that, we need a spatial clustering algorithm to partition the spatial dataset into regions whose signatures are a good match with respect to a given set of popular signatures.

In the remainder of this section we will propose a framework for annotating regions with the matching popular signatures. It first collects signatures using a sampling approach; second, it identifies a set of popular signatures from the collected

signatures using a clustering approach; third, it uses a spatial clustering algorithm to identify regions with a good match with the set of popular signatures. As steps 1 and 2 are kind of straightforward, we will not discuss those further. As far as the third step is concerned, we run CLEVER using the following popular signature interestingness function $i_{POP}(c)$:

Let $cld = d(s(c), closest(s(c), P))$

$$i_{POP}(c) = \begin{cases} 0, & cld > D \\ (D - cld)^{\eta}, & \text{otherwise} \end{cases} \tag{5.5}$$

where $s(c)$ is the signature of a cluster c, P is a set of popular signatures, $closest(s(c), P)$ computes the closest signature in P to $s(c), d$ denotes the Euclidean distance, D is a match threshold, and η is a form parameter having a value in $(0, \infty)$.

Popular building-type signatures describe the compositions of urban patches which frequently occur in different parts of a city. To obtain a set of popular signatures, we first randomly created 1000 small spatial clusters and extracted their building-type signatures. Next, we applied a distance-based outlier detection technique to remove 10% of the building-type signatures as outliers - signatures were sorted by their 3-nearest neighbor distances to other signatures in the set. Signatures with the largest 3-nearest neighbor distances were removed from the signature set. Next, we clustered the remaining signature set using K-means with different k values ranging between 6 and 10 several times, and identied the clustering with the lowest squared average distance of the objects in the dataset to the cluster centroid they belong to. Finally, we extracted the centroids from the best clustering as the

popular signatures. Table 5.1 lists nine popular building-type signatures that were obtained as the result of this process.

Table 5.1: Popular building-type signatures for 2008

Signature ID	Single House	Garage	Commercial Building	Light Building	Collective House	School
$S1$	77%	3%	2%	2%	17%	0%
$S2$	87%	4%	1%	3%	4%	1%
$S3$	2%	6%	0%	0%	92%	0%
$S4$	99%	0%	0%	0%	0%	0%
$S5$	48%	1%	46%	3%	2%	0%
$S6$	4%	0%	96%	0%	0%	0%
$S7$	37%	22%	4%	1%	32%	4%
$S8$	62%	6%	13%	12%	4%	1%
$S9$	85%	1%	14%	0%	0%	0%
Dataset	78%	4%	7%	3%	8%	1%

Table 5.2 summarizes a popular signature clustering result which was created using CLEVER and the popular signature interestingness function with parameters $k' = 20$, $\beta = 1.005$, $D = 0.1$, and $\theta = 2$. We use 0.1 as the threshold for the Euclidean distance between a cluster signature and its closest popular signature to indicate a good match. 14 out of 16 urban patches have good matches with their popular signatures. Cluster 3 is quite unusual as it is dominated by light buildings and is not close to any popular signature in Table 5.1 at all, which is indicated by its very high Euclidean distance of 0.49 to its closest popular signature $S8$.

Our approach uses a spatial clustering algorithm to identify the scope of a popular signature. We claim that the urban patches identified by our approach exhibit a much better match with the popular signature set.

Table 5.2: Popular building-type clustering result for 2008

Cluster ID	Single House	Garage	Commercial Building	Light Building	Collective House	School	No. of Buildings	Closest Signature	Distance
0	89%	4%	2%	0%	5%	0%	56	$S2$	0.04
1	75%	7%	4%	0%	13%	0%	69	$S1$	0.07
2	73%	8%	6%	2%	12%	0%	52	$S1$	0.09
3	29%	2%	9%	45%	15%	0%	55	$S8$	0.49
4	72%	6%	11%	1%	10%	0%	157	$S1$	0.13
5	88%	4%	2%	3%	5%	0%	199	$S2$	0.02
6	100%	0%	0%	0%	0%	0%	112	$S4$	0.01
7	44%	1%	46%	5%	4%	0%	100	$S5$	0.05
8	87%	4%	1%	3%	3%	1%	335	$S2$	0.01
9	85%	1%	13%	1%	1%	0%	320	$S9$	0.01
10	77%	5%	8%	0%	10%	0%	39	$S1$	0.09
11	77%	3%	1%	1%	17%	2%	198	$S1$	0.03
12	36%	20%	3%	4%	34%	4%	142	$S7$	0.05
13	99%	1%	0%	0%	0%	0%	121	$S4$	0.01
14	98%	2%	0%	0%	0%	0%	57	$S4$	0.02
15	89%	0%	0%	0%	11%	0%	27	$S2$	0.09

5.3.3 Querying a spatial dataset with signatures

Although the presented popular signature mining algorithm has been originally developed to determine the scope of a set of popular signatures, it can be used in conjunction with any signature set P. This enables us to use the same algorithm for querying spatial datasets for the presence of particular query signatures. For example, in the experiment summarized in Table 5.2, we came across cluster 3, which was dominated by light buildings and it might be interesting to see if its signature $Q_1 = (29\%, 2\%, 9\%, 45\%, 15\%, 0\%)$ occurs in other areas of the city; along the same line we might want to see, if there are regions with a high density of schools in residential areas captured by the signature $Q_2 = (70\%, 0\%, 0\%, 0\%, 0\%, 30\%)$. Finally, we might like to see if the popular signature $Q_3 = (2\%, 6\%, 0\%, 0\%, 92\%, 0\%)$ (named $S3$ in Table 5.1) occurs anywhere in the dataset, as it did not match any cluster signature.

We run CLEVER using the popular signature interestingness function for the signature set $P = \{Q1, Q2, Q3\}$ with parameters $D = 0.1$, $\eta = 3$, and $\beta = 1.2$. The spatial clusters shown in Figure 5.2 are annotated with the corresponding signatures if the distance between the cluster signature and its closest query signature in P is 0.1 or less. Table 5.3 lists the signatures of these three clusters that are close to the query signatures as well as the closest query signatures and their distances to the closest query signatures.

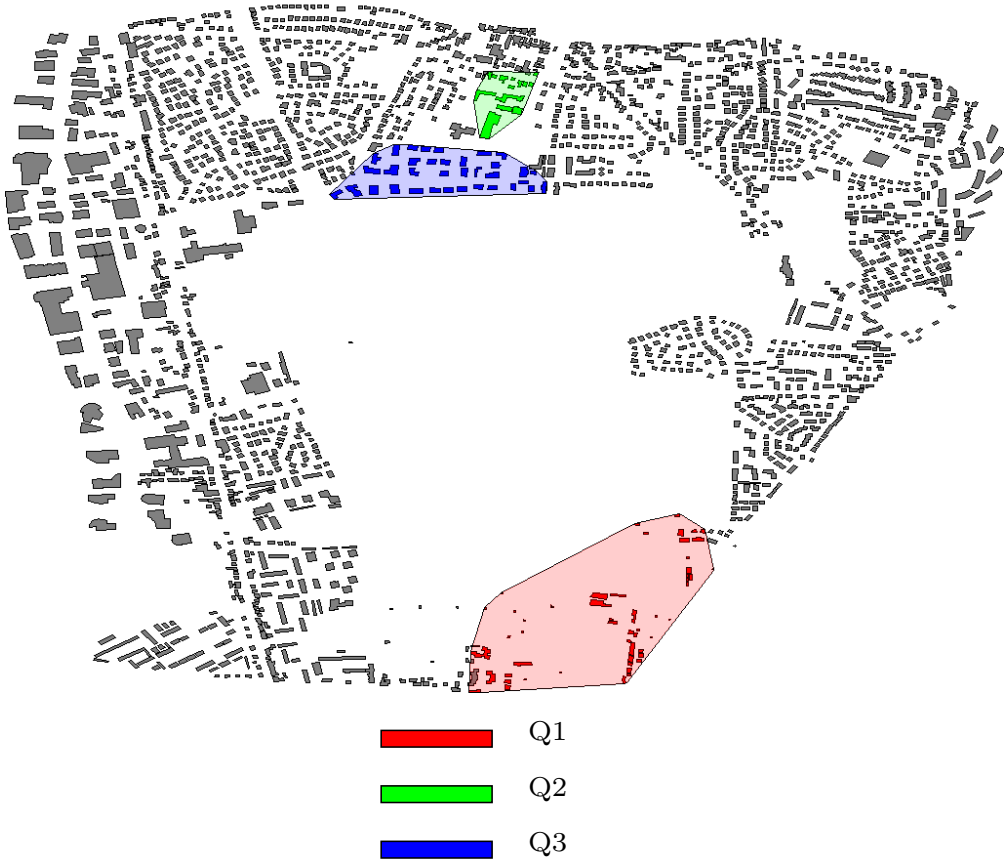

Figure 5.3: Visualization of clusters matching query signatures

Table 5.3: Clusters matching query signatures

Cluster ID	Matched Signature	Single House	Garage	Commercial Building	Light Building	Collective House	School	Distance
5	$Q1$	29.63%	1.85%	9.26%	44.44%	14.81%	0%	0.009
11	$Q3$	2.78%	5.56%	0%	0%	91.67%	0%	0.010
13	$Q2$	66.67%	0%	0%	0%	0%	33.33%	0.047

As can be seen, the algorithm rediscovered the same region (cluster 5) with a majority of light buildings identified by the popular signature clustering algorithm but no other regions which match the query signature $Q1$. Moreover, a single region (cluster 11) which almost perfectly matches the popular signature $Q3$ was found. Finally, we were able to find a single region (cluster 13) with a mixture of schools and single houses, but the match of its signature with the query signature $Q2$ is of medium quality, as the Euclidean distance between the two signatures is about 0.047.

5.3.4 Sensitivity analysis

CLEVER has been designed to find a "good" solution for what is, in general, an NP-hard problem relying on randomized hill climbing. As all optimization procedures that start with randomly created initial solutions, CLEVER - as K-means - is sensitive to initialization, as different initializations may lead to different, alternative solutions. In this section we discuss the results of two experiments which analyze CLEVER's sensitivity to initialization, and how close CLEVER gets to the "optimal" solution.

To analyze CLEVER's sensitivity to initialization, we ran the building-type purity clustering procedure 20 times with parameters $k' = 20$, $\beta = 1.05$, $\eta = 3$, and $th = 0.5$,

and collected the following run characteristics: $q(X)$, number of the clusters in the final clustering, number of iterations, and the number of clusterings generated during the run. The sampling procedure used in this (and the next) experiment is as follows: first sample 15 clusterings in the neighborhood of the current clustering, then - if there is no improvement - 30 solutions, and finally 180 solutions; if none of the 225 sampled clusterings improves the current clustering, the search ends. According to the results reported in Table 5.4, CLEVER terminated after at an average 32 iterations and searched on average 1400 clusterings. Although CLEVER starts from different initial clusterings, the quality of the clustering results is relatively stable around 729 with a standard deviation of 24; however, the number of the obtained final clusters differs quite significantly between the twenty runs, ranging between 3 and 23. This fact indicates that the obtained 20 final clusterings - although having a similar quality with respect to $q(X)$ - differ from each other significantly.

5.3.5 Performance analysis for CLEVER

Table 5.5 gives some performance characteristics for the clustering results that were reported in Section 5.3 in terms of the iterations needed, the number of clusterings generated, and the wall clock time. CLEVER was run on a dataset containing 2039 objects on a computer with the processor running at 3 GHz and 8 GB main memory.

Table 5.4: Building-type purity sensitivity results

Run ID	$q(X)$	No. of Clusters	No. of Iterations	Generated Clusterings
1	776.81	7	38	1635
2	764.68	8	43	1920
3	756.20	10	25	645
4	747.56	11	39	1830
5	746.39	12	29	1245
6	744.51	9	30	1470
7	741.23	11	24	1170
8	738.21	3	31	1470
9	737.03	13	29	1245
10	736.27	16	45	1950
11	727.90	11	39	2010
12	726.31	8	48	2175
13	719.12	10	23	960
14	716.62	23	36	1395
15	715.18	14	20	525
16	710.86	16	26	1380
17	707.44	9	31	1140
18	693.47	18	37	1605
19	688.78	16	31	1665
20	685.85	16	24	1005
Mean	729.02	12.05	32.40	1422
STD	24.63	4.55	7.88	444.40
Max	776.81	23.00	48.00	2175.00
Min	685.85	3.00	20.00	525.00

Table 5.5: Performance characteristics of the reported clustering results

	No. of Iterations	No. of Clusterings	Time Elapsed
Section 5.3.1	30	1485	32.92 S
Section 5.3.2	35	1590	33.65 S
Section 5.3.3	44	2670	38.26 S
Section 5.3.4	34	1422	31.15 S

5.4 Conclusion

This chapter introduced a spatial clustering algorithm to identify contiguous regions in the space of the spatial attributes which are uniform with respect to their signatures, which represent statistical summaries for the objects belonging to a particular cluster. The second idea advocated in the chapter is to mine spatial data for the presence of particular signatures. We claim that these two types of signature-based spatial clustering have broad applications in urban computing.

The proposed method defines the task of finding uniform regions formally as a maximization problem. Various objective functions and corresponding algorithms were introduced. In particular, we introduced a prototype-based clustering algorithm named CLEVER, which identies uniform regions in a spatial dataset by maximizing a plug-in measure of uniformity, relying on a randomized hill climbing approach. Moreover, polygon models which capture the scope of a spatial cluster and histogram-style distribution signatures were used to annotate the content of a spatial cluster; both play a key role in summarizing the composition of a spatial dataset. We claim that the presented approach is novel and unique as existing clustering algorithms are not suitable for this task as they minimize distance-based objective functions, whereas assessing uniformity relies on non-distance based uniformity measures. The efficacy of the proposed method was demonstrated by a challenging real-world case study centering on analyzing the composition of the city of Strasbourg in France based on building characteristics.

Chapter 6

Conclusion

In this research we have addressed the problem of spatial and spatio-temporal clustering and post-processing, which are important tasks in spatial data mining. Specifically, we have focused on clustering polygons, analyzing and creating summaries for clusters of polygons. Our research is motivated by the fact that polygons can represent many types of spatial objects, such as buildings, pollution hotspots, and counties. The goal of our research is to produce spatially compact and conceptually coherent clusters of polygons, and to provide potentially useful information and summarized knowledge for domain experts using our post-processing analysis techniques.

6.1 Summary of significant research contributions

Specific contributions of this research in the area of polygon-based spatial data mining are listed below.

- Distance of spatial polygons: We have developed two distance functions, i.e., the Overlay distance function and the Hybrid distance function to measure the distance between a pair of polygons, especially overlapping polygons.

- Density-based spatial clustering: We have proposed a density-based spatial clustering algorithm for polygons known as Poly-SNN that extends the density-based concepts of the shared nearest neighbor algorithm from points to polygons.

- Density-based spatio-temporal clustering: We have introduced two spatio-temporal clustering algorithms for polygons by taking into account both spatial and temporal domains of polygons.

- Post-processing analysis: We have developed several post-processing analysis techniques to further analyze the identified spatial clusters of polygons and spatio-temporal clusters of polygons, to evaluate the clusters, and to create summarizes.

- Change analysis: We have proposed an algorithm to discover the change patterns within spatio-temporal clusters of polygons.

- Identify uniform regions from spatial datasets: A formal definition of the task of finding uniform regions from spatial datasets is given. We also investigate an algorithm to identify such uniform regions from spatial datasets.

6.2 Directions for future research

This research can be extended in many different directions for future research, including:

- Automatic procedures for the parameter selection for the spatial and spatio-temporal clustering algorithms: The shared nearest neighbor based spatial and spatio-temporal clustering algorithms require several user-defined parameters that have significant impact on clustering results. These user-defined parameters need to be changed and adapted according to the datasets being clustered and the desired granularity of cluster results. Automatic procedures for optimal parameter selection need to be investigated for this task.

- Constraint-based clustering algorithm: The constraint-based spatial and spatio-temporal clustering algorithms need to be developed to take into consideration physical obstacles and facilitators that may be present when clustering spatial and spatio-temporal data.

- New algorithms for obtaining a set of popular signatures from spatial datasets: Currently we use sampling and k-means to obtain a set of popular signatures of spatial data. Such popular signatures lose the spatial meaning since they are aligned with cluster centroids. New algorithms need to be investigated to address this limitation.

Bibliography

[1] D. Joshi, A. Samal, and L. Soh, "A dissimilarity function for clustering geospatial polygons," in *Proc. the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*, Seattle, Washington, USA, November 2009.

[2] K. Buchin, M. Buchin, and W. C., "Computing the frchet distance between simple polygons in polynomial time," in *Proc. the 22nd ACM Symposium on Computational Geometry*, Sedona, Arizona, USA, June 2006.

[3] M. Atallah, C. Ribeiro, and S. Lifschitz, "Computing some distance functions between polygons," *Pattern Recognition*, vol. (24)8, pp. 775–781, 1991.

[4] J. Hangouet, "Computing of the hausdorff distance between plane vector polylines," in *Proc. the 8th International Symposium on Computer-Assisted Cartography*, Charlotte, North Carolina, USA, February 1995.

[5] J. Han, M. Kamber, and A. Tung, "Spatial clustering methods in data mining: a survey," *Geographic Data Mining and Knowledge Discovery*.

[6] D. Joshi, A. Samal, and L. Soh, "Density-based clustering of polygons," in *Proc. the IEEE Symposium on Computational Intelligence and Data Mining*, Nashville, TN, USA, April 2009.

[7] M. Kulldorff, "A spatial scan statistic," *Communications in Statistics: Theory and Methods*, vol. 26, pp. 1481–1496, 1997.

[8] S. Iyengar, "On detecting space-time clusters," in *Proc. the 10th ACM SIGMOD International Conference on Knowledge Discovery and Data Mining*, Seattle, Washington, USA, August 2004.

[9] M. Wang, A. Wang, and A. Li, "Mining spatial-temporal clusters from geodatabases," *Lecture Notes in Computer Science*, vol. 4093, pp. 263–270, 2006.

[10] D. Birant and A. Kut, "St-dbscan: An algorithm for clustering spatial-temporal data," *Data and Knowledge Engineering*, vol. 60, pp. 208–221, 2007.

[11] S. Rinzivillo, D. Pedreschi, M. Nanni, F. Giannotti, N. Andrienko, and G. Andrienko, "Visually driven analysis of movement data by progressive clustering," *Information Visualization*, vol. 7, pp. 225–239, 2008.

[12] Y. Li, J. Han, and J. Yang, "Clustering moving objects," in *Proc. the 10th ACM SIGMOD International Conference on Knowledge Discovery and Data Mining*, Seattle, Washington, USA, August 2004.

[13] D. Joshi, A. Samal, and L. Soh, "Spatio-temporal polygonal clustering with space and time as first-class citizens," *GeoInformatica*, vol. 17, pp. 387–412, 2013. [Online]. Available: Doi:10.1007/s10707-012-0157-8

[14] W. Ding, R. Jiamthapthaksin, R. Parmar, D. Jiang, T. Stepinski, and C. F. Eick, "Towards region discovery in spatial datasets," in *Proc. the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD)*, September 2008, pp. 88–99.

[15] C. Eick, B. Vaezian, D. Jiang, and J. Wang, "Discovery of interesting regions in spatial datasets using supervised clustering," in *Proc. the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, September 2006.

[16] O. Celepcikay and C. Eick, "A regional regression framework for geo-referenced datasets," in *Proc. the 17th ACM SIGSPATIAL International Conference on Advances in GIS (GIS)*, November 2009, pp. 326–335.

[17] S. Vucetic and Z. Obradovic, "Discovering homogeneous regions in spatial data through competition," in *Proc. the ICML Conference*, 2000, pp. 1095–1102.

[18] C. Sheng, Y. Zheng, W. Hsu, M. L. Lee, and X. Xie, "Answering top-k similar region queries, database systems for advanced applications," *Lecture Notes in Computer Science*, vol. 5981, pp. 186–201, 2010.

[19] D. Applegate, T. Dasu, S. Krishnan, and S. Urbanek, "Unsupervised clustering of multidimensional distributions using earth mover distance," in *Proc. the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, August 2011.

[20] C. Cortes and D. Pregibon, "Signature-based methods for data streams," *Data Mining and Knowledge Discovery*, vol. 5, pp. 167–182, 2001.

[21] P. Wong, H. Foote, R. Leung, D. Adams, and J. Thomas, "Data signatures and visualization of scientic data sets," in *Proc. the 8th Pacific Conference on Computer Graphics and Applications*, Hong Kong, China, October 2000.

[22] Z. Yin, L. Cao, J. Han, C. Zhai, and T. Huang, "Geographical topic discovery and comparison," in *Proc. the WWW Conference*, Hyderabad, India, 2011.

[23] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *Proc. the KDD Conference*, Beijing, China, 2012.

[24] L. Ertoz, M. Steinback, and V. Kumar, "Finding clusters of different sizes, shapes, and density in noisy high dimensional data," in *Proc. the 3rd SIAM International Conference on Data Mining*, San Francisco, CA, USA, May 2003.

[25] "Texas commission on environmental quality," [Online; accessed 20-Jun-2011]. [Online]. Available: http://www.tceq.state.tx.us

[26] N. Cressie, *Statistics for Spatial Data*. Wiley, 1993.

[27] C. Chen, V. Rinsurongkawong, C. Eick, and M. Twa, "Change analysis in spatial data by combining contouring algorithms with supervised density functions," in *Proc. the 13th Asia-Pacific Conference on Knowledge Discovery and Data Mining*, Bangkok, Thailand, April 2009.

[28] S. Wang, C. Chen, V. Rinsurongkawong, F. Akdag, and C. Eick, "A polygon-based methodology for mining related spatial datasets," in *Proc. the ACM SIGSPATIAL International Workshop on Data Mining for Geoinformatics*, ser. DMG '10. New York, NY, USA: ACM, 2010, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/1869890.1869891

[29] S. Wang and C. Eick, "A polygon-based clustering and analysis framework for mining spatial datasets," *GeoInformatica*, 2009.

[30] R. Lu and R. Turco, "Air pollutant transport in a coastal environment. part i: Two-dimensional simulations of sea-breeze and mountain effects," *Journal of the atmospheric sciences*, vol. 51, pp. 2285–2308, 1994.

[31] "The associated press," February 2008.

[32] Y. Zheng and X. Zhou, "Computing with spatial trajectories," *Springer-Verlag New York Inc.*, 2011.

[33] L. Windham, M. Laska, and J. Wollenberg, "Evaluating urban wetland restorations: Case studies for assessing connectivity and function," *URBAN HABITATS*, vol. 2, pp. 130–146, 2004.

[34] C. Chen, N. Shaikh, P. Charoenrattanaruk, C. F. Eick, N. Rizk, and E. Gabriel, "Design and evaluation of a parallel execution framework for the clever clustering algorithm," in *Proc. the International Conference on Parallel Computing*, Ghent, Belgium, August 2011.

[35] C. Eick, R. Parmar, W. Ding, T. Stepinki, and J. Nicot, "Finding regional colocation patterns for sets of continuous variables in spatial datasets," in *Proc. the 16th ACM SIGSPATIAL International Conference on Advances in GIS (GIS)*, Irvine, CA, USA, November 2008.

[36] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, 1990.

[37] H. Edelsbrunner, D. Kirkpartick, and R. Seidel, "On the shape of points in the plan," in *IEEE Transactions on Information Theory*, 1983, pp. 551–559.

[38] A. Moreira and M. Santos, "Concave hull: a k-nearest neighbors approach for the computation of the region occupied by a set of points," in *Proc. the International Conference on Computer Graphics Theory and Applications (GRAPP)*, 2007, pp. 61–68.

[39] "Boston geographic information systems," [Online; accessed 13-May-2012]. [Online]. Available: http://www.bostongis.com/postgis_concavehull.snippet

[40] A. Ruas, J. Perret, F. Curie, A. Mas, A. Puissant, G. Skupinski, D. Badariotti, C. Weber, P. Gancarski, N. Lachiche, A. Braud, and J. Lesbegueries, "Conception of a gis platform to study and simulate urban densification based on the analysis of topographic data," *Cartography and GIScience*, vol. 1, pp. 413–430, 2011.