Master's Degree in Computer Science
Dissertation/Internship
Final Report

A Constraint-Based Clustering Algorithm for Detection of Meaningful Places

Supervisors:
Prof. Carlos Lisboa Bento, PhD
Prof. Stefan van der Spek, PhD
Date: 1 July, 2014

Frederico José Neto Marques
fmarques@student.dei.uc.pt

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

GPS devices generate a large amount of trajectory data. However these data do not contain the user-level notion of "place". A location is a pair of coordinates without any significance to the user whereas a place represents a meaningful location, such as “home”, “work”, based on the observation of routines and their embedded semantic meaning. One of the available mechanisms to extract knowledge from these data is through the application of clustering techniques.

Clustering is a process to group objects based on their similarity, which in our case will allow us to detect intentional stops. Detecting intentional stops allows us to understand where the user spends most of his time, and, thus, to model mobility patterns. Recent clustering algorithms integrate both trajectory sample points and background geographic information. The main drawbacks of the existing approaches are: the user has to specify which physical spaces (places) he considers relevant to its trajectories; and geographic information is used to constrain the clustering algorithm and not to create a physical representation of a place.

Location-based Social Networks (LBSN), like Foursquare and Twitter, support hundreds of millions of user-driven footprints. Those global-scale footprints provide a unique opportunity to model human activity - understand how social aspects can affect human mobility patterns - and geographical areas by means of place categories.

The aim of our proposal is the creation of a robust spatio-temporal, i.e. density and time based, clustering algorithm for discovering intentional stops from the trajectories of users, in presence of noisy data. We also incorporate background geographic information - enriched with semantic labels gathered from Foursquare - to create a physical representation for the discovered intentional stops. Finally, we characterize aggregate activity patterns by finding the distributions of different activity categories over a city geography and study how social aspects can affect human mobility patterns.

Keywords

“Clustering Algorithms”, “Geographic Constraints”, “GPS trajectories”, “Human Mobility”, “Land Use”, “Places”, “Social Networks”

Table of Contents

Index of Figures

Index of Tables

1 Introduction

Devices fitted with location-based services, such as GPS or mobile devices are becoming ubiquitous and therefore enable us to collect huge quantity of positioning data representing people's movements. These devices supply the user's location as a pair of coordinates (latitude & longitude) without any semantic notion, which is not human-readable. However, many different applications such as city planning or determining social interactions require extracted knowledge from trajectories data to understand human mobility patterns. This allows labelling locations, e.g. "work", "home", which represent a meaningful place, i.e. stop, to the user. Many efforts have been directed to the learning of places, i.e. determining physical locations with significance (POIs) in our life, from users routines [1] [2].

For this particular problem one of the possible approaches is the application of clustering techniques on user's trajectories to find their intentional stops [3]. Clustering is a process to group objects based on their similarity and, as reported in surveys [4] [5] [6], on data clustering. Clustering algorithms can be roughly divided into: Partitioning Methods [7] [8] [9] [10], Hierarchical Methods [11] [12] [13], Density-Based Methods [14] [15] [16] [17] [18] and Grid-Based Methods [19] [20]. Recently [3] based on [21] proposed an algorithm that extracts stops, i.e. a semantically important part of a trajectory, and moves from trajectories. In [1], the authors proposed a time-based approach to describe a significant location as a place where user stays longer than a specific time threshold. However, the significant locations discovered by these algorithms are represented by a geographical point or a point plus radius. Also, almost none of them have taken into account constraints on the clustering process. To fill this gap, emerged clustering algorithms [22] [23] [24] [25] that integrate both trajectory sample points and background geographic information as obstacles on the clustering process. COD-CLARANS [22] defines obstacles by building visibility graphs to find the shortest distance among data objects in the presence of obstacles. AUTOCLUST+ [23] builds a Delaunay structure to cluster data points considering obstacles. Structures used by these algorithms are very expensive to build in terms of performance. Thus, to improve previous algorithms, Zaïane proposed DBCluC [24], which is an algorithm that models constraints using simple polygons. The authors created a method for reducing the edges of polygons representing obstacles by identifying a minimum set of line segments, called obstruction lines. However these existing approaches still have some drawbacks: the user has to specify which physical spaces he considers relevant to its trajectories; geographic information is used to constrain the clustering algorithm and not to create a physical representation of a place.

Due to the pervasiveness of cell phones and the popularity of mobile social media a variety of works that focus on characterizing urban environments exploiting geo-tagged information have emerged. Through location-based services in social media applications of smartphones such as Foursquare and Twitter people can share their activity related choices (check-ins) providing unprecedented amount of user-generated data on human movement and activity participation - connection between virtual social life and real-world activities. This data contains detailed geo-location information, which reflects extensive knowledge about human movement behaviour [26]. In addition, the venue category information for each check-in is recorded from which user activities can be inferred. Focusing on Twitter, [27] and [28] have used geo-tagged Twitter datasets and its semantic content to study and characterize crowd mobility. Similarly, in [29], the authors used geo-located tweets, together with their content, to create geographic language models at varying levels of granularity (from zip codes to countries). The authors use these models to predict both the location of the tweet and the user based on linguistic content changes. In [30], the authors investigate 22 million check-ins across 220,000 users to understand human mobility patterns by analysing the spatial, temporal, social, and textual aspects associated with these footprints. Foursquare has been used by [31] to model crowd activity patterns in London and New York city using spectral clustering. In [32] the authors collected data from Foursquare to analyze user check-in dynamics, demonstrating how it reveals meaningful spatio-temporal patterns. In [26], the authors used both Foursquare and Twitter data to analyze urban human mobility and activity patterns using location-based data.

In this document we propose a robust spatio-temporal (i.e. density and time based) clustering algorithm for discovering meaningful places from the trajectories of users in presence of noisy data. We also aim to incorporate background geographic information - enriched with semantic labels gathered from Foursquare - to create a physical representation for the discovered intentional stops. We can describe the main contributions of this work as:

- The creation of an algorithm that discovers the intentional stops from user trajectories. To overcome the existing approaches the user does not have to specify which physical spaces he considers relevant;
- Use geographic information - enriched with semantic labels gathered from Foursquare - in the form of shapefiles to represent the intentional stops instead of using geographic information to constrain the clustering algorithm. Thus, we can add semantic meaning to these places;
- Process of semantic enrichment of shapefiles through Foursquare;
- Characterization of the aggregate activity patterns by finding the distributions of different activity categories over a city geography and study how social aspects can affect human mobility patterns.

This document is organized as follows. In Chapter 2, we present a synthesis of the scientific and technological study conducted regarding the existing clustering algorithms and social networks. In Chapter 3 we present the methodology followed in this project. We describe the datasets and the process used to collect it; how we model geographic information, i.e. constraints, and how we integrate them into the clustering algorithm and finally how we assign semantic labels to the shapefiles. In Chapter 4 we present the results of our work. In Chapter 5 we present the executed tasks for the First and the Second Semesters. Finally, in Chapter 6 we report the conclusions and future work.

2 State of the Art

This chapter presents a synthesis of the scientific and technological study conducted for the project. In the following sections we will cover:

- **Clustering Algorithms –** we made a survey of related academic work of existing clustering algorithms, with and without constraints. Also, we will present an overview of these algorithms according to certain metrics;
- **Social Networks –** we made an analysis of the social networks used - **Twitter** and **Foursquare** - to gather the data. Also, a survey of related academic work is made.

2.1 Clustering Algorithms

Cluster analysis or simply clustering is the process of organizing/partitioning a collection of data (or observations or patterns), usually represented as a vector of measurements, or a point in a multidimensional space into subsets (or clusters) based on similarity [4] [5] [6]. A cluster is therefore a collection of objects, which are similar (share commons characteristics) to one another and are "dissimilar" (or unrelated) to the objects in other clusters. We can observe the steps that a typical pattern clustering activity involves in Figure 1 and an example of the clustering process in Figure 2. Cluster analysis has been addressed in many contexts to understand data and so can be applied in many research fields such as:

- Business Intelligence [33] [34] [35] – to organize customers with similar behaviour;
- Biology [36] [37] – to classify plants and animals according to their features;
- Web Search [38] [39] [40] – to document classification and discover groups of similar access patterns;
- Image Pattern Recognition [41] [42] – to discover clusters in handwritten character recognition systems;
- City Planning [43] – to identify groups of houses according to their house type or geographical location. It has also an important role in understanding human mobility patterns.

It should be noted that the increase of location-based devices and cheap storage mechanisms allowed the collection of huge amounts of data, making cluster analysis a highly active topic in data mining research. Clustering is considered a challenging research field and these algorithms should satisfy some important requirements such as:

- Scalability to large datasets;

- The ability to deal with different types of attributes;
- Discovering clusters with arbitrary shape;
- The ability to deal with noise and outliers;
- Requirements for domain knowledge to determine input parameters;
- High dimensionality;
- Interpretability and usability;

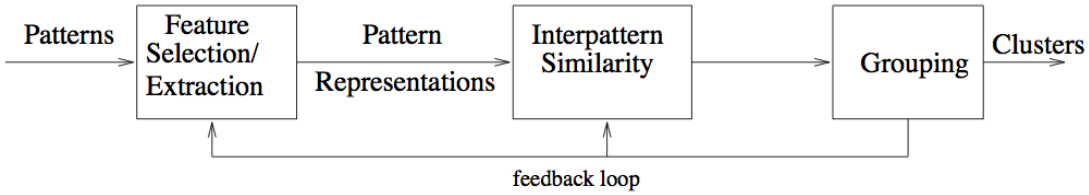


Figure 1 - Stages in clustering [44].

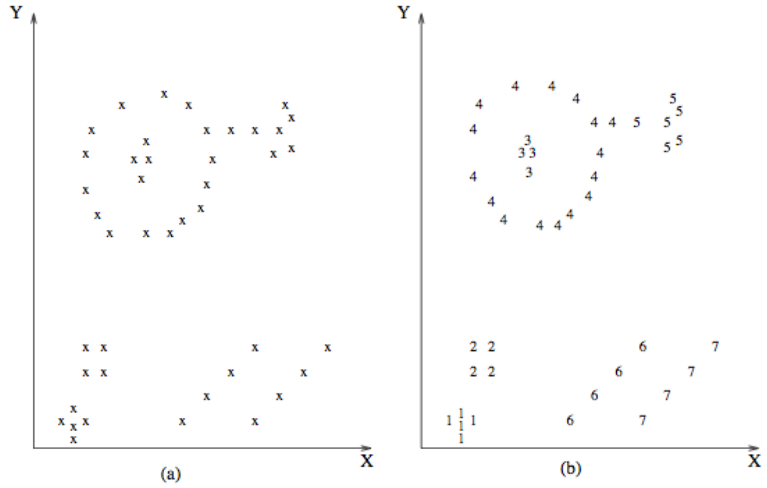


Figure 2 - Data Clustering [44].

The fact that the notion of "cluster" is not precisely defined leads to a wide variety of clustering algorithms [45] in literature. In [46], the authors suggest the division of clustering methods into two main groups: hierarchical and partitioning methods. As reported in surveys [4] [5] [6] on data clustering methods and suggested by [47] the major fundamental clustering methods can be classified into the following categories, which are discussed in this chapter:

- Partitioning Methods [7] [8] [9] [10];
- Hierarchical Methods [11] [12] [13];
- Density-Based Methods [14] [15] [16] [17] [18];
- Grid-Based Methods [19] [20];
- Time-Based Methods [1];
- Constraint-Based Methods [3] [22] [23] [24] [25];

2.2 Clustering Algorithms without Constraints

2.2.1 Partitioning Methods

Perhaps the most popular class of clustering algorithms is the Partitioning Methods. The partitioning process is basically a division of the set of data objects into several exclusive groups (or clusters) such that each data object is in exactly one subset. This method typically requires that the number of desired output clusters will be pre-set by the user and that represents a problem.

Formally, given a data set, *D*, of *n* objects, and *k*, the number of clusters to form, a partitioning algorithm organizes the objects into *k* partitions $(k \leq n)$, where each partition represents a cluster [5]. In brief the goal is to find a partition of *k* clusters that optimizes the chosen partitioning criteria.

The most well-known and commonly used partitioning methods are: K-Means and K-Medoids.

K-Means

This algorithm partitions the data into *K* clusters ($C_1, \ldots, C_k$)represented by its centroid. The centroid can be defined by the mean (usually weighted average) of all the instances belonging to that cluster. Next it is presented a description of this algorithm:

- It is chosen a number *K*, where *K* is a user-specified parameter (random). *K* represents the number of desired clusters;
- Each point is assigned to its nearest cluster centre (centroid) according to the Euclidean distance between the two;
- Centroids are re-calculated based on the points assigned to the cluster;
- The second and third steps are repeated until centroids remain the same (convergence);

Despite the wide popularity, this algorithm has some drawbacks such as: sensitivity to initial configuration, lack of robustness, unknown number of clusters, empty clusters, and handling only spherical clusters.

K-Medoids

K-Medoids or PAM (Partition around Medoids) [48] unlike K-Means use medians (medoids – which is the most representative/centric point for a group of a points) of each cluster instead of mean. The K-Medoids method has two advantages comparatively to K-Means. First it is more robust in the presence of noise and outliers because a medoid is less influenced by outliers or others extreme values than a mean, second it

presents no limitations on attribute types. However, its processing is more costly in terms of performance. In order to reduce the computational load of the basic K-Medoid algorithm on large data sets, enhanced algorithms CLARA [7] and CLARANS [9] are proposed.

CLARA (Clustering LARge Applications) is a sampling-based method that instead of taking the whole data set into consideration uses a random sample of the data set (draw multiple samples), applies PAM on each sample, and gives the best clustering as the output. The most relevant difference between PAM and CLARA is that PAM searches for the best K-Medoids among a given data set, whereas CLARA searches for the best K-Medoids among the selected sample of the data set [5].

CLARANS (Clustering Large Applications based on RANdom Search) [9] draws sample of neighbors dynamically and the clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of K-Medoids. It is more efficient and scalable than both PAM and CLARA.

2.2.2 Hierarchical Methods

The idea behind a hierarchical algorithm is the creation of a hierarchical decomposition of the given set *D* of *n* data objects, in other words, a tree of clusters (several levels of nested partitionings), also known as a dendrogram. Hierarchical methods are categorized according to two different approaches, based on how the hierarchical decomposition is formed:

- **Agglomerative** (*bottom-up*) – this approach starts with each object forming a separate group and recursively merges two or more most appropriate (close to one another) clusters or groups. The process continues until a stopping criterion: all the groups are merged into one or a termination condition holds;
- **Divisive** (*top-down*) - starts with one cluster with all *n* objects that belongs to data set *D* and recursively splits the most appropriate cluster into smaller clusters. The process continues until a stopping criterion: each object is in one cluster or a termination condition holds;

Most hierarchical clustering algorithms are variants of the single-link and complete-link algorithms. The difference between these two algorithms consists on the way they characterize the similarity between a pair of clusters. On one hand single-link merge the two clusters with the smallest maximum pairwise distance while complete-link merge the two clusters with the smallest minimum pairwise distance.

CURE

CURE (Clustering Using REpresentatives) is an agglomerative algorithm proposed by [11] - Figure 3. It tries to overcome the disadvantages of the centroid (geometric method) and all-points (graph method) approaches by presenting a hybrid of the two - it uses 'scattered points' as representation of the cluster's shape, which allows more precision than a standard spheroid radius. It takes special care with outliers and with label assignment stage. It also uses two devices to achieve scalability: random sampling and partitioning. Summing up, the algorithm:

- Identifies a set of well scattered points (referred to as representatives), representative of a potential cluster's shape;
- Shrunks the originally selected scattered points to the geometric centroid of the cluster by user-specified factor α;
- Merges clusters (the distance between two clusters is the distance between two of the closest representatives of each cluster) and re-calculates their representatives.

From Figure 3 we can acknowledge the agglomeration process in CURE algorithm: clusters before and after merge and shrinkage - the arrow represents the connection between the two closest representatives.

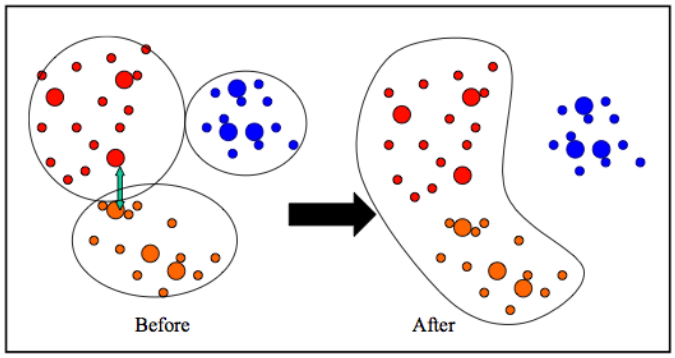


Figure 3 - Agglomeration in CURE [6].

BIRCH

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is an algorithm proposed by [13] and it was designed to handle very large data sets. This algorithm applies an incremental and dynamic clustering of incoming objects and it only analyses/scans data one time, which is sufficient to make clustering decisions. BIRCH – see Figure 4 - algorithm has two key phases:

- **First Phase** - builds a clustering feature tree (CF tree) while scanning the data set. A CF Tree is a compact storage for data on points in a cluster where each

entry represents a cluster of objects and it is characterized by a 3-tuple: (N^1, LS^2, SS^3).

- **Second Phase** – uses an arbitrary clustering algorithm to cluster the leaf nodes of CF Tree;

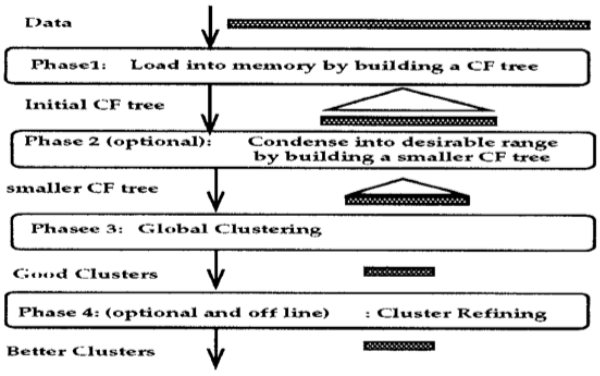


Figure 4 - BIRCH process [13].

ROCK

ROCK (RObust Clustering using linKs) is an agglomerative algorithm proposed [49] based on the notion of links as a way to measure the similarity/proximity between a pair of data points with categorical attributes. ROCK combines, from a conceptual point of view, nearest neighbor, relocation, and hierarchical agglomerative methods. In this algorithm, cluster similarity is based on the number of points from different clusters that have neighbors in common [50]. This algorithm has three key steps:

- Random Sampling – draw random sample from the data set;
- Clustering with links – cluster the data set with link technique;
- Labelling data on data set – label the remaining data on disk.

CHAMELEON

CHAMELEON is an agglomerative algorithm proposed by [12] that uses a dynamic model to obtain clusters. To model the objects of the data set it uses the commonly used *K-nearest neighbor* graph approach – see Figure 5. The use of *K-nearest neighbor* algorithm has some advantages such as: reduce noise in the data set avoiding data points that are far away and capture the concept of neighbourhood dynamically. CHAMELEON algorithm

[1] Number of data points in the cluster

[2] Linear sum of the N data points

[3] Square sum of the N data points

consists on two-phase approach - see Figure 6 provides an overview of the overall approach used by CHAMELEON to find the clusters in a data point:

- **First Phase** – uses a graph partitioning algorithm to divide the data set into a large number of individual clusters;
- **Second Phase** - uses an agglomerative hierarchical algorithm to merge these individual clusters. In this phase, CHAMELEON takes into account internal characteristics of the clusters themselves, such as inter-connectivity and closeness of the clusters, to model cluster similarity.

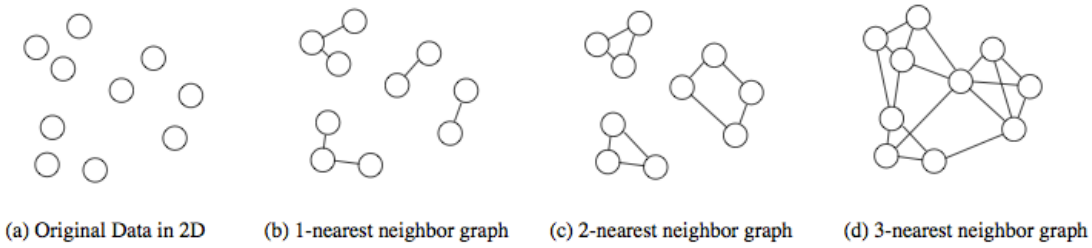


Figure 5 - Example of *K-nearest neighbour* graph approach [12].

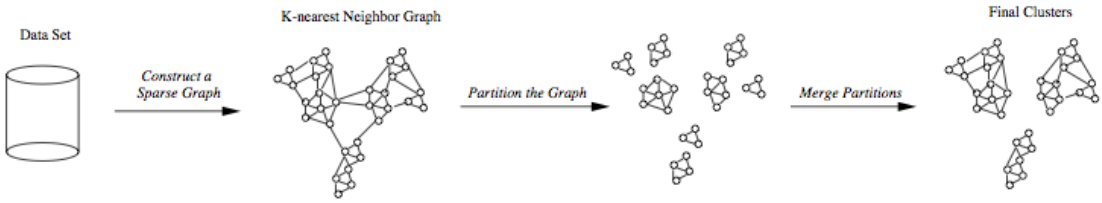


Figure 6 - CHAMELEON process [12].

2.2.3 Density-Based Methods

Until now we have only talked about methods that find spherical-shaped clusters and cluster objects based on the distance between objects. Density-Based algorithms are based on the notion of density - clusters are dense regions (high density) in the data space, separated by regions of lower object density (sparse regions) – and are very popular for the purpose of database mining. In this section it will be presented the most representative algorithms and new approaches that have emerged in recent times.

DBSCAN

DBSCAN (Density Based Spatial Clustering of Applications with Noise) is an algorithm proposed by [14] and it is designed to discover clusters with an arbitrary shape and noise in a spatial database. The basic idea for the algorithm is that for each point of a cluster, the cardinality of the neighborhood of a given radius (ε) has to exceed a threshold (*MinPts).* To better understand this, there are some notions to take into account in this algorithm:

- Parameters – there are two particular parameters that define cluster density, the ε (*Eps)* and *MinPts*. The *Eps* parameter represents the maximum *radius* of a neighbourhood and the *MinPts* parameter specifies the density threshold (minimum number of points required to form a cluster);
- *Density* - for a particular point in the data set, *density* can be measured by counting the number of points within a specified radius, *Eps*, of that point. It's important to emphasize that the density of any point will depend on specified radius;
- A point can be classified into three distinct categories – **core point** (point in the interior of a dense region and satisfies the following condition $\#Pts$[4] $(\varepsilon - neighborhood) \geq MinPts$); **border point** (point on the edge of a dense region) and **noise point** (point that are not in any dense region/sparsely occupied region);
- *Directly density-reachable* – one point *p* is directly density-reachable from a point *q* if is part of its $\varepsilon - neighborhood$;
- Density-reachable - one point *p* is density-reachable from a point *q* with respect to *Eps* and *MinPts* if there is a chain of points $p_1, \dots, p_n,\ p_1 = q, p_n = p,$ such that p_{i+1} is directly density-reachable from p_i [14];
- *Density-connected* - One point *p* is *density-connected* to a point *q* with respect to *Eps* and *MinPts* if there is a point *o* such that both, *p* and *q* are *density-reachable* from *o* with respect to *Eps* and *MinPts* [14].

OPTICS

OPTICS (Ordering Points to Identify the Clustering Structure) is an algorithm proposed by [15]. Unlike DBSCAN, OPTICS does not explicitly produce a data set clustering. Instead it computes an augmented cluster-ordering (that represents the density-based clustering structure) of the data set objects, i.e. store the order in which the objects are processed. The main advantage of this approach compared to the previous method is that it isn't limited to one global parameter setting, instead cluster-ordering contains information, which is equivalent to this parameterization. This information consists of two new concepts for each object:

- **Core-distance** – is the smallest distance threshold (ε') that makes *p* a core object;
- **Reachability-distance** – minimum *radius* that makes *p density-reachable* from a core object *q*.

[4] Number of points

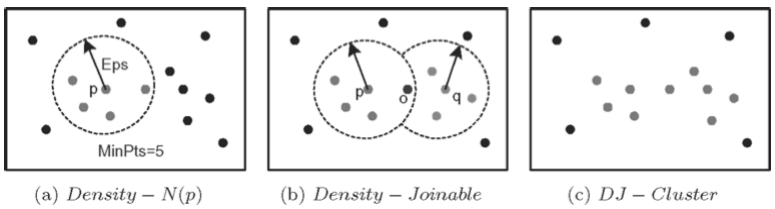
DENCLUE

DENCLUE (DENsity-based CLUstEring) is an algorithm proposed by [16] that generalizes DBSCAN, K-Means and Hierarchical Clustering. DENCLUE algorithm estimates/models the local density of the data set using a mathematical function in a very similar way to the kernel probability density function estimators. Thus, the authors introduced a new definition here: **influence function -** a function that describes the impact of a data point within its neighbourhood. This influence function is copied to each data point yielding the density function. Clusters can be determined mathematically by identifying **density-attractors** that represents the local maxima of the overall density function (i.e. sum of the influence function of all data points). This is another definition introduced by the authors. This algorithm has two key steps:

- **First Step** – is a pre-processing step. It constructs a map of the data set using hypercubes. Hypercubes contain: number of data points, pointers to data points and the sum of data values (for mean). Only populated cubes are saved in B^+ tree;
- **Second Step** – this step uses only highly populated cubes and cubes that are connected to them. Then determine density attractors for all points using hill climbing.

DJ-Cluster

DJ-Cluster (Density-and-Join-Cluster) is an algorithm proposed by [17] and it was designed to overcome the memory issues in DBSCAN. In this algorithm the authors introduce a new concept: the **Density-Joinable** definition – see Figure 7. This concept consists on: *N(p)*[5] is density-joinable to *N(q)*, denoted as $J(N(p), N(q))$, if there is a point o such that both *N(p)* and *N(q)* contain *o*. From Figure 7 we can acknowledge the density-based join concepts introduced in DJ-Cluster algorithm: (a) illustrates the density-based neighbourhood *N* of a point *p*; (b) illustrates that *N(p)* is density-joinable to *N(q)*; (c) illustrates the final cluster in grey color.



(a) *Density – N(p)* (b) *Density – Joinable* (c) *DJ – Cluster*

Figure 7 - Density-based join concepts [17].

[5] Neighbourhood of the point

P-DBSCAN

P-DBSCAN (Photo-DBSCAN) is an algorithm proposed by [18] specialized for the problem of analysis of places and events using a collection of geo-tagged photos. According to the authors, this algorithm aims to improve the existing density-based algorithms by considering that the users (owners of the photos) are those who determine the importance of a cluster, instead of considering that the points have equal "importance". The authors introduce two new concepts:

- **Density** – in this case the neighborhood density corresponds to the number of people who take photos in the area;
- **Adaptive density** – if one cluster contains areas with big differences in density basically the idea consists on splitting this cluster into smaller clusters.

2.2.4 Time-Based Methods

Time-Based Algorithm that uses Radio Frequency-Emissions

This Time-Based clustering algorithm was proposed by [1] for extracting significant places (i.e. where the user spends a considerable amount of time) from a trace of coordinates. These location coordinates are obtained by listening for RF-emissions from known access points – WI-FI positioning. On a first approach the authors used known clustering algorithms such as K-Means and GMM (Gaussian mixture model) but they reached the conclusion that clusters generated by theses algorithms contain unimportant coordinates (transitory coordinates between significant places). To overcome the drawbacks of these algorithms they proposed an algorithm – see Figure 8 (A and B represent significant places) - that only considers that a new place was found when:

- The distance from the previous place to a new location is beyond a threshold d;
- New locations span a significant time threshold t.

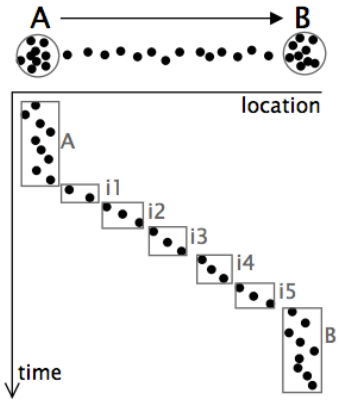


Figure 8 - Time-Based algorithm [1].

2.3 Constraint-Based Clustering Algorithms

The previous methods don't consider the use of constraints. The core idea of the following algorithms consists on introduction of background knowledge, in this specific case, GIS information, to improve clustering analysis.

2.3.1 Geographic Constraints

COD-CLARANS

COD-CLARANS (Clustering with Obstructed Distance based on CLARANS) is an algorithm proposed by [22] based mainly on CLARANS [9] [51] and it is designed for handling obstacles. The algorithm to handle obstacles – see Figure 9 (Locations with obstacles (left); Clusters formed when ignoring obstacles (right) - is called **COD** (Clustering with Obstructed Distance) and is a derivation of K-Medoids approach (instead of K-Means since the mean of a set of points it is not well defined when obstacles are involved). Basically **COD** is a change in the **distance-error function (**e.g. Euclidean distance), referred as **obstructed distance**, i.e. distance of the shortest distance between two points such that obstacles are avoided. The COD-CLARANS algorithm consists on the following phases:

- First Phase – pre-processes the data, constructing both **BSP** (Binary Space Partition) tree and **Visibility Graph**[6];
- Second Phase – The **Visibility Graph** is pre-processed in order to compute **obstructed distances** between points and **medoids**. Then it is applied a *Micro-clustering* step to reduce the complexity of the data set. Finally it is estimated the **pruning function (*E'*),** used to reduce the search space by minimizing distance (lower bound) errors when selecting cluster representatives.

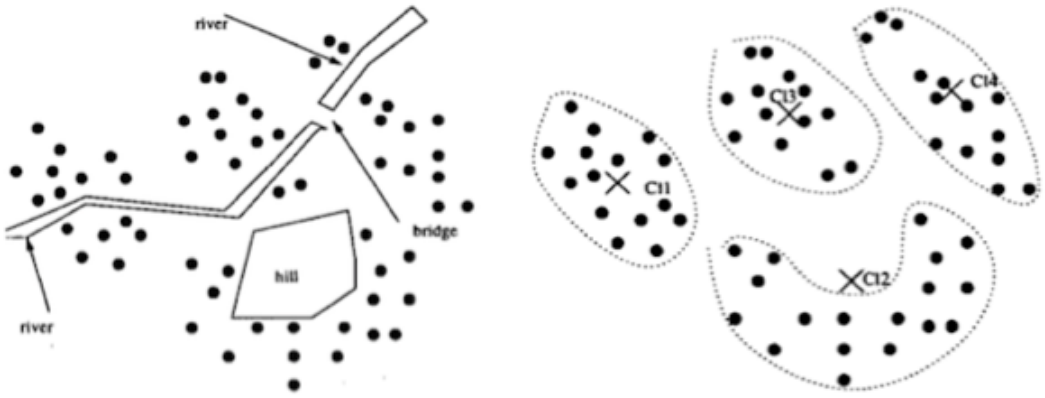


Figure 9 - COD Algorithm [22].

[6] Defines the relationship between obstacles and data objects.

AUTOCLUST+

AUTOCLUST+ is an algorithm proposed by [23] – see Figure 10 - that integrates information of data layers, particularly from various sets of obstacles (such as rivers, mountain ranges, or highways), to produce accurate clustering results. It is based on principles of AUTOCLUST [52]:

- Uses Voronoi[7] and Delaunay[8] Diagrams as the data model and structure, respectively;
- Delaunay Diagram is a data structure where all points are represented and linked by edges based on mean and standard deviations of distances between points [53];
- Closeness is represented by points linked by short edges;
- Doesn't need any user-specified parameter, which is an advantage.

Briefly, AUTOCLUST consists in three phases (each one of them is an edge correction phase) that constitutes a process to automatically find cluster boundaries – see Figure 10. AUTOCLUST+ model obstacles as a set of line segments that obstruct the edges from the Delaunay Diagram and consists in four phases:

- First Phase – construction of Delaunay Diagram;
- Second Phase – it is calculated for all points the average of the standard deviations in the length of incident edges;
- Third Phase – remove from the Delaunay Diagram the edges that intersect any obstacle. These removed edges are replaced by a detour path, i.e. shortest path between two objects;
- Fourth Phase – apply AUTOCLUST.

From Figure 10 we can recognize the tree phases of the AUTOCLUST+ algorithm: (a) Delaunay Diagram; (b) After Phase I (initial 2 clusters); (c) After Phase II (2 clusters with refinement); (d) After Phase III (3 clusters).

[7] Method for decomposing space into regions (where each region has a seed) that consists on all points closer to that seed than to any other.

[8] Corresponds to the dual graph of the Voronoi Diagram.

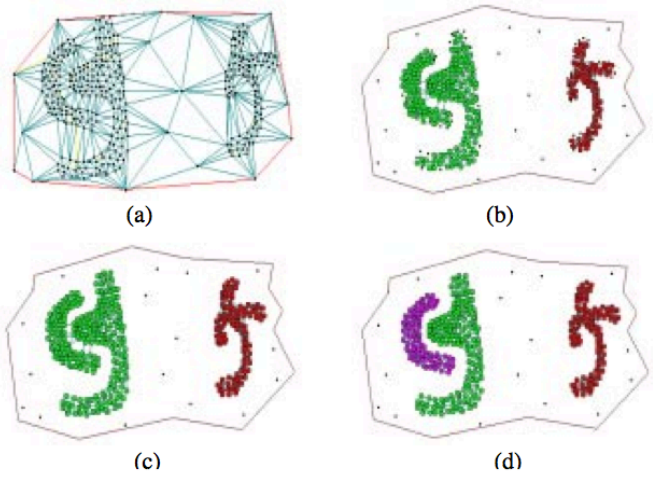


Figure 10 - AUTOCLUST three phases [23].

DBCluC

DBCluC (Density-Based Clustering with Constraints) is an algorithm proposed by [24] derived from DBSCAN, which takes into account physical constraints, i.e. polygons that represents obstacles such as rivers, highways, mountain ranges, etc. The authors proposed a method for reducing the edges of polygons to a minimum line segments, called **obstruction lines** what enhance the clustering performance. Thus, this derivation of DBSCAN takes into account visibility spaces created by the polygon edges and uses **obstruction lines** to stop the propagation of point neighborhood.

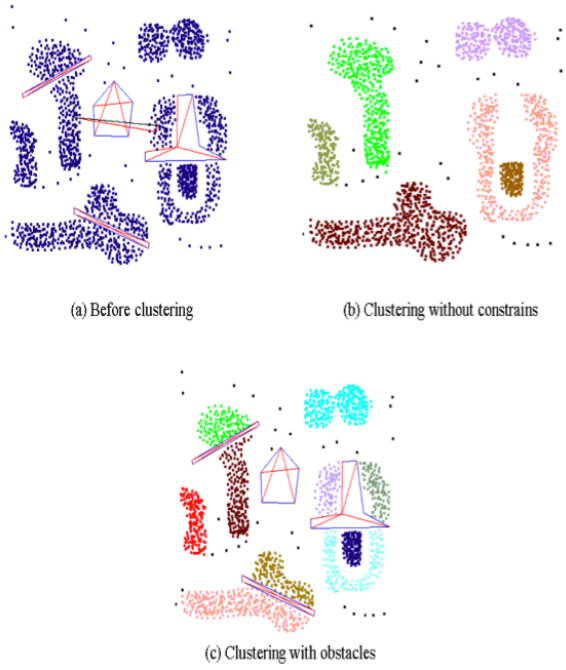


Figure 11 - DBCluC algorithm [24].

2.3.2 Geographic and Time Constraints

SMoT

SMoT (Stops and Moves of Trajectories) is a pre-processing algorithm proposed by [25] to reduce complexity associated to trajectory data analysis. This pre-processing consists on interpreting original trajectories into a set of *stops* and *moves* through the interception among user-specified relevant geographic objects (**candidate stops**, i.e. polygons) and trajectories – see Figure 12. A stop represents a meaningful place in user's trajectories where he stays for a period beyond a defined threshold. A *move* represents fragments of the trajectory outside the boundaries of the polygons and between them. The algorithm can be described as: for each point of the trajectory *T* if it intersects a polygon (**candidate stop**) and if the duration of intersection is ≥ *given threshold*, that point is categorized as a *stop*.

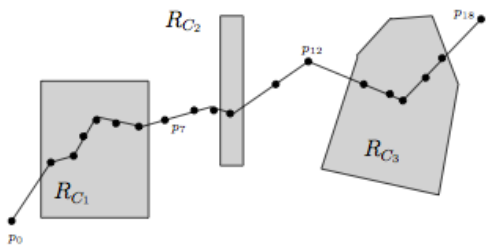


Figure 12 - SMoT algorithm [25].

CB-SMoT

CB-SMoT (Clustering-Based SMoT) is a spatio-temporal algorithm proposed by [3] based on DBSCAN and SMoT. The difference regarding previous algorithm - see Figure 13 - is the introduction of speed-based spatio-temporal clustering approach, i.e. the authors consider that the parts of the trajectory in which speed is lower than in other parts of the same trajectory, correspond to interesting places [3]. Thereunto they introduced new definitions on DBSCAN algorithm:

- ***Eps-linear-neighborhood*** - for a point *p*, ***Eps-linear-neighborhood*** is a set of points before and after *p* in the trajectory whose distance from *p* is ≥ *Eps*;
- **Neighborhood** - consider a minimal duration instead of minimal number of points;
- **Quantile Function** – used to automatically estimate *Eps* value.

Basically CB-SMoT after identifying the slower parts of a trajectory (i.e. potential stops) using the modified DBSCAN tests if they intersect geographic information for a minimal

duration. If the potential stops do not intersect any kind of geographic information it still can be an interesting place.

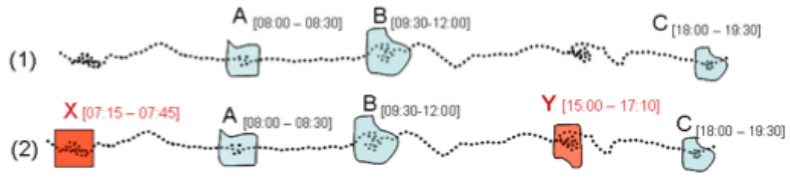


Figure 13 - SMoT algorithm (top); CB-SMoT (bottom) [3].

2.4 Comparison of the Several Algorithms

In this section is summarized the previous algorithms according with some relevant metrics to the context of this project – see Table 1. The metrics considered are:

- The ability to discover clusters with **arbitrary shapes** and not only convex or spherical clusters;
- The ability to **handle robustly with noise and outliers** avoiding degenerate results;
- Sensitive to initialization (parameters);
- The ability to handle large data sets;
- Use obstacles as constraints on the clustering algorithm.

Table 1 - Comparison of the several clustering algorithms.

Algorithms	Handle clusters with arbitrary shapes	Robust to the presence of noise/outliers	Sensitive to initialization (parameters)	Handle large data sets	Use obstacles as constraints
Partitioning Methods	✗	✗	✓	✗	✗
CURE	✓	✓	✓	✓	✗
BIRCH	✗	✓		✓	✗
ROCK		✗	✓	✓	✗
CHAMELEON	✓	✗	✓	✓	✗
DBSCAN	✓	✓	✓	✓	✗
OPTICS	✓				✗

DENCLUE	✓		✓		✗
DJ-Cluster	✓	✓	✓	✓	✗
COD-CLARANS	✗	✗	✓	✗	✓
AUTOCLUST+	✓	✓	✗	✗	✓
DBCluC	✓	✓			✓

Partitioning methods are known for not handling the discovery of clusters with an arbitrary shape, i.e. they are only suitable for concave spherical clusters. Furthermore they are very sensitive to noise and they also have difficulty in clustering data containing outliers.

Hierarchical methods will not undo what was done previously, i.e. they do not revisit once constructed (intermediate) clusters with the purpose of their improvement [5]. Thus, merge or split decisions, if not well chosen, may lead to low-quality clusters.

Density-based methods can discover clusters with arbitrary shapes and the number of clusters does not need to be specified beforehand however they are very sensitive to the parameterization.

Constraint-based methods use physical obstacles as constraints; however, taking into account these constraints during the clustering process is costly.

In conclusion, partitioning and hierarchical methods are designed to find spherical-shaped clusters. They inaccurately identify convex regions, where noise or outliers are included in the clusters. However, CURE and CHAMELEON algorithms are an exception because they can discover clusters with an arbitrary shape. To overcome the problems of the partitioning and hierarchical algorithms emerged the density-based algorithms. These algorithms, e.g. DBSCAN, OPTICS, DENCLUE and DJ-Cluster, can find clusters of arbitrary shape and model clusters as dense regions in the data space, separated by sparse regions. None of the algorithms talked about so far take into account constraints. Thus, algorithms such as COD-CLARANS, AUTOCLUST+ and DBCluC have emerged, however they are not very efficient in terms of performance. Although DBSCAN is very sensitive to parameterization it is one of the most used algorithms and presents itself as the best basis for the creation of our algorithm. The fact that none of these algorithms consider physical obstacles to represent places is the essential point for creating a new algorithm.

2.5 Social Networks

2.5.1 Twitter

Twitter is an online social network and a microblogging service that enables users to send and read "tweets," which are messages limited to 140 characters. Registered users of Twitter are able to read and post tweets via the web, SMS or mobile applications. Twitter was founded in 2006 and it is estimated that these days has a community with more than 500 million registered users. From Figure 14 we can acknowledge the Twitter growth over the years. Through Twitter users can truly get the pulse of the planet because Twitter is a reflection of what's happening in the world at any given time. As a result, Twitter's data has been coveted by both computer and social scientists to better understand human behaviour and dynamics [54].

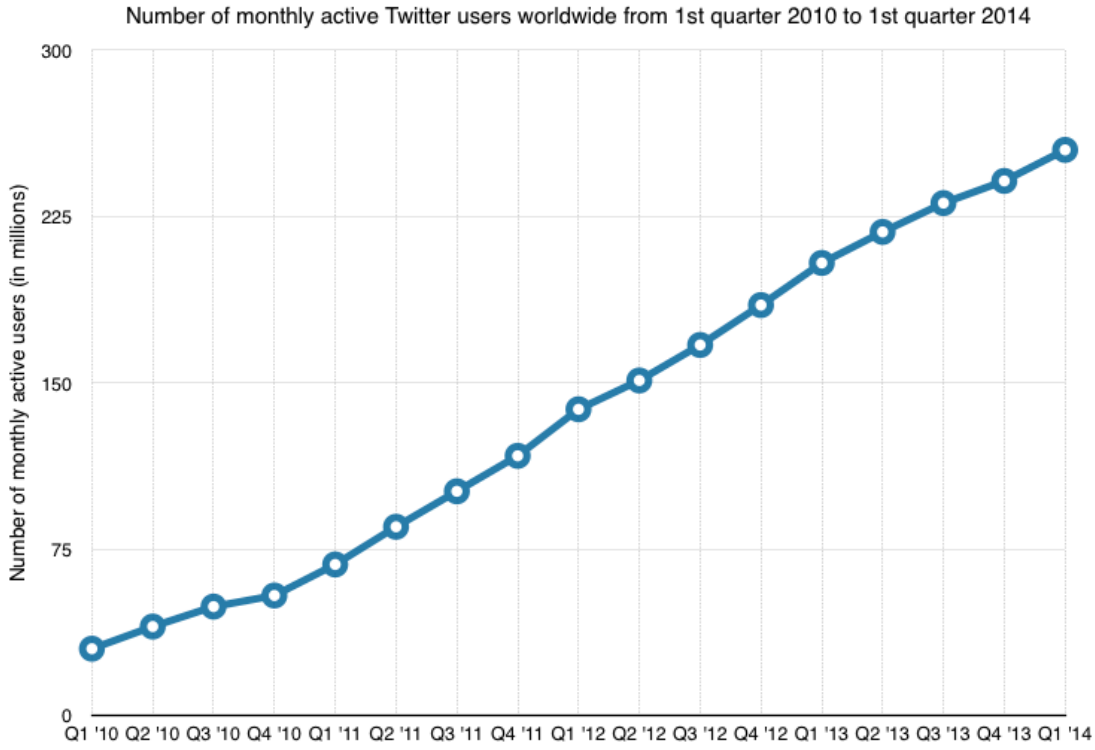


Figure 14 - Number of monthly active Twitter users worldwide [55].

Twitter API

Social media data is often difficult to obtain, with most social media sites restricting access to their data. Twitter's policies lie opposite to this. The Twitter platform consists of three major [56] set of APIs:

- **REST APIs** - perform actions on Twitter and explores the existing set of data;
- **Streaming APIs** – perform access to Tweets in real time that allows anyone to retrieve at most a 1% sample of all the data by providing some parameters;

- **Firehose** - very similar to the Twitter's Streaming API but the Twitter Firehose guarantees delivery of 100% of the tweets that match some sort of "search" criteria.

Twitter's Search API gives you access to a data set that already exists from tweets that have occurred. Through the Search API users request tweets that match some sort of "search" criteria. The criteria can be keywords, usernames, locations, named places, etc. All Twitter information is represented in the form of objects (tweets, users, entities, and places). Every object has a unique ID associated with and the access to their information is carried through this same ID. However, the Twitter Search API is limited by Twitter's rate limits. Unlike Twitter's Search API, Twitter's Streaming API is a push of data as tweets that happen in near real-time. The main drawback of this method is that provides only a sample of tweets that are occurring. To circumvent this situation Twitter introduced the Firehose that guarantees the delivery of 100% of the tweets that match some sort of "search" criteria.

2.5.2 Foursquare

Foursquare is a location-based social network founded in 2009 where users receive points and virtual badges for 'checking in' at selected venues. It is estimated that these days Foursquare has a community with more than 45 million registered users. From Figure 15 we can see the Foursquare growth over the years. Users can use the Foursquare application on their mobile devices and when at a place they can check-in, letting their friends or the world (if they edit their privacy settings accordingly) know where they are. Also we can see where your friends have checked-in, which helps you meet up with them. These check-ins can be further pushed to other social networking platforms such as Twitter and Facebook. After user has checked in, it can write reviews and tips for the location, which will be available to other Foursquare users. Those check-ins power the Foursquare Explore search engine, which provides personalized recommendations of the best places nearby. In addition, more than 1.6 million businesses from large brands to small merchants have used its advertising and merchant tools to attract and maintain customers. More than 50,000 developers use the Foursquare API to add location to their apps. It has applications for iPhone, Android, BlackBerry, Windows Phone and other smartphones.

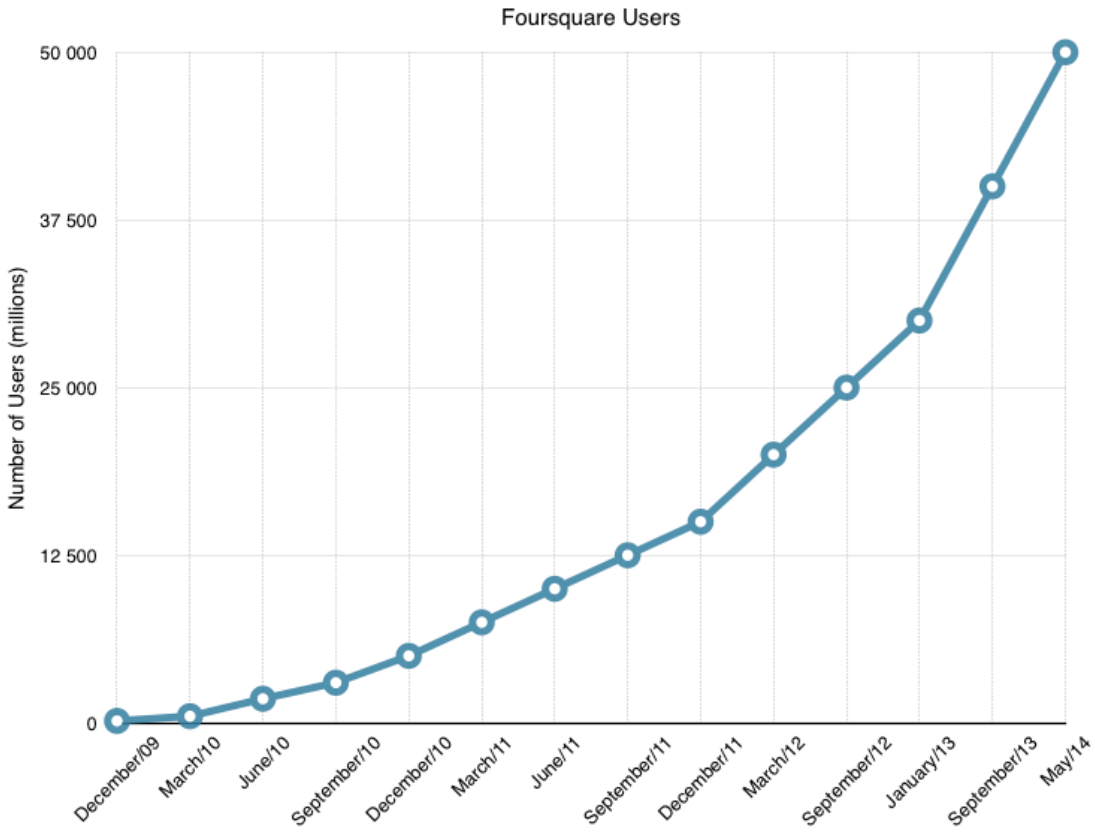


Figure 15 - Foursquare Growth chart [57].

Foursquare API

The Foursquare API allows application developers to interact with the Foursquare platform and provides methods for accessing a resource such as users, venues, check-ins, events, etc. The Foursquare platform provides four ways to access data depending on user's needs:

- **Core API** – focused for all users. Users can check in, view their history, see where their friends are, create tips and lists, search for and learn more about venues, and access specials and recommendations [58];
- **Real-Time API** – focused to notifies venue managers when users check in to their venues [58];
- **Venues Platform** – focused to allows developers to search for places and access the information about them;
- **Merchant Platform** – focused to help registered venue owners manage their foursquare presence and specials [58].

2.5.3 Related Work

Due to the pervasiveness of cell phones and the popularity of mobile social media a variety of works that focus on characterizing urban environments and human mobility patterns using mobile social networks have emerged. In [30], the authors investigate 22 million check-ins across 220,000 users to understand human mobility patterns by analysing the spatial, temporal, social, and textual aspects associated with these footprints. For the gathering process they used check-ins from different sources such as Foursquare, Twitter, Gowalla amongst others. In a first approach considering the temporal distribution of check-ins they analysed both daily and weekly patterns. In a second approach they considered three statistical properties - user displacement, radius of gyration and returning probability - to study and model human mobility patterns. In a final approach they tried to understand how factors like geography and economic status can restrict and influence human mobility patterns. In [31], the authors proposed an approach based on a spectral clustering algorithm to identify user communities that visit similar categories of places. Since Foursquare API imposes rate limits they resorted to Twitter messages which contain Foursquare check-ins to collect the data. The dataset used contains approximately 12 million check-ins. Their algorithm creates a representation - based on a grid with equally sized geographic areas - according to the categories of nearby places and the attached social activity modelled through the number of check-ins that took place at those. In [59], the authors proposed an approach based on PSMM (Periodic & Social Mobility Model) [60] for predicting user location. This approach uses two components to make the prediction: (1) the first is based on spatio-temporal information - coordinates and times of user check-ins - and uses the radiation model and (2) the second operates on social information between different users based on the frequency of matching check-ins of user's friends. The dataset used was gathered from Foursquare. In [32], the authors collected data from Foursquare to analyse user check-in dynamics, demonstrating how it reveals meaningful spatio-temporal patterns. Their assumption suggests that activity in Foursquare is influenced by a temporal and spatial point of view. Also, they described how check-ins could help to investigate user transitions from one place or activity to the next. In [26], the authors analysed urban human mobility and activity patterns using location-based data gathered from Foursquare and Twitter. They characterize aggregate activity patterns by finding the distributions of different activity categories over city geography. To locate each check-in activity, the authors constructed a virtual grid by dividing the map into square cells of size 200meters × 200meters. For each cell - ranked places - of this grid they computed the frequency of check-ins to understand which of these cells represent a popular place. They found that people do not select they destinations randomly, quite the opposite people select places based on their popularity.

3 Planning

3.1 Original Planning

The Gantt diagram - see Figure 16 - illustrates the original planning for this project. As the research of the state of the art was being conducted we found other problems that we considered most important to address. Thus, there are some deviations in the focus of the work.

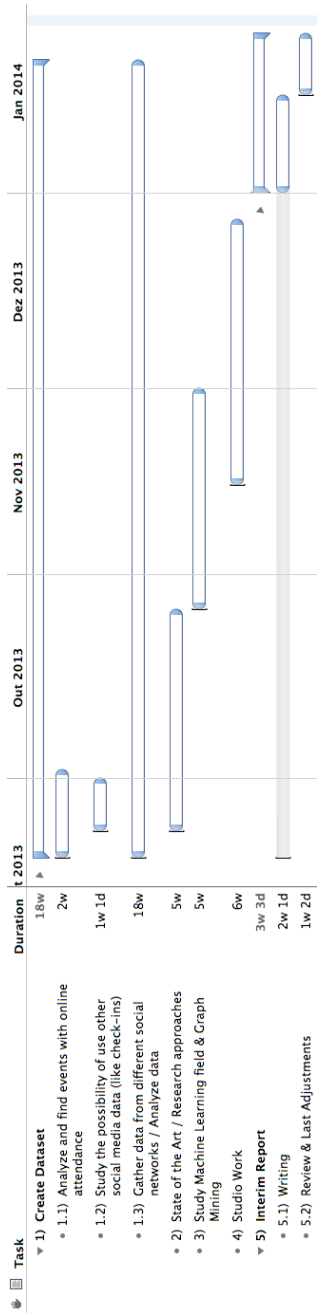


Figure 16 - Original Planning for the first Semester.

3.2 Executed Tasks

From Figure 17 we can see the executed tasks during the first and the second semesters.

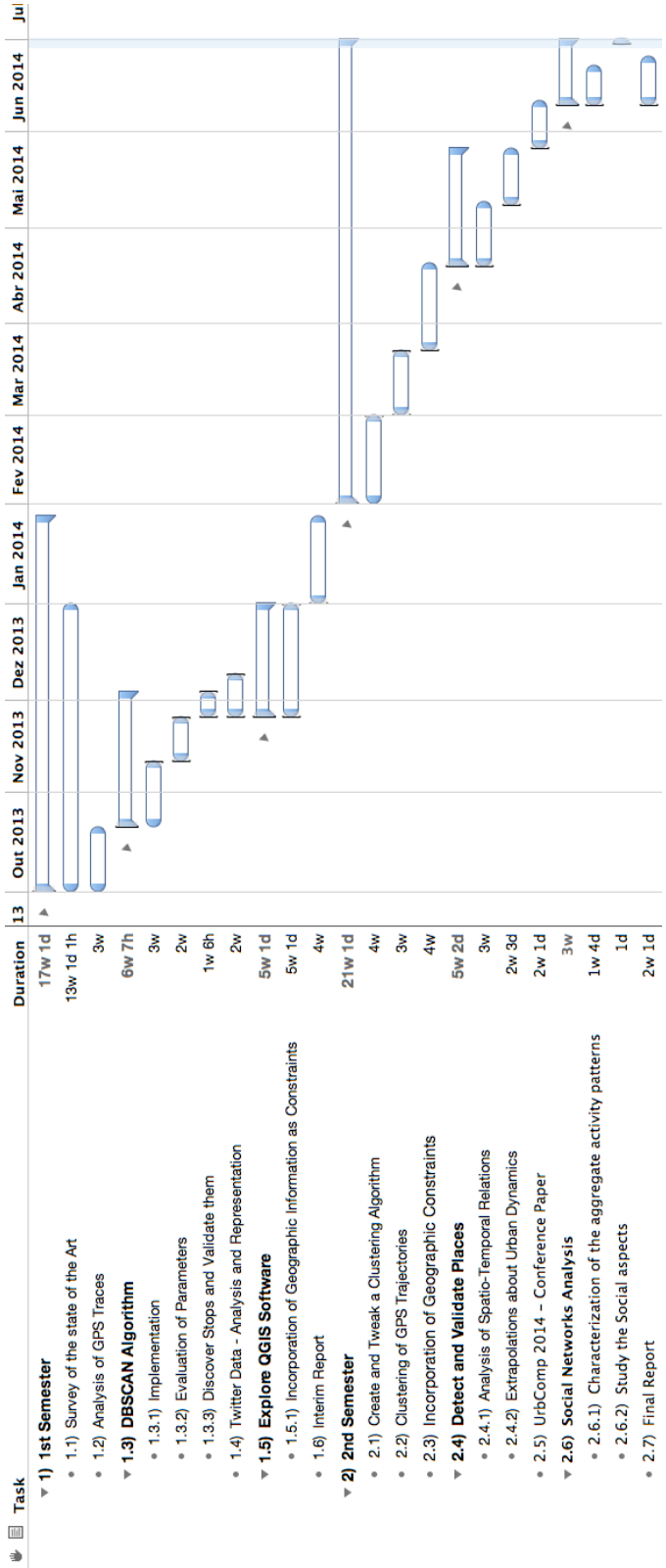


Figure 17 - First and Second semesters Gantt diagram.

4 Methodology

In the previous chapter we discussed several clustering algorithms and their major advantages and disadvantages. The purpose of this analysis was to understand which are the best candidates for discovering intentional stops in trajectories.

In this chapter we present the methodology followed on our work - Figure 18. The first step consists on GPS data collection through different applications for mobile devices. Next, we apply the first part of our algorithm, which can be subdivided into two phases:

- **First Phase** - since GPS data comes with noise and errors we created some processes to remove them;
- **Second Phase** - since DBSCAN is very sensitive to parameterization we implemented a heuristic to determine the parameters Eps and MinPts automatically from the properties of each single trajectory.

The second part of our algorithm consists on the creation of a density and time based clustering algorithm to discover intentional stops from trajectory data. After we have intentional stops we apply background geographic information - shapefiles - to assign semantic meaning to these locations. Since some of the polygons that belong to the shapefiles don't have any category (semantic label) embedded we used venues gathered from Foursquare to the semantic enrichment process. In the following sub-sections we will explain in more detail each step listed above.

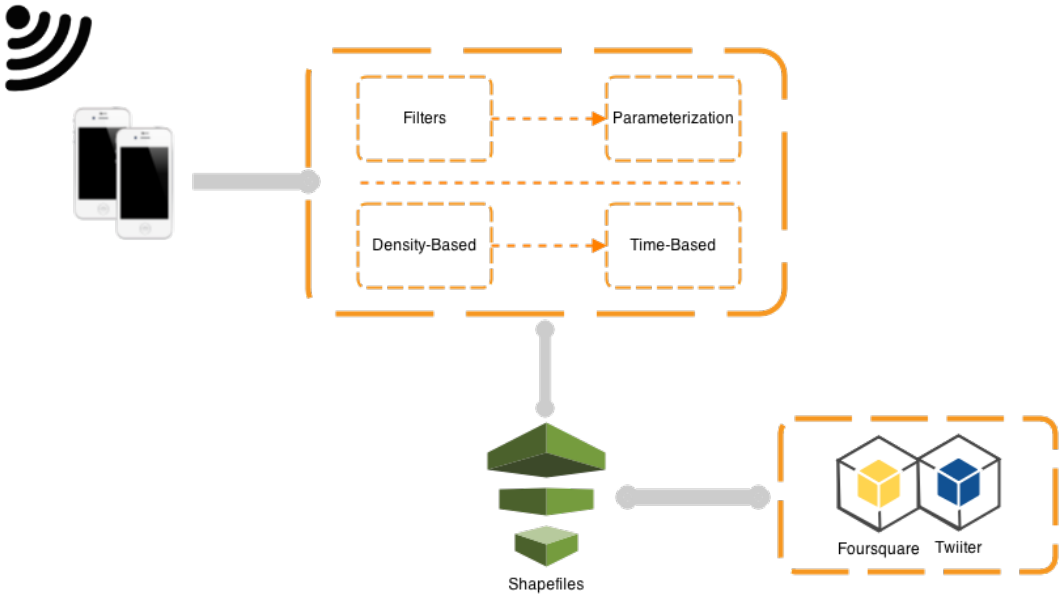


Figure 18 - Proposed Architecture.

4.1 Data Collection

For this work we used data traces provided by TU Delft[9] and also data traces collected through different applications for mobile devices such as SenseMyCity[10] and Moves[11] . It's important to emphasize that we used data traces with and without annotations to different roles. Data traces with annotations already contain information about meaningful places. Annotations have a key role because they allow the validation of the discovered locations by our algorithm. Also, we gathered data from Foursquare and Twitter - Location-Based Social Networks (LBSN).

4.1.1 TU Delft Data

These datasets were provided by 73 users from TUDelf University. The users are mostly workers and students at the university and they used GPS data loggers for the collection process. These data loggers retrieve new location every five seconds and users manually annotated the dwelling and POIs such as bars, university buildings, supermarkets and homes. In Figure 19 we show the number of GPS points per user. Also, in Figure 20 and in Figure 21 we can see the distribution of TU Delft data by gender and age respectively. In short, the dataset is balanced by gender however the majority of the users have an age between fifteen and thirty years old.

[9] Delft University of Technology

[10] http://futurecities.up.pt/site/crowdsensor-sensemycity-prototype-and-testbed

[11] http://www.moves-app.com/

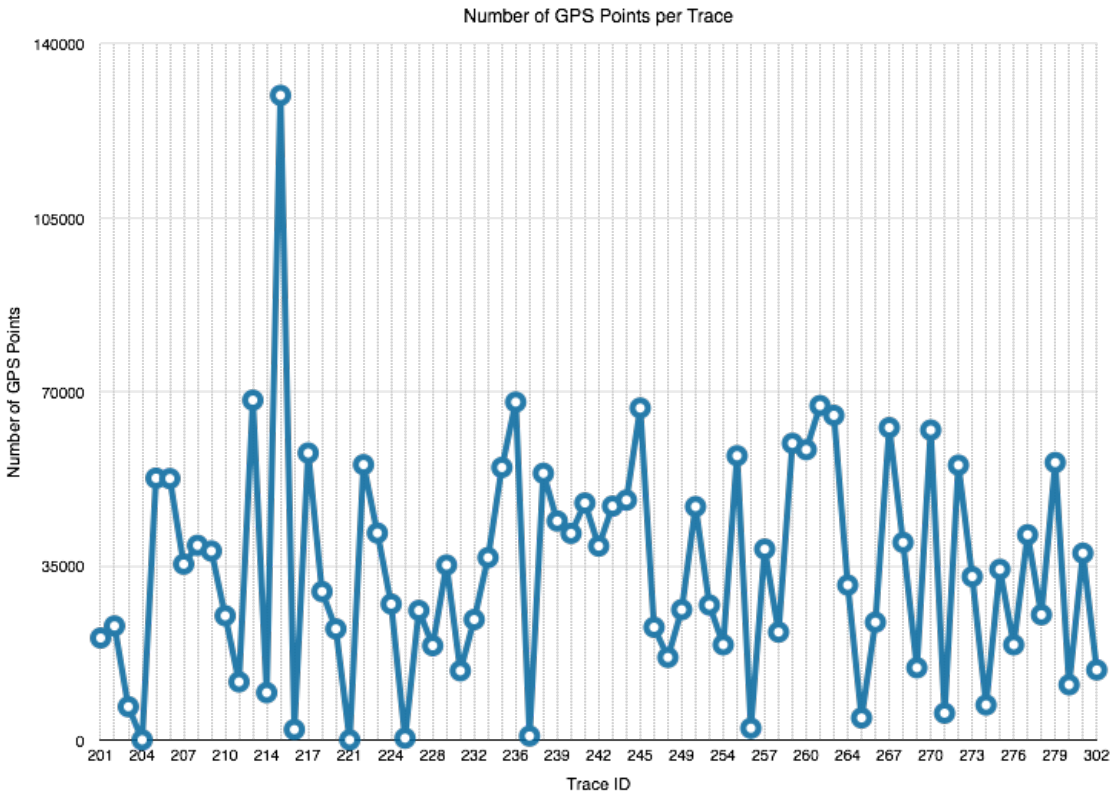


Figure 19 - Number of GPS points per user.

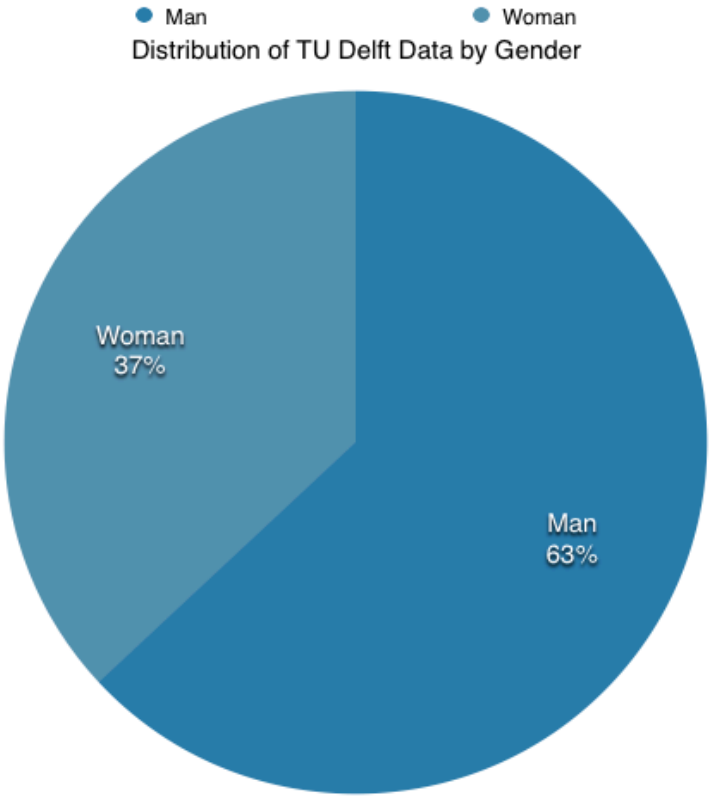


Figure 20 - Distribution of TU Delft data by gender.

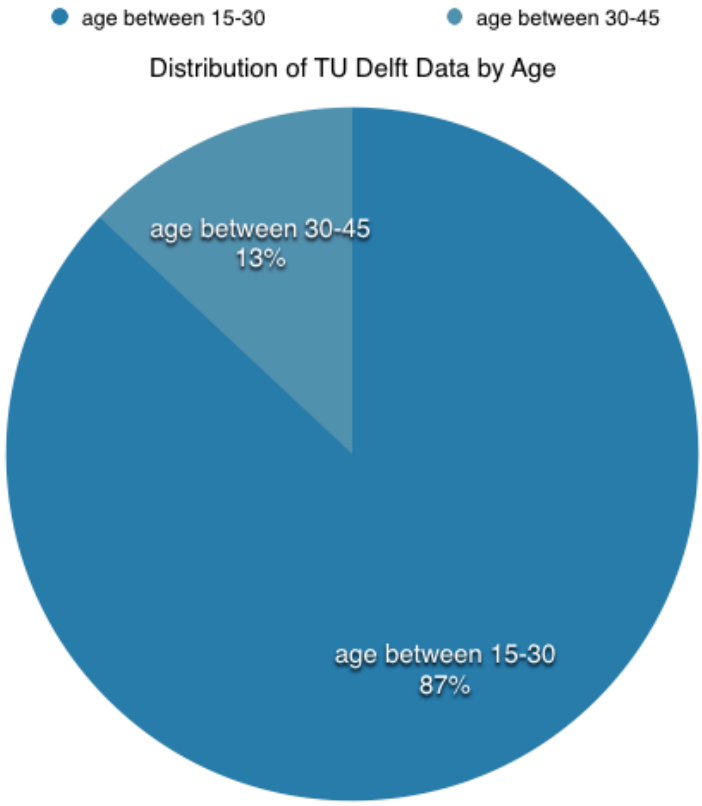


Figure 21 - Distribution of TU Delft data by age.

4.1.2 SenseMyCity + Moves Data

SenseMyCity is an Android application developed by University of Porto under the Future Cities project that uses built-in sensors to gather geo-indexed data. It enables, through the use of sensors, the registration of the everyday life of users for further analysis. The user can specify the granularity, i.e. interval between data collection, and for our study we considered a granularity of five seconds. Since SenseMyCity only gathers geo-indexed data without annotations we used Moves application to complement these data. Moves is an application that automatically annotates our daily routines and show where user stays and for how long like a daily storyline. The process of data collection was carried out by students and researchers from Department of Computer Science, University of Coimbra (UC).

4.1.3 Twitter Data

This dataset is collected from a widely used social media tool called Twitter where users can post short messages up to 140 characters. These short messages are specifically called "Tweets". When permissions are given by the users, each of their tweets are attached with a corresponding geo-location. While it is possible to collect tweets from Twitter, it takes a considerable amount of effort to gather a significant number of geo-tagged tweets because of certain practical limitations: first, Twitter's RESTful API solely supports the

simplest near-by search by means of the specification of a center location and a radius. Furthermore, each query can only obtain a maximum of 1,500 tweets per week. To circumvent this situation we used Twitter's Streaming API to push tweets that happen in near real-time by making a request for a specific type of data - filtered by geographic area - without needing to worry about polling or REST API rate limits. Through Twitter's Streaming API we create a boundary region for Delft and extract all the tweets observations within that region. For each tweet, we captured latitude, longitude, time, and tweet text. The location crawler ran from 01/01/2014 to 28/02/2014, resulting in a total collection of 1,873 distinct users and 19,487 unique tweets, as we can acknowledge from Table 2. Also, in Figure 22 we show the distribution of the number of tweets gathered per day during this period.

Table 2 - Properties of the Twitter dataset.

Dataset	Number of Distinct Users	Number of Tweets
Twitter	1,873	19,487

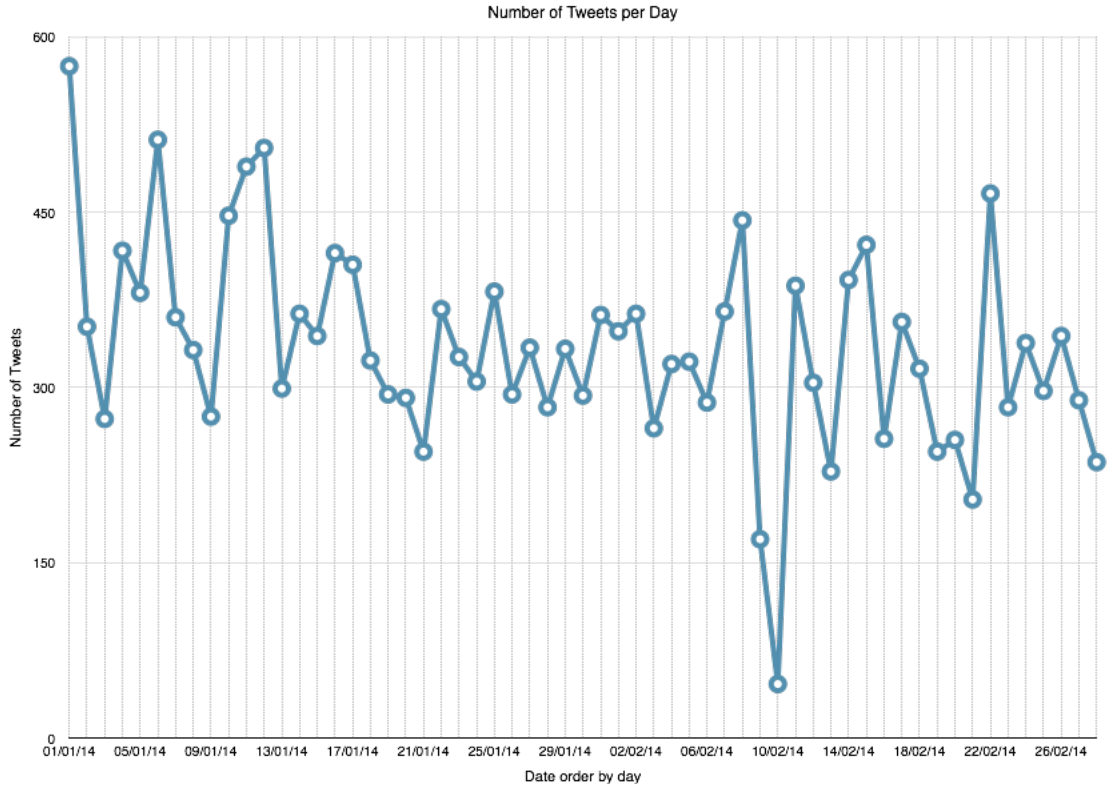


Figure 22 - Distribution of the number of tweets per day during two months.

4.1.4 Foursquare Data

This dataset is collected from a widely used social media tool called Foursquare. For the gathering process we used Foursquare API to extract as many venues as possible in the region of Delft and collect the check-ins at these venues. This process is possible through the search venues method that returns a list of venues near the current location.

The main drawback of this method is that is subject to result limits (up to 50 venues). To circumvent this situation we created a grid for the entire Delft area that consists on a bounding box that contains locations (pair of latitude and longitude) and a radius of 100 meters between them. For each location we ran the search venues method thus guaranteeing the collection of all venues for Delft area. However, different locations can return the same venue so it is necessary to filter the data in order to remove duplicate venues. The venue crawler returned total collection of 4,480 distinct venues and 604,807 check-ins that its users have carried out since the inception of the service, as we can acknowledge from Table 3. For each venue, or place, we are aware of its geographic coordinates. Further, category information about each place has been crowd sourced by Foursquare users. Thus a venue has been associated with a semantic label that signifies its type. There are two hierarchical level of places categories in Foursquare, a more general one that describes the place in an abstract way (for instance Nightlife Spot) and a more specific one (for instance Bar). In the context of the present work we will use the more general category hierarchy for which we have nine variations: Arts & Entertainment, College & University, Food, Professional & Other Places, Nightlife Spot, Great Outdoors, Shop & Service, Travel & Transport and Residence. In Table 4, we show the more general category hierarchy and their respective specific categories considered. Also, in Figure 23 we can see the distribution of each abstract category in our Foursquare dataset. Our assumption is that Foursquare check-in activity can be used to identify the type of urban activity occurring in a city's neighbourhood.

Table 3 - Properties of the Foursquare dataset.

Dataset	Number of Distinct Venues	Number of Check-ins
Foursquare	4,480	604,807

Table 4 - Abstract and specific categories of Foursquare.

Abstract Categories	Specific Categories
Arts & Entertainment	Art Gallery, Casino, Concert Hall, Historic Site, Movie Theatre, Museum, Music Venue, Stadium, Theme Park, Zoo, etc.
College & University	College, Fraternity House, University, etc.
Food	Restaurant, Bakery, Café, Coffee Shop, etc.
Nightlife Spot	Bar, Pub, Nightclub, etc.
Outdoors & Recreation	Athletics & Sports, Beach, City, Castle, Island, Lake, Mountain, Park, Playground, Plaza, River, etc.

Abstract Categories	Specific Categories
Professional & Other Places	Convention Center, Factory, Fair, Government Building, Library, Medical Center, Office, Post Office, School, etc.
Residence	Home (private), Residential Building (Apartment/Condo), etc.
Shop & Service	Clothing Store, Convenience Store, Pharmacy, Food & Drink Shop, Furniture / Home Store, Gym /Fitness Center, Mall, Market, Supermarket, Store, Plaza, Bookstore, Boutique, Miscellaneous Shop, Automotive, etc.
Travel & Transport	Airport, Bus Station, Ferry, Hotel, Subway, Taxi, Train Station, etc.

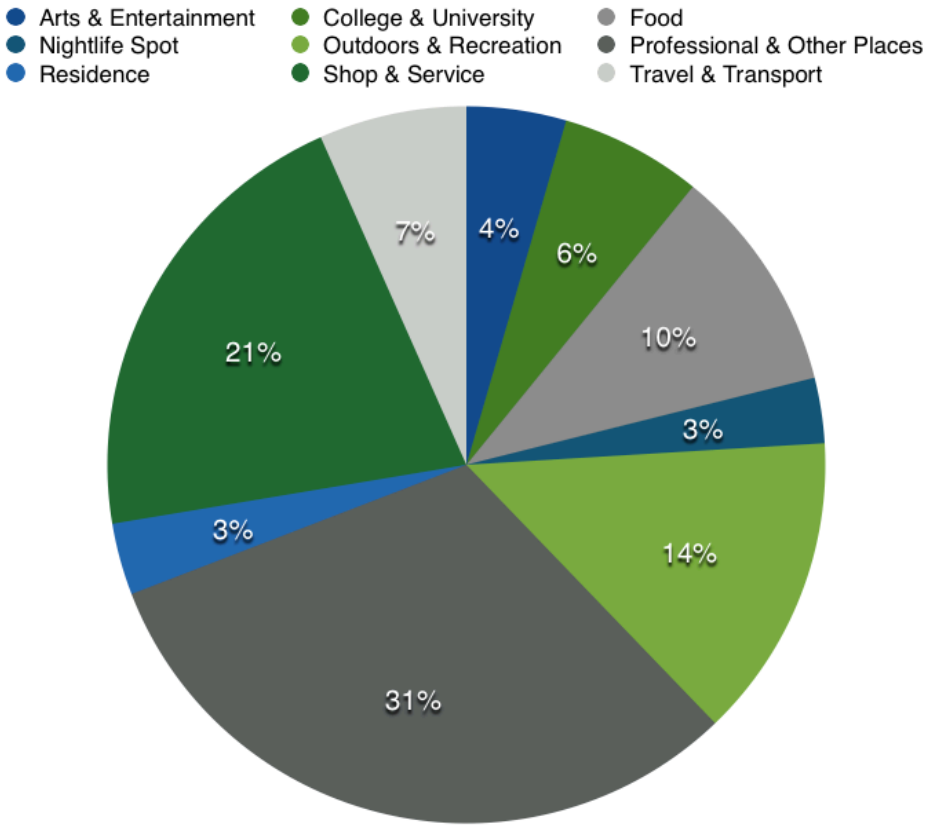


Figure 23 - Distribution of each abstract category in our Foursquare dataset.

4.2 Data Filters

GPS like any location-based service often produce inaccurate estimates due to a variety of reasons. There are several factors that introduce errors into location systems such as:

- Multipath interference – satellite signal is reflected off tall buildings, power lines, water and other interfering objects. This causes the signal to be delayed before it reaches the receiver;
- Atmospheric interference - the atmosphere can slow the satellite signal;
- Geometric Dilution of Precision (GDOP) computation – errors caused by the geometry of the group of satellites from which signals are being received. DOP depends on the position of the satellites: how many satellites you can see, how high they are in the sky, and the bearing towards them;
- Almanac and Ephemeris errors - warm start/cold start problem results in missing GPS points at the beginning of the trip due to the time the GPS receiver needs to acquire the position of at least four satellites in view.

As most of the GPS data comes with noise it is necessary to clean/filter non-relevant data. For this process the following approaches were considered:

- Remove duplicated entries based on their location;
- Bypass points whose distance between two consecutive GPS points is higher than 20 meters. The average human walking speed is around 5Km/h and GPS gets a new location every 5 seconds for these traces. Considering the mobility mode "walking", based on previous variables, we can conclude that at most one person travels 6,9 meters in 5 seconds. Considering also the GPS deviation, a value between 0-10 meters, the distances between two consecutive points that are higher than 20 meters can be discarded;
- Remove points with degenerated DOP values;
- Remove points above elevation considering the specific country topology.

4.3 Clustering of GPS points

In this section we propose a robust hierarchical clustering algorithm (density and time based) to extract intentional stops from GPS traces based on a three-layered model. The idea is to overcome previous approaches, replacing the intervention of the user, i.e. remove the need for the user to explicitly refer which are their intentional stops. Our approach is based on [61], however we introduced some modifications on definitions. As shown in Figure 24, the lowest and the first level is the trajectory data, which is a set of GPS Points (see Definition 1), the middle level is the intentional stops extracted from user's trajectory data (see Definition 2) and the highest level represents the background geographic information as places (see Definition 3). This hierarchical clustering algorithm firstly extracts clusters from trajectory data through a density-based algorithm; secondly for each cluster conducts a time-based algorithm to identify the intentional stops and finally assigns identified intentional stops to a physical place represented as background geographic information, i.e. polygons.

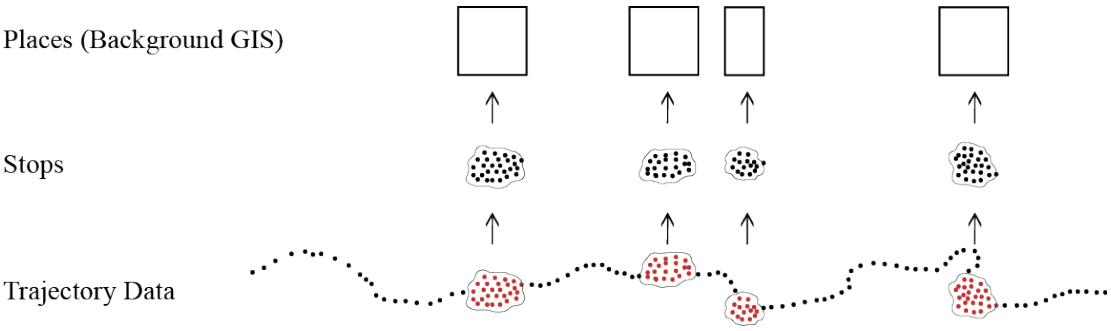


Figure 24 - A three-layered model to extract intentional stops from GPS traces.

Definition 1. (Trajectory Data) GPS Trace is a set of GPS points representing mobility of the user. A GPS point is a pair of coordinates with timestamp $p = (t, lat, lon)$ where *lat* and *lon* represents latitude-longitude location and *t* the timestamp.

Definition 2. (Stops) Intentional stop is a set of GPS points where user spends more time than a threshold. Each intentional stop has GPS points sorted by their timestamp. To define the dwell time for each intentional stop we consider the entry and exit times.

Definition 3. (Places/Polygons) A physical place comprises intentional stops inside a polygon representing a place.

The Density and Time-Based algorithm is shown in Figure 24 and represent the lowest and the middle levels set out above. DBSCAN is a well-known density-based clustering which is designed to discover the clusters and the noise in a spatial database. For us a cluster represents a stop on a single trajectory and to extract the clusters we implemented an algorithm based on DBSCAN. According to [3] a stop represents a place where user stays more than a specific time threshold, i.e. dwell time, however DBSCAN does not consider the temporal dimension.

The basic idea behind DBSCAN is that, for each point of a cluster, the cardinality of the neighborhood of a given radius *Eps* (ε) has to exceed a threshold *MinPts*. One of the biggest drawbacks of this algorithm is that its very sensitive to parameterization. Thus to determine a good parameterization the user needs to know well the characteristics of each trajectory. To overcome this problem, according with [14] we implemented a heuristic based on *k-th nearest neighbor* to automatically determine parameters (line 3). After determining the algorithm parameterization we apply DBSCAN on the trajectory data to extract clusters that represent stops (line 4). However, our goal is to create an algorithm that combines the two dimensions: spatial and temporal. Therefore, for each cluster C_i found, all the points that belong to it are sorted by their timestamp to get the entry and exit times (line 6). If the user spends time in the same place but in different periods of the day this is represented by one single cluster. If these different periods of time have a gap between them above δ_{gap}, then the algorithm split the current cluster C_i into mini clusters MC_i which each mini cluster represents a different period of the day

(lines 7-8). Finally, a new meaningful place is found when user spends more time than $\delta_{dwell\ threshold}$ on each MC_i (line 10).

Algorithm 1 Extract Meaningful Places from GPS Traces (Step 1)

```
1:  function ExtractPlaces(T)
2:      RemoveNoise(T)
3:      DetermineParameters(T)
4:      clusters ← DBSCAN(T)
5:      for each cluster C_i in clusters do
6:          Sort ∀ points ∈ C_i by timestamp
7:          if C_i(out) − C_i(in) > δ_gap then
8:              Split C_i into MC_i
9:          if MC_i > δ_dwell threshold then
10:             newLocation ← MC_i
```

Figure 25 - A Density and Time-Based Algorithm to extract intentional stops.

4.4 Incorporating Constraints and Place Detection

This subsection presents our main contribution: the use of constraints to assign semantic meaning to the locations discovered by the previous algorithm. Unlike other approaches, which use geographic constraints, i.e. physical obstacles, to split clusters that intersect them, we use these physical obstacles to represent intentional stops from the trajectory data where user spends considerable amount of time because we believe that these are meaningful places to the user. For the convenience of comparison of clustering results, Figure 26 illustrates cluster points, and obstacles: (a) before clustering in absence of obstacles; (b) clusters in the presence of obstacles, approaches conducted by other authors where obstacle split the main cluster into others clusters; (c) clusters in the presence of obstacles where the obstacle incorporates the points that lie within.

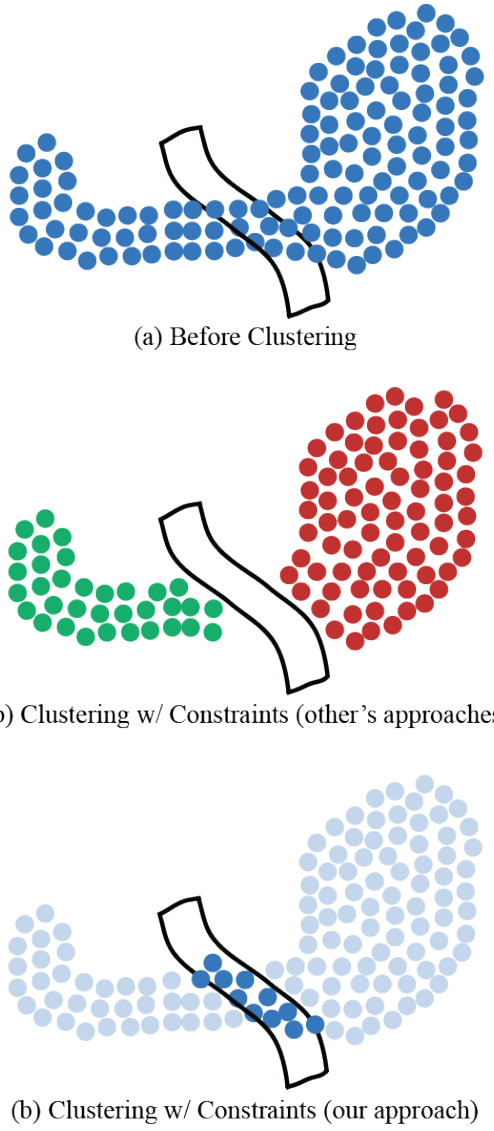


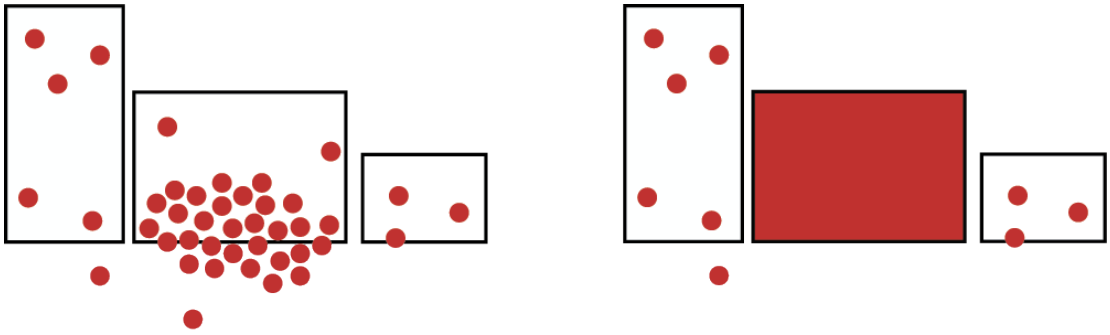
Figure 26 - A comparison between the different clustering algorithms with constraints.

To extract meaningful places from GPS traces incorporating constraints we create the following algorithm presented in Figure 27, which use background geographic information provided by Openstreetmap. After finding intentional stops (clusters), the algorithm intersects each point P_i that belong to the clusters with shape files (lines 3-4) and this process is performed through QGIS application. Basically, shape files are a representation of a city in the form of polygons that contains buildings like bars, universities, supermarkets and homes. As each cluster is a set of points without embedded semantic meaning we create a representation where polygons with more points inside represent the relevant places instead of a set of points - Figure 28. For the convenience of comparison of representation results, Figure 28 illustrates cluster points and polygons: (a) representation of clusters without semantic meaning; (b) the polygon with more points inside becomes representative of the place instead of a set of points.

Algorithm 2 Extract Meaningful Places From GPS Traces (Step 2)

```
1: function IncorporateConstraints(location)
2:     polygon = shapefile
3:     polygons = {}
4:     for each point P_i in location do
5:         if P_i ∈ polygon then
6:             polygon = P_i
7:             Add polygon to polygons
8:     meaningfulPlaces = Select polygons with more points inside
```

Figure 27 - Algorithm responsible for incorporating constraints.



(a) Before Clustering (b) Clustering with Constraints

Figure 28 - Representation of the meaningful places through physical obstacles.

4.5 Semantic Enrichment of Places

Since some of the polygons that belong to the shapefiles don't have any category (semantic label) embedded we used venues gathered from Foursquare for the semantic enrichment process. Each venue contains a geo-location thus using QGIS we crossed venues locations with polygons. This process was run through "select by location" method, which verifies if a point is within a polygon and merges its information. In short, we classified 1,072 polygons with Foursquare venues categories. In Figure 29 we see the distribution of each abstract category after the semantic enrichment process and in Figure 30 we show polygons (red) with semantic label collected from Foursquare.

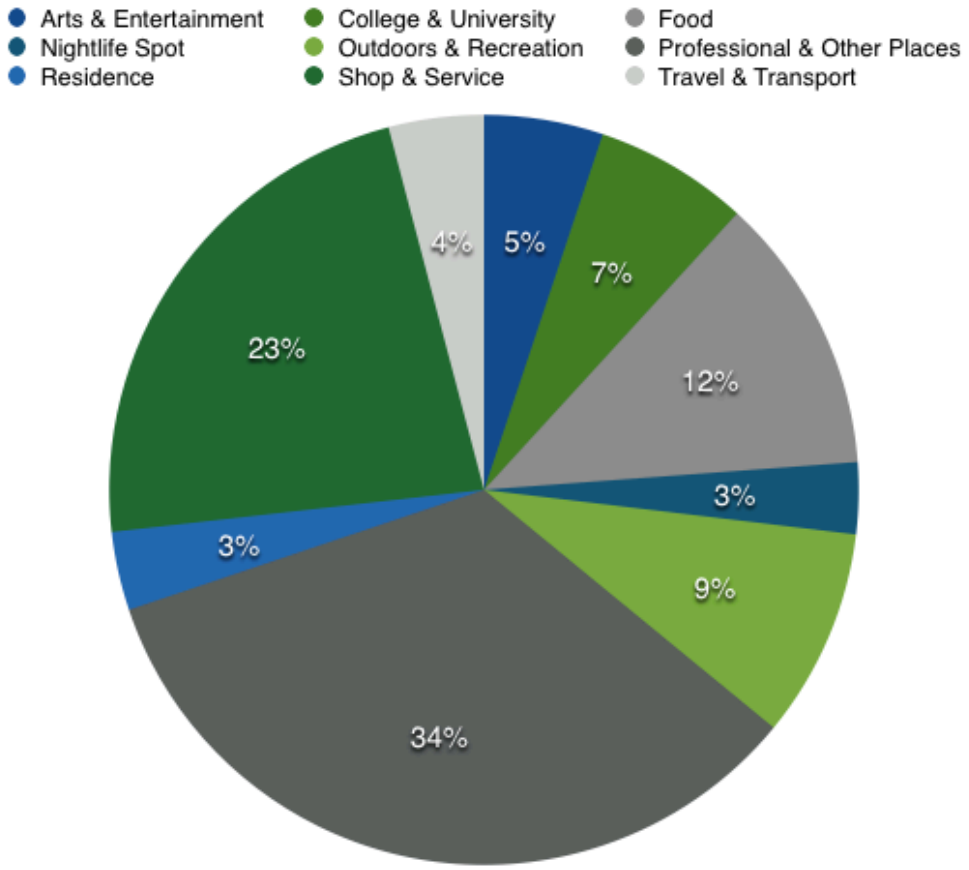


Figure 29 - Distribution of each abstract category after the semantic enrichment process.

Figure 30 - Polygons with semantic label collected from Foursquare (red).

5 Results

We conduct a series of experiments to evaluate our approach. This chapter reports the results of these experiments and the characterization of the aggregate activity patterns by finding the distributions of different activity categories over a city geography and study how social aspects can affect human mobility patterns.

5.1 Constraint-Based Algorithm

To perform the experiments we have used: trajectory data with annotations collected in the city of Delft; and shapefiles for this city. These annotations already contain the places where user spends a considerable amount of time and will be used to validate the intentional stops discovered by our algorithm. For the experiments we considered the parameter δ (dwell threshold) as 300 seconds. Also the shape file dataset considered has around 100 thousand buildings as potential places for the assigning process. As shown in Table 5 through our spatio-temporal algorithm we found several intentional stops where user spends more than 300 seconds. However, some of those intentional stops may represent traffic jam, traffic lights and even position deviations situations. Considering only the intentional stops that are inside polygons we can represent them as physical places and so add semantic meaning. This process would also reduce significantly the number of intentional stops considered. Sometimes the same building may contain several intentional stops, especially large buildings, e.g. someone go to the mall and spends considerable amount of time in different locations. Looking for the candidate stops provided by the annotations and places that our algorithm assigned correctly we conclude that the approach followed can be very precise. However, several factors may introduce errors into location systems and so, it's possible that sometimes the buildings are incorrectly assigned as we can see in Figure 32. For the convenience of comparison of the results obtained through our approach, Figure 31 and Figure 32 illustrates the intentional stops and the polygons: (a) trajectory data which contains all the GPS points; (b) intentional stops that are inside of the boundaries of the polygons; (c) buildings that represent intentional stops with interest/semantic meaning; (d) annotated data which contains the candidate stops.

Table 5 - Number of clusters discovered with and without Constraints.

Dataset	#1	#2	#3	#4	#5	#6	#7	#8	#9
Number of GPS Points	1080	419	1845	899	1864	999	1230	258	1759
Number of Clusters (algorithm without Constraints)	22	9	72	20	81	26	28	4	14
Number of Clusters (algorithm with Constraints)	11	4	17	7	8	21	22	3	9

Table 6 – True Positive and False Positive rate.

Dataset	#1	#2	#3	#4	#5	#6	#7	#8	#9
Intentional Stops (through annotations)	4	4	8	3	9	4	8	6	7
True Positive Rate (%)	4	3	8	3	7	3	6	2	5
False Positive Rate (%)	1	1	1	0	2	1	2	4	2

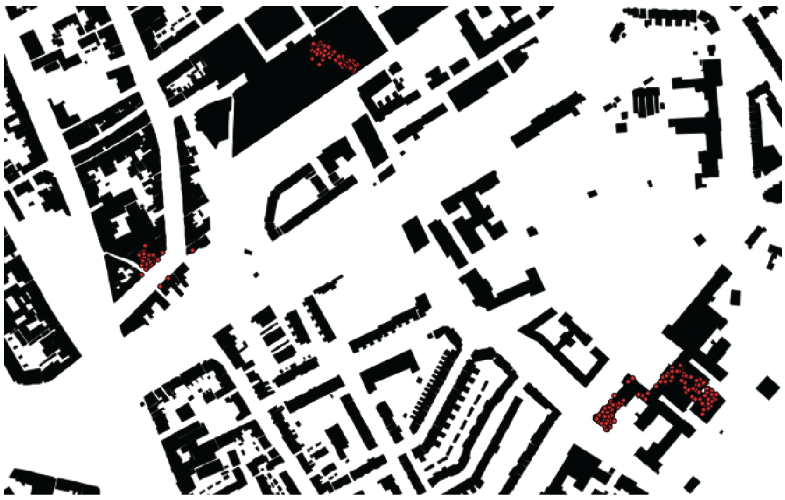
Table 7 - Conclusions about the algorithm.

Weighted average number of Places assigned correctly (%)	Average number of reduction intentional stops considering Constraints (%)
83%	51%

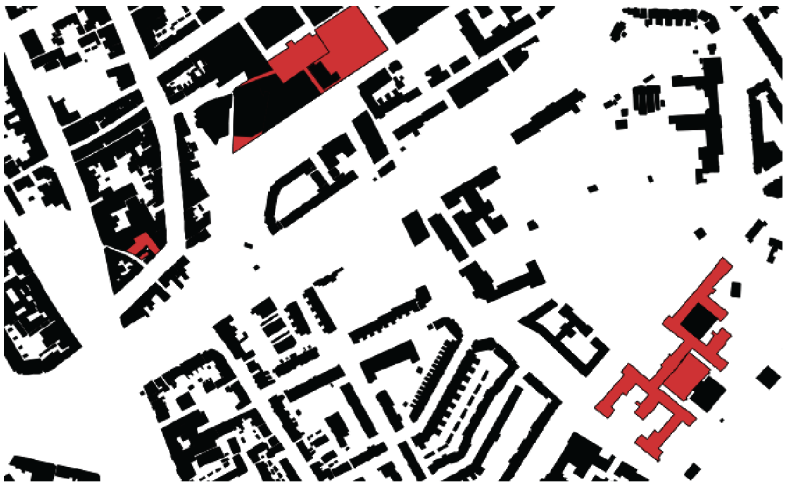
Conclusions: Summing up, with our algorithm constrained by geographical information we obtained an **83%** weighted average number of places assigned correctly and a **51%** average number of reduction intentional stops – see Table 7. Also we show in Table 6 the true positive rate (corresponding to the proportion of intentional stops which are correctly identified) and the false positive rate (corresponding to the proportion of intentional stops which are incorrectly identified). To emphasize that for the **#5** dataset we obtained a large reduction because most of the clusters were discovered in areas such

as streets. Although that there are several inherent factors that introduce errors into location systems, not directly related to the user, and also that is very common that GPS signal may be lost inside buildings, we demonstrate the effectiveness of the proposed algorithm. To further improve this algorithm we could introduce other sensors, such as motion, and WI-FI. Furthermore with our approach we eliminated intentional stops that represent traffic jams and traffic lights because these are not represented as buildings.

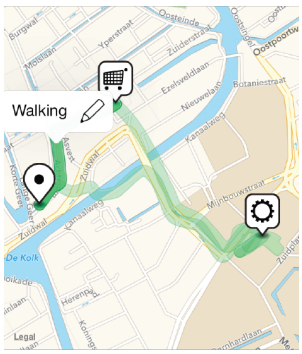
(a)



(b)



(c)



(d)

Figure 31 - Experiment with 100% of accuracy in the process of assigning places.

Figure 32 - Experiment with errors in the process of assigning places.

5.2 Twitter

In this sub-section we present an analysis of the dynamics of Twitter's users activity. Through Twitter a large number of people around the world share updates and information about their whereabouts. We can consider the utilization of such geo-tagged tweets, which, in a sense, are a life log of each user, indicating where a user exists and what she/he is currently doing. Our aim is to understand the crowd activities reflected on Twitter and to understand which are the places of aggregation of Twitter users in the city of Delft. For the purpose of this work we only used geo-tagged tweets. Using QGIS we intersected both tweets and venues from Foursquare to understand which are the places where Twitter's users share updates and information - see Figure 33. Thus, the places with more activity are - Professional & Other Places, Shop & Service, College & University and Food – while the remaining places have a smaller number of shares. In Figure 34 and Figure 35 we plot the characterization of the venues by number of check-ins and by categories respectively.

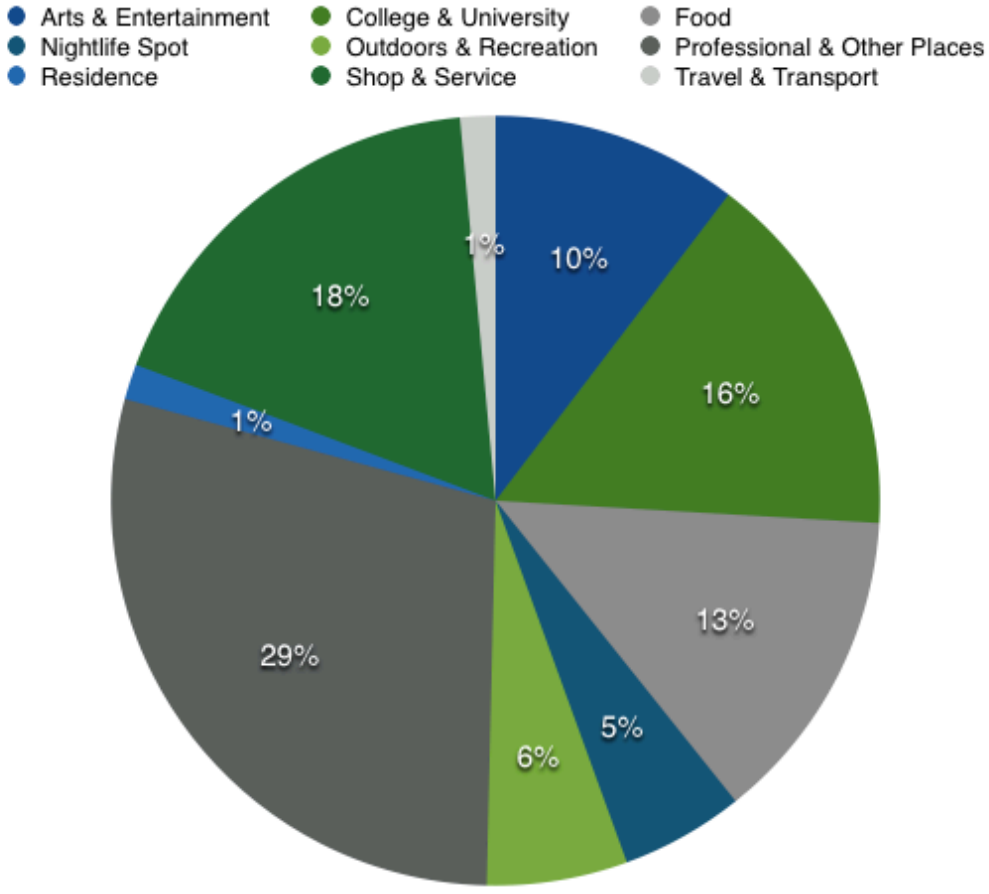


Figure 33 - Distribution of the places (categories) where Twitter's users share their updates.

Table 8 - Number of check-ins for the most popular venues.

Venue Abstract Category	**Venue Specific Category with number of tweets (distinct users)**
Food	Vietnamese Restaurant (224); French Restaurant (66); Scandinavian Restaurant (26)
Professional & Other Places	Office (67); Conference Room (30)
Shop & Service	Salon / Barbershop (46)
College & University	College Library (43); College Lab (33); College Classroom (35)
Arts & Entertainment	Movie Theater (54)
Nightlife Spot	Sports Bar (38)

Conclusions: Summing up, the category Professional & Other Places is dominant in Twitter community (Delft). This means that category Professional & Other Places has the largest number of distinct venues where users share their information. However, from Table 8 we can see that there is a pattern concerning the appearance of tweets in Food and College & University places. The most popular place among Twitter users in Delft is a **Vietnamese Restaurant** - named Huong Viet - with 224 distinct tweets. Also, College & University category is well represented which leads to the conclusion that Delft is a town with a very strong education activity. Thus, we confirmed that crowd activities determined via Twitter can reflect and characterize living spaces in urban areas.

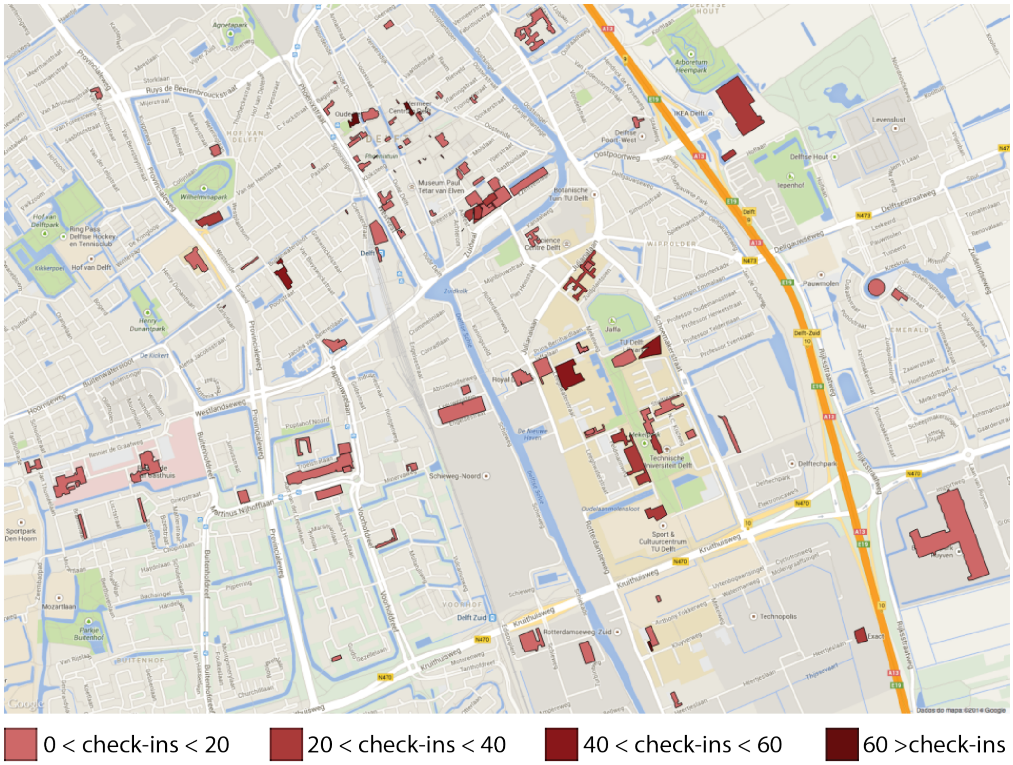


Figure 34 - Characterization of the venues by number of tweets.

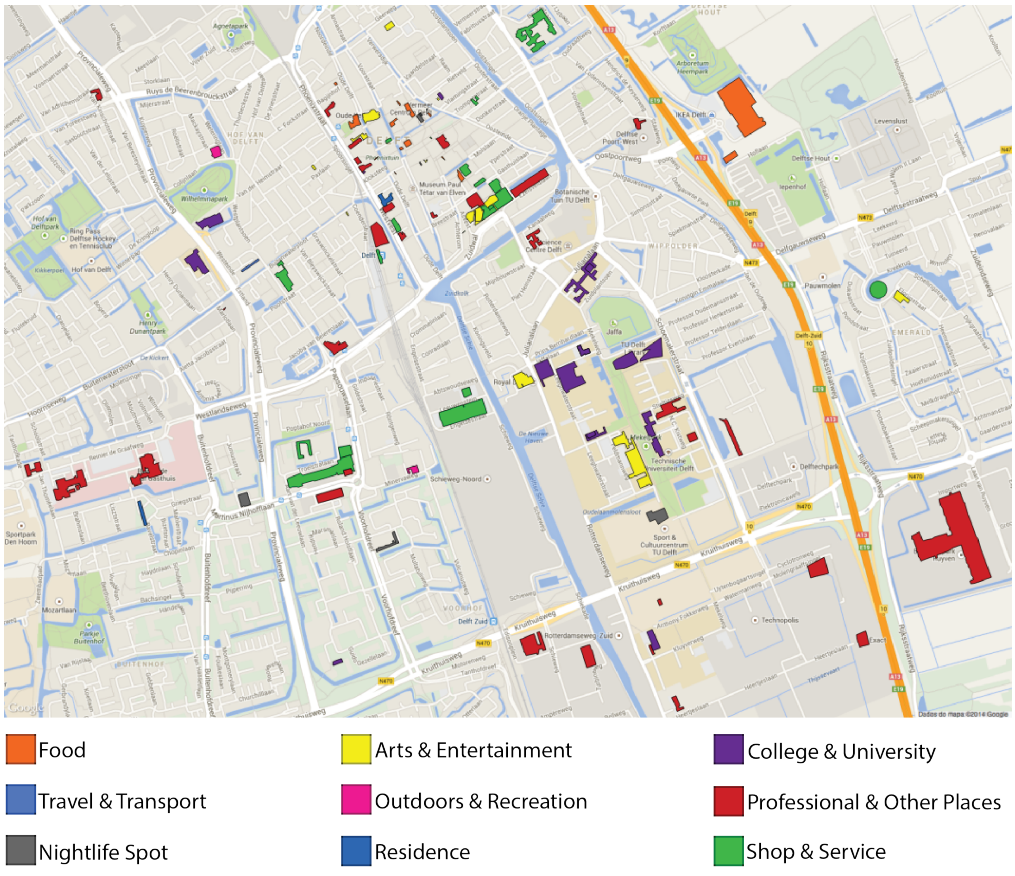


Figure 35 - Characterization of the venues by category.

5.3 Delft

In this sub-section we present an analysis of the dynamics of Delft's users activity. Our aim is to understand how social aspects can affect human mobility patterns. First, using our algorithm with geographic constraints we discovered the intentional stops and also added semantic labels to these locations with information collected from Foursquare. Second, for each intentional stop we extracted information about the user who spent a considerable amount of time in these locations. To this study we considered both gender and age of the users. Considering the gender, from Figure 36 and Figure 37 we can see the distribution of the places visited by women and men respectively. In short, the categories are well balanced for the two genders. However, analysing the specific categories in more detail we can draw some assumptions. From Table 9 we can see that women activity is focused on places of cultural interest and stores. On the other hand, men activity is focused on technology places and on Fitness Centers. Another interesting point is that men prefer fast food restaurants. From Figure 38 we can see the distribution of the places visited by people with an age between 15-30 and 30-45. All the places are mostly visited by people with an age between 30 and 45 years old. However, analysing the specific categories in more detail we can draw some assumptions. From Table 10 we can acknowledge that people with an age between 30 and 45 years old value the use of public transports most. Also, their activity is focused on places of cultural interest. On the other hand from Table 10 we can acknowledge that people with an age between 15 and 30 years old go to Gyms more. Another interesting point is that nightlife spots are mostly visited by older people.

Table 9 - How often users (gender) visit specific places.

Place (specific category)	How often (Men)	How often (Women)
Hardware Store	6	0
Gym/Fitness Center	4	0
Fast Food Restaurant	4	0
Retail	5	11
Department Store	0	5
College Library	1	8
Arts Centre	1	4

Table 10 - How often users (age) visit specific places.

Place (specific category)	How often (age between 15 and 30)	How often (age between 30 and 45)
Arts Centre	0	5
Train Station	0	23
Gym / Fitness Center	3	1
Department Store	4	1
Furniture / Home Store	2	30
Pub	0	8
Sports Bar	0	9

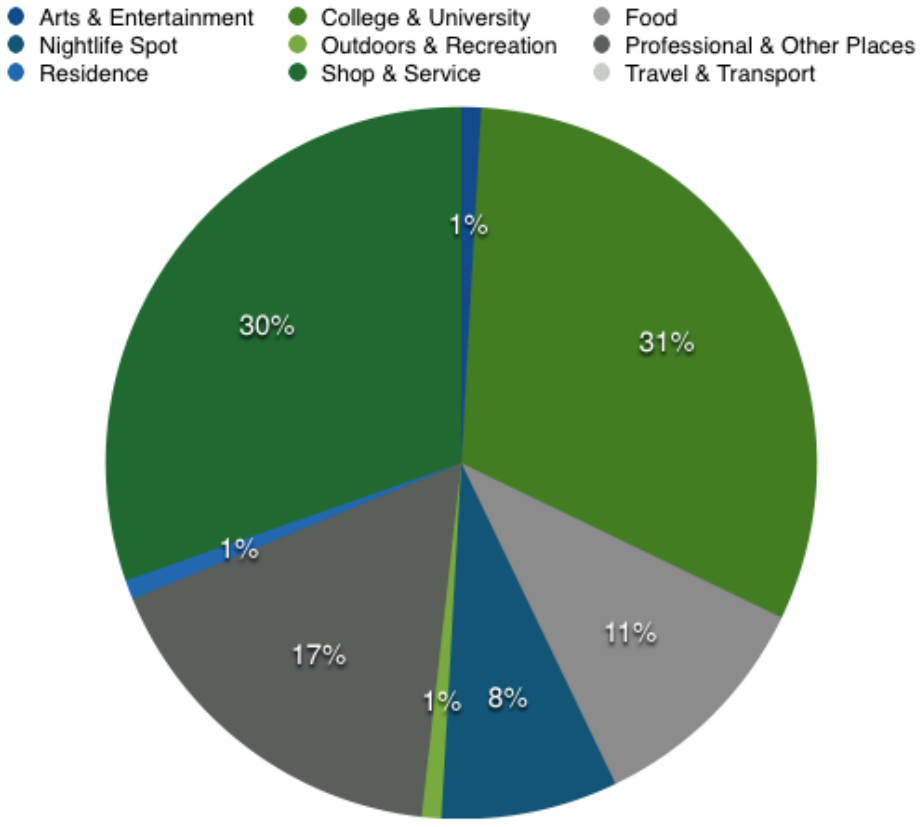


Figure 36 - Distribution of the places (categories) for women.

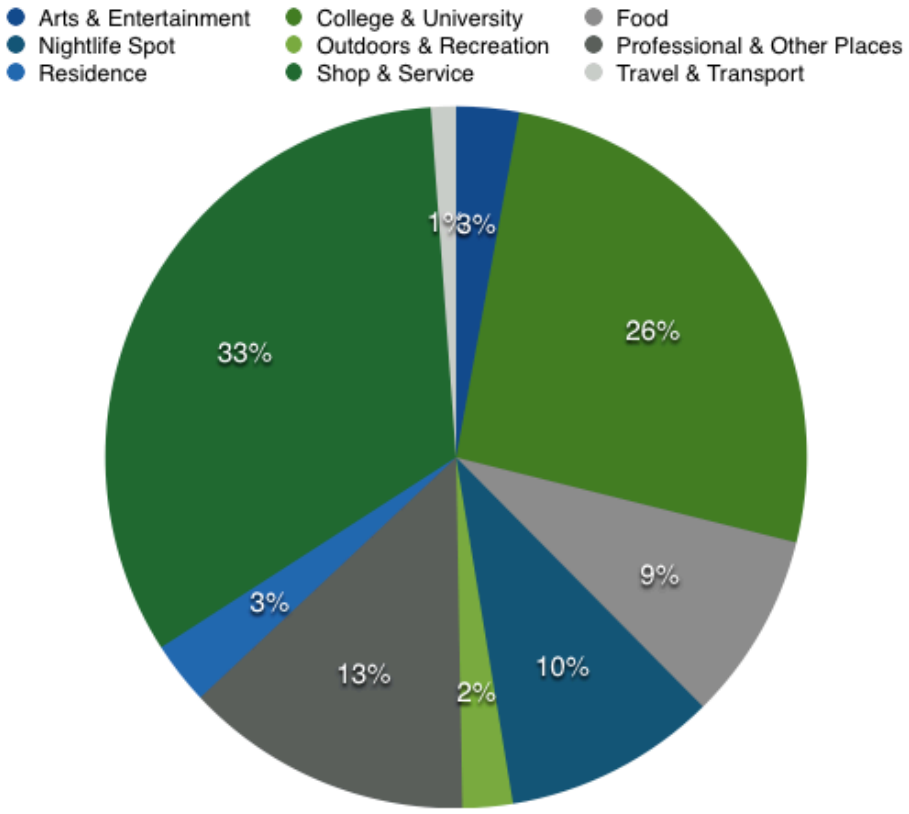


Figure 37 - Distribution of the places (categories) for men.

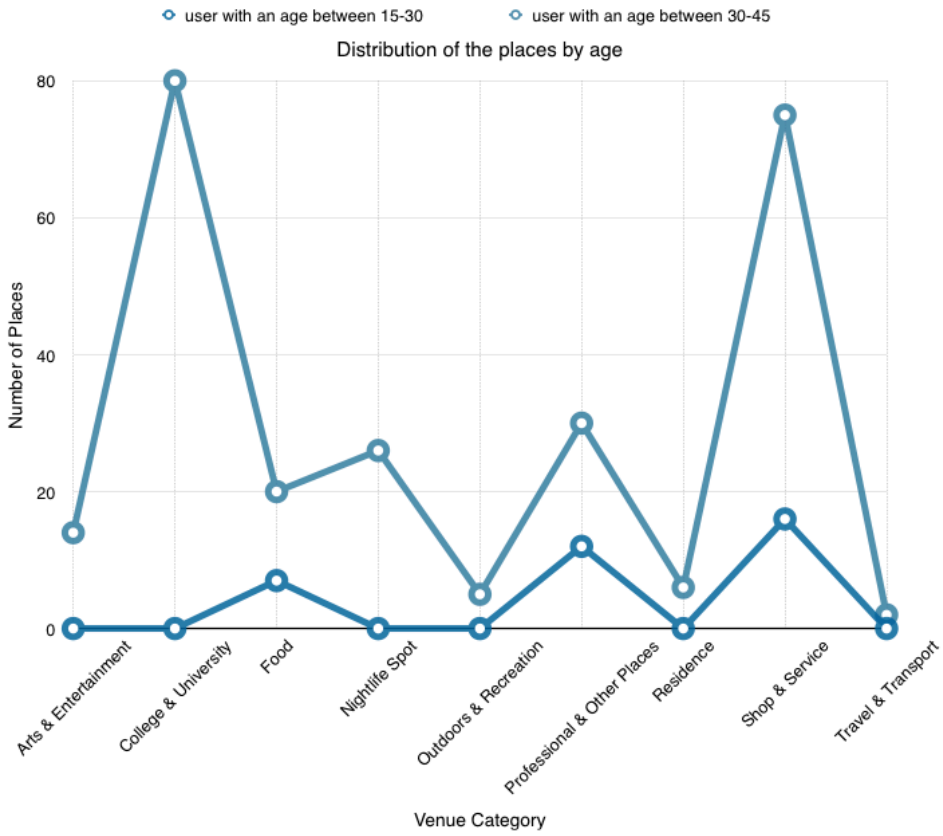


Figure 38 - Distribution of the places by user's age.

6 Conclusions and Future Work

Devices fitted with location-based services, such as GPS or mobile devices are becoming ubiquitous and therefore enable us to collect huge quantity of positioning data representing people's movements. Therefore, trajectory data represents a good basis to extract knowledge and mobility patterns but these data do not carry user-level notion of "place". One of the available mechanisms to extract knowledge from these data is through the application of clustering techniques. However the existing approaches have some drawbacks such as: the user has to specify which physical spaces he considers relevant to its trajectories; geographic information is used to constrain the clustering algorithm and not to create a physical representation of a place. In this work we introduce a new approach to discover intentional stops from the trajectories of users, in presence of noisy data. It is a spatial and time based clustering algorithm with constraints, i.e. background geographic information. In general, the main contributions of this work include:

- The creation of an algorithm that discovers the intentional stops from user trajectories. To overcome the existing approaches the user does not have to specify which physical spaces he considers relevant, i.e. candidate stops;
- Use geographic information in the form of shapefiles to represent the intentional stops instead of using geographic information to constrain the clustering algorithm;
- Process of semantic enrichment of shapefiles through Foursquare;
- Characterization of the aggregate activity patterns by finding the distributions of different activity categories over a city geography and study how social aspects can affect human mobility patterns.

The results drawn from a dataset of real GPS trajectories demonstrate the effectiveness of the proposed algorithm. With our algorithm constrained by geographical information we obtained a **83%** average number of places assigned correctly. Also our approach reduces trajectory data (**51%**) to only the relevant stops and creates a new representation for them based on background geographic information. Since some of the geographic information that belongs to the shapefiles doesn't have any category (semantic label) embedded we used venues gathered from Foursquare for the semantic enrichment process. In short, we classified **1,072 polygons** with Foursquare venues categories. Also, we conclude that social aspects can affect human mobility patterns. In future works, we will perform experiments of our algorithm for larger datasets. We intend to create a simple heuristic for the ambiguity problems caused by GPS deviations during the assigning procedure. Finally, temporal variations of geographic user activity can be taken into account in order to characterise area and users at certain periods of a day (i.e.,

morning, night etc.). Moreover, additional semantic information such as topics discussed at areas could be mined by data sourced from user tips, tags and comments.

References

1 Kang J. H., Welbourne W., Stewart B., and Gaetano B. Extracting places from traces of locations. In *WMASH '04 Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots* (2004), ACM New York, 110-118.

2 Brouwers N., and Woehrle M. *Detecting Dwelling in Urban Environments Using GPS, WiFi, and Geolocation Measurements*.

3 Palma A. T., Bogorny V., Kuijpers B., Alvares L. O. A clustering-based approach for discovering interesting places in trajectories. In *SAC '08 Proceedings of the 2008 ACM symposium on Applied computing* (2008), ACM New York, 863-868.

4 Jain A.K., Murty M. N., and Flynn P. J. Data clustering: a review. *ACM Computing Surveys (CSUR)*, 31, 3 (September 1999), 264-323.

5 Han J., Kamber M., and Pei J. *Data Mining: Concepts and Techniques*. 2012.

6 P., Berkhin. Survey Of Clustering Data Mining Techniques. In *Grouping Multidimensional Data*. Springer Berlin Heidelberg, 2006.

8 Bradley P. S., Fayyad U. M., and Reina C. Scaling Clustering Algorithms to Large Databases. In *Knowledge Discovery and Data Mining* (1998), AAAI Press, 9-15.

7 Kaufman L., and Rousseeuw P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 2005.

9 Ng R. T., and Han J. Efficient and Effective Clustering Methods for Spatial Data Mining. In *VLDB '94 Proceedings of the 20th International Conference on Very Large Data Bases* (San Francisco 1994), Morgan Kaufmann, 144-155.

10 H., Ralambondrainy. A conceptual version of the K-means algorithm. *Pattern Recognition Letters*, 16, 11 (Nov. 1995), 1147-1157.

11 Guha S., Rastogi R., and Shim K. CURE: an efficient clustering algorithm for large databases. In *SIGMOD '98 Proceedings of the 1998 ACM SIGMOD international conference on Management of data* (New York 1998), ACM New York, 73-84.

12 Karypis G., Han E., and Kumar V. Chameleon: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, 32, 8 (Aug. 1999), 68-75.

13 Zhang T., Ramakrishnan R., and Livny M. BIRCH: an efficient data clustering method for very large databases. In *SIGMOD '96 Proceedings of the 1996 ACM SIGMOD international conference on Management of data* (NY 1996), ACM New York, 103-114.

14 Ester M., Kriegel H., Sander J., and Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining* (1996), AAAI Press, 226-231.

15 Ankerst M., Breunig M. M., Kriegel H., and Sander J. OPTICS: Ordering Points To Identify the Clustering Structure. In *SIGMOD '99 Proceedings of the 1999 ACM SIGMOD international conference on Management of data* (1999), ACM Press, 49-60.

16 Hinneburg A., Hinneburg E., and Keim D. A. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In *Proceedings of the 4th International Conference on Knowledge Discovery and Datamining (KDD'98)* (1998), AAAI Press, 58-65.

18 Kisilevich S., Mansmann F., and Keim D. P-DBSCAN: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In *COM.Geo '10 Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application* (NY 2010), ACM New York.

17 Zhou C., Frankowski D., Ludford P., Shekhar S., and Terveen L. Discovering personally meaningful places: An interactive clustering approach. *ACM Transactions on Information Systems (TOIS)*, 25, 3 (July 2007).

19 Wang W., Yang J., and Muntz R. R. STING: A Statistical Information Grid Approach to Spatial Data Mining. In *VLDB '97 Proceedings of the 23rd International Conference on Very Large Data Bases* (San Francisco 1997), Morgan Kaufmann Publishers Inc, 186-195.

20 Agrawal R., Gehrke J., Gunopulos D., and Raghavan P. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In *SIGMOD '98 Proceedings of the 1998 ACM SIGMOD international conference on Management of data* (NY 1998), ACM New York, 94-105.

21 Spaccapietra S., Parent C., Damiani M.-L., Macedo J. A. F., Porto F., and Vangenot C. A conceptual view on trajectories. *Data & Knowledge Engineering*, 65, 1 (Apr. 2008), 126-146.

22 Tung A. K. H., Hou J., and Han J. Spatial Clustering in the Presence of Obstacles. In *Data Engineering, 2001. Proceedings. 17th International Conference* (2001), IEEE, 359-367.

23 Estivill-Castro V., and Lee I. AUTOCLUST+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles. In *TSDM '00 Proceedings of the First International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining-Revised Papers* (2001), Springer Berlin Heidelberg, 133-146.

24 Zaïane O. R., and Lee C.-H. Clustering Spatial Data in the Presence of Obstacles: a Density-Based Approach. In *IDEAS '02 Proceedings of the 2002 International Symposium*

on Database Engineering & Applications (Washington 2002), IEEE Computer Society, 214-223.

25 Alvares L. O., Bogorny V., Kuijpers B., Macedo J. A. F. de, Moelans B., Vaisman A. A model for enriching trajectories with semantic geographical information. In *GIS '07 Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems* (NY 2007), ACM New York.

26 Hasan S., Zhan X., and Ukkusuri S. V. Understanding urban human activity and mobility patterns using large-scale location-based data from online social media. In *UrbComp '13 Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing* (NY 2013), ACM New York.

28 Fujisaka T., Lee R., and Sumiya K. Exploring urban characteristics using movement history of mass mobile microbloggers. In *HotMobile '10 Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications* (2010), ACM New York, 13-18.

27 Wakamiya S., Lee R., and Sumiya K. Urban area characterization based on semantics of crowd activities in Twitter. In *GeoS'11 Proceedings of the 4th international conference on GeoSpatial semantics* (2011), Springer-Verlag Berlin, 108-123.

29 Kinsella S., Murdock V., and O'Hare N. "I'm eating a sandwich in Glasgow": modeling locations with tweets. In *SMUC '11 Proceedings of the 3rd international workshop on Search and mining user-generated contents* (NY 2011), ACM New York, 61-68.

30 Cheng Z., Caverlee J., Lee K., and Sui D. Z. Exploring millions of footprints in location sharing services. In *ICWSM* (2011).

31 Noulas A., Scellato S., Mascolo C., and Pontil M. Exploiting semantic annotations for clustering geographic areas and users in location-based social networks.. In *3rd Workshop Social Mobile Web (SMW 2011)* (2011).

32 Noulas A., Mascolo C., Scellato S., and Pontil M. An Empirical Study of Geographic User Activity Patterns in Foursquare. In *ICWSM* (2011).

33 Pensa R. G., Robardet C, and Boulicaut J.-F. A bi-clustering framework for categorical data. In *PKDD'05 Proceedings of the 9th European conference on Principles and Practice of Knowledge Discovery in Databases* (2005), Springer-Verlag Berlin, 643-650.

34 Shmueli G., Patel N. R., and Bruce P. C. *Data Mining for Business Intelligence*. 2010.

35 J., Surma. *Business Intelligence: Making Decisions through Data Analytics*. 2011.

36 Voevodski K., Balcan M.-F., Röglin H., Teng S.-H., Xia Y. Active clustering of biological sequences. *The Journal of Machine Learning Research* , 13, 1 (Jan 2012), 203-225.

38 Carpineto C., Osiński S., Romano G., and Weiss D. A survey of Web clustering engines. *ACM Computing Surveys (CSUR)*, 41, 3 (Jul 2009).

37 Nugent R., and Meila M. An Overview of Clustering Applied to Molecular Biology. In *Statistical Methods in Molecular Biology*. 2010.

39 Zamir O., and Etzioni O. Grouper: a dynamic clustering interface to Web search results. In *WWW '99 Proceedings of the eighth international conference on World Wide Web* (NY 1999), Elsevier North-Holland, 1361-1374.

40 Wang J., Mo Y., Huang B., Wen J., and He L. Web Search Results Clustering Based on a Novel Suffix Tree Structure. In *Autonomic and Trusted Computing*. 2008.

41 Haralick R.M., and Kelly G.L. Pattern recognition with measurement space and spatial clustering for multiple images. In *Proceedings of the IEEE* (1969).

42 Singh S., Singh M., Apte C., and Perner P. Pattern Recognition and Image Analysis. In *Third International Conference on Advances in Pattern Recognition* (2005).

43 W.A.M., Weijermars. *Analysis of urban traffic patterns using clustering*. 2007.

44 Jain A. K., and Dubes R. C. *Algorithms for clustering data*. Prentice-Hall, NJ, 1988.

45 Estivill-Castro V., and Yang J. A Fast and Robust General Purpose Clustering Algorithm. In *Pacific Rim International Conference on Artificial Intelligence* (2000), Springer, 208-218.

46 Fraley C., and Raftery A. E. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*, 41 (1998), 578-588.

48 Kaufman L., and Rousseeuw P. Clustering by Means of Medoids. In *Statistical Data Analysis Based on the L1-Norm and Related Methods* (1987), Faculty of Mathematics and Informatics.

47 Han J., and Kamber M. *Data Mining: Concepts and Techniques*. 2001.

49 Guha S., Rastogi R., and Shim K. ROCK: A Robust Clustering Algorithm for Categorical Attributes. In *ICDE '99 Proceedings of the 15th International Conference on Data Engineering* (WA 1999), IEEE Computer Society, 512.

50 Almeida J.A.S., Barbosa L.M.S., Pais A.A.C.C., and Formosinho S.J. Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering. In *Chemometrics and Intelligent Laboratory Systems*. 2007.

51 Ng R. T., and Han J. CLARANS: A Method for Clustering Objects for Spatial Data Mining. *IEEE Transactions on Knowledge and Data Engineering*, 14, 5 (Sep. 2002), 1003-

1016.

52 Estivill-Castro Vl., and Lee I. AUTOCLUST: Automatic Clustering via Boundary Extraction for Mining Massive Point-Data Sets. In *Proceedings of the 5th International Conference on Geocomputation* (2000), 23-25.

53 Zaïane O. R., Foss A., Lee C.-H., and Wang W. On Data Clustering Analysis: Scalability, Constraints and Validation. In *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)* (2002), 28-39.

54 Morstatter F., Pfeffer J., Liu H., and Carley K. M. Is the Sample Good Enough? Comparing Data from Twitter's Streaming API with Twitter's Firehose. In *ICWSM 2013* (2013).

55 Portal, Statista - The Statistics. *http://www.statista.com/statistics/303681/twitter-users-worldwide/*.

56 BrightPlanet. *http://www.brightplanet.com/2013/06/twitter-firehose-vs-twitter-api-whats-the-difference-and-why-should-you-care/*.

58 Platform, The Foursquare. *https://developer.foursquare.com/overview/*.

57 Insider, Business. *http://www.businessinsider.com/foursquare-surpasses-45-million-registered-users-and-begins-collecting-data-in-new-ways-2-2014-1*.

59 Tarasov A., Kling F., and Pozdnoukhov A. Prediction of user location using the radiation model and social check-ins. In *UrbComp '13 Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing* (NY 2013), ACM New York.

60 Cho E., Myers S. A., and Leskovec J. User movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (NY 2011), ACM New York, 1082-1090.

61 Lv M., Chen L., and Chen G. Discovering personally semantic places from GPS trajectories. In *CIKM '12 Proceedings of the 21st ACM international conference on Information and knowledge management* (NY 2012), ACM New York, 1552-1556.

62 H., Ralambondrainy. A conceptual version of the K-means algorithm. *Pattern Recognition Letters*, 16, 11 (Nov. 1995), 1147-1157.