TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology

Marten Tall 153395IAPM

FINDING REGIONS OF INTEREST FROM GEO-TAGGED PHOTOS

Master's thesis

Supervisor: Priit Järv
Msc.

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Marten Tall 153395IAPM

HUVIPAKKUVATE REGIOONIDE LEIDMINE GEOSILDISTATUD FOTODELT

Magistritöö

Juhendaja: Priit Järv
MSc.

Tallinn 2017

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Marten Tall

Abstract

Many methods and databases for points of interest in the tourism industry have already been developed. However, points of interest are usually limited to a coordinate pair of popular objects. Regions of interest, on the other hand, give much more information, like the shape and size of the object, especially about objects that cannot be easily pinpointed, like castles or parks, and can be used to infer human activities and behaviour by their location. As a result, more accurate applications and suggestion systems can be derived.

The author proposes a robust method for finding regions of interest from geo-tagged photos. For this purpose, different clustering algorithms are analysed and the best suiting algorithm is selected. The method proposed in this thesis will be able to produce geographic areas and names of interesting objects.

The method is validated with an experiment where a random subset of regions of interest are evaluated.

A fully functional framework is developed that implements the proposed method.

This thesis is written in English and is 55 pages long, including 7 chapters, 21 figures and 4 tables.

Annotatsioon

Huvipakkuvate regioonide leidmine geosildistatud fotodelt

Mitmeid huvipunktide andmebaase ja meetodeid on turisminduses juba eelnevalt arendatud. Huvipunktid kirjeldavad tavaliselt mõne populaarse objekti asukoha koordinaate. Huvipakkuvad regioonid see-eest aga võimaldavad anda palju rohkem informatsiooni objektide kohta, nagu selle kuju ja suurus, mis võimaldab teha täpsemaid järeldusi inimeste liikumise ja käitumuse kohta, eriti suuremate objekti puhul, nagu lossid ja pargid. Selle tulemusena on võimalik arendada täpsemaid rakendusi ja soovitussüsteeme.

Töö autor pakub välja meetodi huvipakkuvate regioonide leidmiseks geosildistatud fotodest. Autor analüüsib selleks eelnevalt erinevaid klasterdamisalgoritme ning valib ühe, mille omadused sobivad enim soovitud eesmärgiga. Väljapakutud meetod suudab leida maa-ala ning nimetuse, mis objekti iseloomustab.

Pakutud meetodit valideeritakse eksperimendi käigus, kus juhuslikult valitud genereeritud huvipakkuvate regioonide korrektsust hinnatakse.

Töö käigus valmib funktsionaalne raamistik, mis implementeerib pakutud meetodit.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 55 leheküljel, 7 peatükki, 21 joonist, 4 tabelit.

List of abbreviations and terms

GPS	Global Positioning System
ROI	Region of Interest
POI	Point of Interest
LOF	Local Outlier Factor
SMoT	Stop and Moves of a Trajectory
CB-SMoT	Clustering-Based Stops and Moves of a Trajectory
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
OPTICS	Ordering Points to Identify the Clustering Structure
P-DBSCAN	Photo - Density-Based Spatial Clustering of Applications with Noise
G-DBSCAN	GPU - Density-Based Spatial Clustering of Applications with Noise
AOI	Area of Interest
MST	Minimal Spanning Tree
SOM	Self-organising map
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise

Table of contents

List of figures

List of tables

1 Introduction

Novel applications that take advantage of raw geolocation data are becoming more popular. A lot of different methods for analysing, extracting knowledge, data mining, etc., have already been developed for GPS traces. However, with the rise of ever so popular social media, it has become a rich resource for information. Among some personal data, anonymous information about a person's location can be extracted quite easily. While some applications, like Facebook, make it more difficult, others, like Panoramio give access to their data quite freely – making it able to extract meaningful data by analysing geo-tagged photos uploaded by users.

1.1 Motivation

Location data illustrates people's interests and behavior which can be used in different social, cultural, scientific, etc. applications, such as applications for recommending places to visit while travelling, restaurants to go to after checking in at a hotel, sightseeing near you etc.

Many papers, like [1] and [2] focus on finding points of interest for locating interesting sightseeing, venues, etc., and is now a well-studied problem. However, a point of interest is usually limited to a single point on a map. The author believes that this is not informative enough and can certainly be improved by finding regions of interest, instead. Region of interest is a polygon area on a map that covers a popular object, usually a sightseeing object. Finding regions of interest result in more meaningful and diverse discoveries for objects that cannot be easily pinpointed like streets, parks, bridges, castles, etc. Furthermore, regions of interest allow applications to calculate more precise proximity to large objects thereby improving suggestions for nearby sightseeing.

1.2 The Goal

The main goal of this paper is to develop a robust method for finding regions of interest using geo-tagged, time-stamped and titled photos. The proposed method will be able to extract a label – the name of the object – for each region of interest from the titles of photos. A functional framework will be implemented for the proposed method.

The author will give a thorough analysis of different clustering methods and will give reasoning to why the selected approach outperforms the others in regards of generating regions of interest.

1.3 Research methodology

This research has both qualitative and quantitative elements to it.

Qualitative explanatory methods will be used for analysing different data clustering methods. Data clustering is a data mining method which aims to group together similar objects in datasets and is most frequently used in unsupervised learning. A clustering method with best attributes will be selected as a part of the proposed method based on the study.

Quantitative research will be conducted to evaluate the method proposed by the author. An experiment will take place that measures the correctness of the regions of interest generate by the proposed method.

1.4 Thesis outline

In chapter 2, the state-of-the-art literature overview is given – current developments and directions in research.

In chapter 3, an overview of the theoretical background of different clustering methods is given and a comparative study of these methods is conducted based on the knowledge gathered by the author.

In chapter 4, the method for generating regions of interest proposed by the author is defined and explained.

In chapter 5, an experiment is conducted to evaluate the proposed method.

1.5 Summary of results

As a result of this paper, a comparative study of different clustering algorithms is constructed giving reasoning why HDBSCAN was selected as the core algorithm. A method is proposed for finding regions of interest out of geo-tagged photos and

assigning them an appropriate label. The method is evaluated with an experiment, where randomly selected regions of interest from both DBSCAN and HDBSCAN are rated.

2 Literature Overview

The subject of this study is greatly inspired by the Sightsmap [3] paper, which describes a method of creating a world heat map of touristic places and also extracting and adding semantics to the most heated areas. For the purpose of this, the authors mostly used geo-tagged photos from Panoramio – a now retired photo sharing portal. They developed an algorithm for classifying the popularity in an area (a pixel) on a map that takes into account both the number of photos and unique photographers. Furthermore, labelling the hotspots was done by finding n-grams of the titles of the photographs that exceed a certain threshold of frequency and selecting a n-gram with the most words. Due to the fact that most popular sights on Wikipedia have geo-coordinates, the authors were able to attach the spots that where nearby to the Wikipedia coordinates. Much like the Sightsmap, this paper aims at finding touristic sightseeing, but instead of extracting points of interest (POIs) and calculating the popularity of a spot, a complete framework for finding regions of interest (ROIs) is produced.

2.1 Points of interest

A point of interest is a geographic point that identifies a well-known object. Different methods for finding POIs have been derived. There are a lot of public POI databases already available today, such as OpenPOIs, OpenPoiMap, GeoNames, OpenStreetMap and many others. However, these databases merely offer the coordinates of the object, are usually not tourism oriented and do not include alternative sightseeing that is less well known, but which could still be interesting to tourists. Finding ROIs instead tries to tackle these issues.

2.2 Finding ROIs from GPS traces

Studying trajectory data from GPS traces is a common approach for finding interesting places. Commonly referred to as trajectory data mining methods, which include pattern mining, clustering, classification and knowledge discovery [4], are used for analysis.

Gianotti and others use T-Patterns to discover regions of interest by finding popular points using the notion that several trajectories of distinct people pass through a popular region. As a result, density of each spatial unit is calculated. Intuitively, any region that

exceeds the threshold density is considered to be interesting. The authors add that further complexity can be added to the problem if regression between the trajectories is required as well. [5]

2.3 Finding Stay Points from GPS traces

Khetarpaul and others address the problem by defining stay points. Stay points are locations where an object spends a threshold time *ThresTime* within a threshold distance *ThresDistance*. However, they do not take advantage of the idea of regions but use the mean of all the GPS points within the stay point instead. Interesting location is then derived from all the stay points that are visited by at least a threshold number *ThresCount* of times by unique persons. [6]

Uddin and others suggest that the proposed method of finding ROIs (or stay points) in terms of time and distance has shortcomings because if either of the parameters change, the whole dataset needs to be rescanned and that it is more intuitive to define ROIs (stay points) in terms of speed instead. The method defines an average speed range, a minimum duration of stay and the density of objects in the ROI (much like earlier definitions). An index is built on object speeds and candidate objects are filtered from the dataset that satisfy the speed and duration conditions. Lastly, dense regions [7] of objects are found by extending an existing method that defines density by the number of trajectories passing through the l-neighbourhood of the point. [8]

Chen and others show through experiments that previously mentioned Gianotti and others' method generates too large ROIs to be considered meaningful and precise enough. Calculating density takes into account number of people but does not measure the stay time in the cell, furthermore, popular ROIs are selected by the threshold density but, again, the stay time is not accounted for. They aim to remove redundant ROIs by improving the clustering of stay points by introducing the Local Outlier Factor (LOF) which finds the factors of points having lower density than their neighbours (points being outliers) [9]. Chen and others adopt previously mentioned Uddin and others' heuristics of low movement speed in ROIs and show by experiment that with the combination of his added features, more precise ROIs are indeed generated. [10]

Alternative approach for defining stay points in trajectory data is proposed by Alvares and others. It is called Stops and Moves of a Trajectory (SMoT), which proposes that for predefined candidate stops, if an object spends a threshold time in it, it is, within the trajectory, called a stop and the rest is considered to be a move [11].

However, because this method requires candidate stops to be predefined, it does not find interesting locations that are unknown. Palma and others extend the SMoT algorithm to a clustering based SMoT called CB-SMoT (Clustering-Based SMoT). CB-SMoT takes advantage of the DBSCAN clustering algorithm to find slower parts of the trajectory called potential stops. This way all non-defined interesting places can also be extracted and if it happens to intersect with a candidate stop, it can still be labelled as it would have in the original SMoT algorithm. [12]

2.4 Data clustering

All above-mentioned methods rely heavily on GPS traces and are meant for analysis of a single or a group of trajectories. This raises extensive limitations to the raw input data that can be used because in order to get meaningful results it needs to be continuous and with few errors. Liu and others explain that the quality of GPS data depends on peoples destination preferences, means of transportation (i.e. bus rides or walks might not be tracked) and even traffic conditions [13]. While time-stamped photos can be arranged into trajectories, the points of the trajectory are too sparse to give meaningful results and cannot therefore be used for our purpose. To overcome these limitations, more general approaches could be applied with data clustering algorithms.

There are very many types of clustering methods in the literature and can be categorised in very different ways. Fraley and Raftery suggested that methods could be divided into *hierarchical* and *partitioning* methods [14]. Han and Kamber further suggested that methods could be divided into three categories: *density-*, *model-* and *grid-based* methods [15].

Popular works dealing with extracting popular objects using data clustering methods take advantage of the DBSCAN, OPTICS or mean-based k-means algorithm or their variations.

K-means algorithm sets K groups of points, each containing one random center-point, after which all points are assigned to the nearest group and the center-point of the group is recalculated every time a new point is added [16]. K-means is not a very popular choice for clustering geospatial data because it does not support unclustered areas [17].

DBSCAN is a data clustering algorithm by Ester and others, which is able to discover clusters of any shape with good efficiency. The algorithm takes two parameters as input: *Eps(ε)* is the distance threshold between two points and *MinPts(m_{pts})* is the minimum number of points that a cluster can form of. [18]

OPTICS is similar to DBSCAN in the sense that they both group together close-by data points, but OPTICS is able to find clusters of varying density. OPTICS does not explicitly extract clusters, but uses augmented ordering of the database. This leads to an extended implementation of DBSCAN for an infinite number of distance values, therefore dropping the global density parameter [19]. According to Li and others, density-based clustering of OPTICS is able to produce irregular structures unlike k-means [20].

DBSCAN, being one of the most popular clustering algorithm within the geospatial analysis, many variations of it have been derived. Kisilevich and others introduced an extended DBSCAN algorithm for clustering geo-tagged photos called P-DBSCAN. It is stated that all points (photos) are not equally important: large number of densely populated points that belong to the same author should not form a cluster. In order to find densities by unique authors, a novel concept of density threshold is introduced. Density threshold denotes the number of people in a neighbourhood of a point. Furthermore, adaptive density is also introduced. Adaptive density handles clusters that have varying densities in different areas of the cluster – it is split according to its local densities – and a number of smaller, more equally dense, clusters are created. The authors show by examples that the clusters calculated by their method are indeed more robust. [21]

The main issue with P-DBSCAN and DBSCAN in general is that it needs parameter tuning. Laptev and others published an enhanced implementation that takes no parameters and is said to outperform both DBSCAN and P-DBSCAN. Furthermore, since there are no global input parameters, the algorithm is able to cluster less densely

distributed areas like parks or lakes. To achieve these properties, several modifications need to be done. Firstly, the discretisation of coordinate space, then, Gaussian density estimation together with watershed segmentation from which relevant automatic parameter selection can be done. As a result, the qualities mentioned earlier are achieved. [17]

Other recent DBSCAN developments are: G-DBSCAN which represents data as graphs in order to take advantage of parallel computation with GPUs [22] and DSets-DBSCAN which is a parameterless algorithm that combines the dominant sets and DBSCAN algorithms [23].

Closely related to this paper, is the effort of Liu and others on finding areas of interest (AOI; similar to stay points and ROIs). Liu and others also use density-based clustering on geo-tagged images crawled from Panoramio to find popular areas, but they further introduce check-ins to match AOIs with venues people might want to visit. In order to increase the precision of AOIs, secondary density is introduced, that measures the density of nearby venues to the AOIs to ensure that more dense areas are processed first. Furthermore, all AOIs are ranked by the accumulative counts of visits to and unique users at the AOI. [13]

Clustering photos suits the needs of our problem far better than trajectory analysis and will be subject to the author's proposed method, therefore an overview of common clustering methods is given in the next section.

3 Comparative study of clustering algorithms

3.1 Data clustering

Data clustering is very much like the human need to characterise objects and is therefore a subject to many different scientific practices like mathematics or even biology – the problem remains the same – assigning entities to categories. [24]

Formally, the clusters can be described as a collection of subsets where any instance in *S* belongs to a single subset:

$$C = C_1, \dots, C_k of\ S, \qquad \text{such that:}$$

$$S = \bigcap_{i=1}^{k} C_i\ and\ C_i \cap C_j = \theta\ for\ i \neq j$$

In the upcoming sections the idea behind distance measuring between objects and an overview of the most common clustering methods and their popular implementations will be given. Different clustering algorithms will be analysed and compared to give reasoning how the clustering algorithm for the method proposed in this thesis was chosen.

3.2 Distance measures

Because of the need to group together similar objects, it is important to methodologically find the similarities between different objects. These methods divide into two main types: distance and similarity measures. Most clustering methods use the distance measure rather than similarity measures. Distance between two objects *x* and *y* can be defined as $d(x, y)$. It must be both symmetric and obtain a (in most cases) zero value with identical vectors. Furthermore, the measure can be said to be metric if it satisfies:

1. Triangle inequality $d(x, z) \leq d(x, y) + d(x, z) \quad \forall x, y, z \in S$,
2. $d(x, y) = 0 \Rightarrow x = y, \ \forall x, y, z \in S$.

Commonly used distance measures includes Minkowski metric and Euclidean distance for numeric attributes, contingency table for binary attributes and simple matching or finding dissimilarities between each state for nominal attributes. [24]

Alternatively, as proposed by Duda and others, similarity functions can be applied defined as: $s(x, y)$. Similarity functions need to have following properties:

1. Is symmetrical $s(x, y) = s(y, x)$,
2. Have a large value (between allowed values) when two objects are similar,
3. Have the largest value for identical objects.

[24]

3.3 Partitioning methods

Partitioning methods begin with an initial partitioning of the clusters and then move objects between partitions until all clusters are formed. This usually requires that the number of clusters are known before execution. The next chapter will give an overview of partitioning methods. [24]

Error minimisation algorithms are the most common algorithms used for partitioning because they give good results with isolated and compact clusters. The main idea behind it is to find clusters in a way that minimises some error criterion. The leading criterion is the Sum of Squared Error (SSE), which measures the total Euclidean distance. The easiest and most popular example is the K-means algorithm. [24]

K-means starts with K cluster-centres which are either chosen randomly or using some probabilistic heuristics and then changes them in order to decrease the error function [24]. K-means is popular because of:

1. **Linear complexity -** $O(T * K * m * N)$, where T – number of iterations, K – number of clusters, m – sample size, N – number of attributes on each sample element [24].

   As stated by Dhillon and Modha [25]:

2. **Easy to understand and implement**

3. **Speed of convergence**

4. **Adaptability to sparse data**

3.4 Density-based methods

3.4.1 Method

The main idea behind density-based clustering methods is to separate clusters between sparse and dense regions [26]. Unlike hierarchical and partitioning methods, density-based methods are able to create non-spherical clusters [26]. A cluster is formed if its density of the neighbourhood exceeds a minimum limit – meaning that the neighbourhood needs to contain a minimum number of instances within a limited reach [24].

3.4.2 DBSCAN

DBSCAN is a well-known density-based clustering method. DBSCAN defines *core objects* – objects with dense neighbourhoods which are connected to form clusters. DBSCAN requires two mandatory parameters to do so:

1. $\boldsymbol{\varepsilon}$ – maximum distance an object can expand its neighbourhood. Therefore, *ε-neighbourhood* is the radius with length ε from an object.

2. ***MinPts*** – minimum density/minimum number of neighbours for a dense cluster.

[26]

A **core object** (point *x* on Figure 1) is required to have *MinPts* objects in its *ε-neighbourhood*, if the requirement is not met for the object, but it resides in the *ε-neighbourhood* of a core object, it is called a **border point** (point *y* on Figure 1). All other points that do not follow previously mentioned criteria classify as **noise points** – outliers (point *z* on Figure 1). [27]

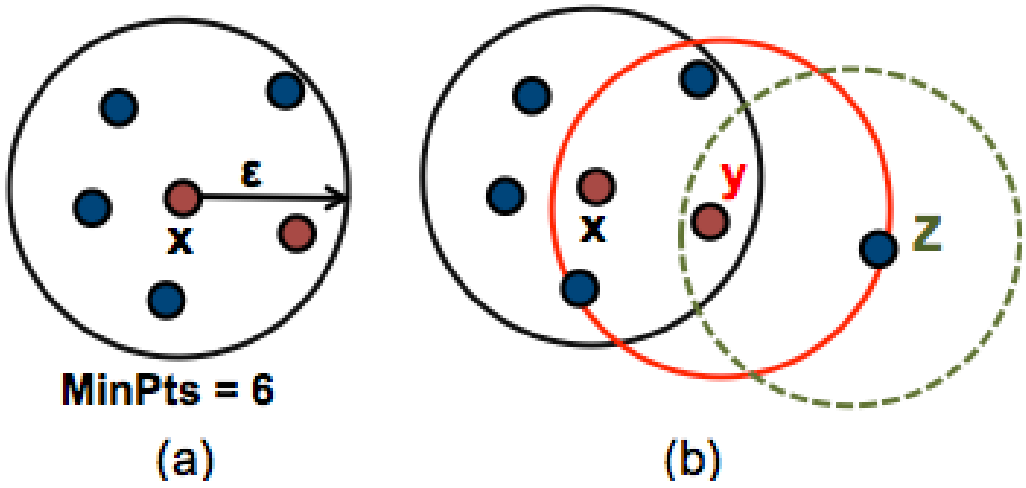

Figure 1. DBSCAN – data point classifications: (a) ε-neighbourhood of point x, (b) border point y and noise point z.

Source: [27]

By definition from Ester and others [18]: a cluster *C* is a non-empty subset of all the points if:

1. $\forall\, p, q$ that are *density-reachable*,
2. $\forall\, p, q$ that are *density connected*.

Directly density-reachable (Figure 2 a): A point *p* is directly density-reachable from point *q* if

1. Point *p* is in the *ε*-neighbourhood of point *q*;
   $p \in N_{eps}(q)$,
2. Point *q* is a core point.

Density-reachable (Figure 2 b): A point *p* is density-reachable from point *q* if there exists a chain of directly density reachable point between *q* and *p*.

Density-connected (Figure 2 c): A point *p* is density-connected to a point *q* if there exists a point *o* that is density-reachable to both *p* and *q*. [18]

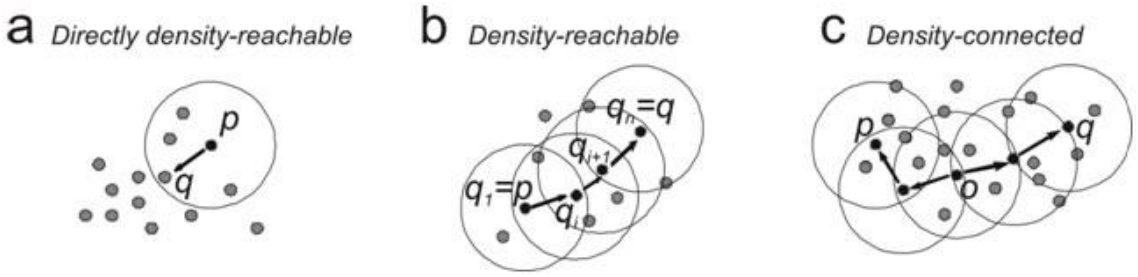

Figure 2. DBSCAN – definitions.

Source: [28]

3.5 Grid-based methods

Grid-based methods are fast algorithms [15], that divide the space into a grid and process clustering operations on each cell [24].

3.6 Graph-theoretic methods

Graph-Theoretic clustering methods generate clusters using, as the name implies, graphs where each edge of the graph connect the cluster's objects. A popular graph-theoretic algorithm was introduced by Zahn, which uses a Minimum Spanning Tree (MST) at its core. Another method, proposed by Urquhart [29], creates graphs based on limited neighbourhood sets. Graph theoretic and hierarchical methods are, to some extent, connected:

1. Hierarchical single-link clusters are subgraphs of the MST of the objects. Every subgraph consists of objects where each is connected with a minimum of one other by a similarity function threshold value.

2. Hierarchical complete-link clusters are subgraphs where each object is connected to all other objects and the set is maximal – a maximal complete subgraph.

[24]

3.7 Hierarchical methods

Hierarchical methods create clusters using recursive top-down or bottom-up partitioning methods on objects and can be classified as:

1. **Agglomerative clustering** – At first each original object forms a cluster of its own and are then merged together until needed clusters are formed.
2. **Divisive clustering** – All objects form a single cluster and is then recursively divided into sub-clusters until needed clusters are formed.

In order to get the clusters, the dendrogram created by the hierarchical method needs to be cut at some needed similarity level. [24]

According to Jain and others, merging clusters is done using similarity measures. Different approaches include:

1. **Single-link clustering** (also known as connectedness, the minimum method or the nearest neighbour method) – As defined by Sneath and Sokal, distance between two clusters is the minimum distance between any objects of each of the clusters and the similarity between two clusters is said to be the maximum similarity between any objects of each of the clusters.
2. **Complete-link clustering** (Also known as the diameter, the maximum or the furthest neighbour method) – As proposed by King, complete-link method is the reverse to single-link method – the maximum distance of all the distances between any objects of the two clusters.
3. **Average-link clustering** (also known as minimum variance method) – As derived by Ward, average-link method defines the distance between two clusters as the average of the distances between all of the objects in separate clusters.

[24]

Complete-link clustering methods usually generate better results – more compact cluster and more useful hierarchies – than average-link and single-link methods, even though the latter is considered to be more versatile. Overall, hierarchical methods can be characterised with strengths in:

1. **Versatility** – works equally well on different types of data sets,
2. **Multiple partitions** – creates multiple nested partitions.

Main disadvantages include:

1. **Scaling** – the minimum computational complexity is the number of objects squared – $O(n^2)$,
2. **No back-tracking** – methods are never able to revert what was done earlier.

[24]

3.7.1 HDBSCAN

The method proposed by this paper uses the HDBSCAN algorithm at its core. HDBSCAN was introduced by Campello and others in 2013. It is important to have an excellent understanding of the underlying algorithm. This section will give a thorough explanation of the matter.

HDBSCAN is a hierarchical clustering method that produces a complete density-based clustering hierarchy which is used for filtering out only the most significant clusters. The authors suggest that existing algorithms have many issues. DBSCAN and DENCLUE are only able to produce flat labelling instead of a hierarchical, *SkeletonClu* fails to automatically simplify its hierarchies and both *SkeletonClu* and OPTICS are only able to produce flat clusters using a global density threshold. Most importantly, most clustering methods rely heavily on user defined parameters. HDBSCAN promises to tackle all shortcomings of the currently available methods. [30]

With HDBSCAN, a modified DBSCAN algorithm called DBSCAN* is introduced. DBSCAN* extracts clusters from a graph where all adjacent vertices are data points that are *ε-reachable* [30].

ε-reachable – Two core objects x_p and x_q are ε-reachable if x_p is in the ε-neighbourhood of point x_q and vice versa, formally [30]:

$$x_p \in N_\varepsilon(x_q) \text{ and } x_q \in N_\varepsilon(x_p)$$

DBSCAN* shares many characteristics with the previously introduced algorithm DBSCAN, however, all objects that are not core-objects (as defined by DBSCAN) are considered to be noise. Also, the concept of border objects originally introduced in DBSCAN is not a part of DBSCAN* either. All data points in DBSCAN* partitions are core objects. From now on, HDBSCAN will be defined as a hierarchical relation to DBSCAN*. [30]

HDBSCAN has an input parameter m_{pts} which defines the minimum number of points a cluster can have and the number of data points considered for finding the *core distance*. Four new notions *core distance*, *ε-core distance*, *mutual reachability distance* and *mutual reachability graph* in regards to m_{pts} are defined by Campello and others to formulate the density-based hierarchy. [30]

Given a dataset $X = \{x_1, \dots, x_n\}$ and a square matrix D with size *n*, containing distances $d(x_p, x_q)$ $x_p, x_q \in X$ between all the data points in X, these definitions apply [30]:

- **Core Distance** – The core distance $d_{core}(x_p)$ is the distance from a point x_p to the m_{pts}-th closest point $x_{m_{pts}}$, including x_p [30],

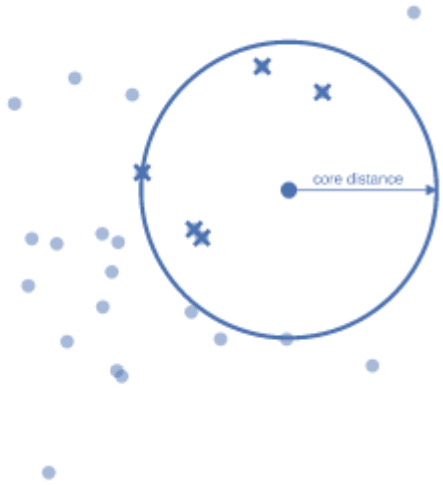

Figure 3. HDBSCAN – core distance.

Source: [31]

Figure 3 illustrates the core distance with regards to $\boldsymbol{m_{pts} = 6}$ and x_p as the centroid of blue circle (included in m_{pts}) – distance between the core object and the outermost (6^{th}) m_{pts} object.

- **ε-core Object** – A point x_p is called an ε-core object for every core distance that does not exceed ε, formally [30]:

$$d_{core}(x_p) \leq \varepsilon$$

- **Mutual Reachability Distance** – The mutual reachability distance between two points x_p and x_q is the maximum out of core-distances of x_p and x_q and the (regular) distance between x_p and x_q, formally [30]:

$$d_{mreach}(x_p, x_q) = \max\{ d_{Core}(x_p), d_{core}(x_q), d(x_p, x_q) \}$$

Figure 4 illustrates the mutual reachability distance in regards to points x_p (blue core point) and x_q (green core point), where maximum out of 3 distances is $\boldsymbol{d_{core}(x_q)}$ (The green core distance).

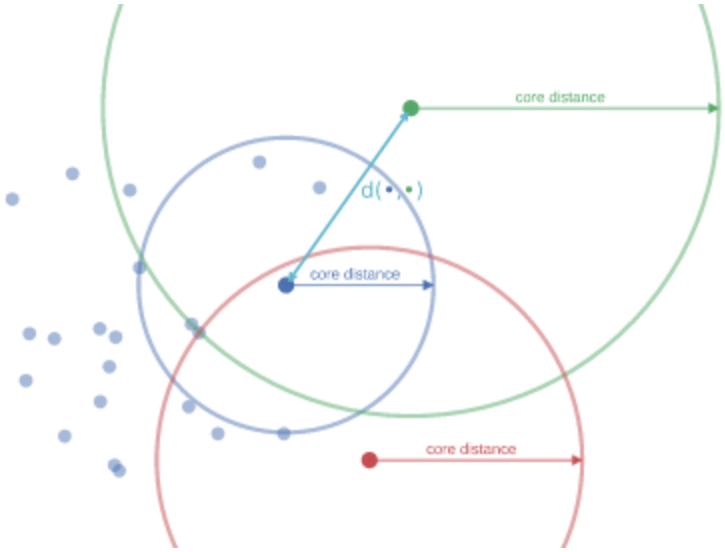

Figure 4. HDBSCAN – Mutual Reachability Distance.

Source: [31]

- ***Mutual Reachability Graph*** – Mutual reachability graph $G_{m_{pts}}$ is a complete weighted graph where all the vertices are the points in *X* and the edges between corresponding vertices are the mutual reachability distances [30].

The clusters can be extracted from $G_{m_{pts}}$ by removing all edges with weights that exceed ε and labelling them as noise – we are now left with a subgraph $G_{m_{pts,\varepsilon}} \subseteq G_{m_{pts}}$. This process can be repeated many times for values $\varepsilon \in [0, \infty]$ for achieving nested, hierarchical partitions. [30]

In order to merge partitions, a new method is proposed by Campello and others. They argue that instead of running *Simple-Linkage* over the mutual reachability graph, it would be more efficient to use an extended *Minimum Spanning Tree* of the graph where each vertex *o* is connected to itself with an edge (of the core distance of *o*). HDBSCAN uses an implementation of Prim's algorithms on ordinary list search for creating the MST. [30]

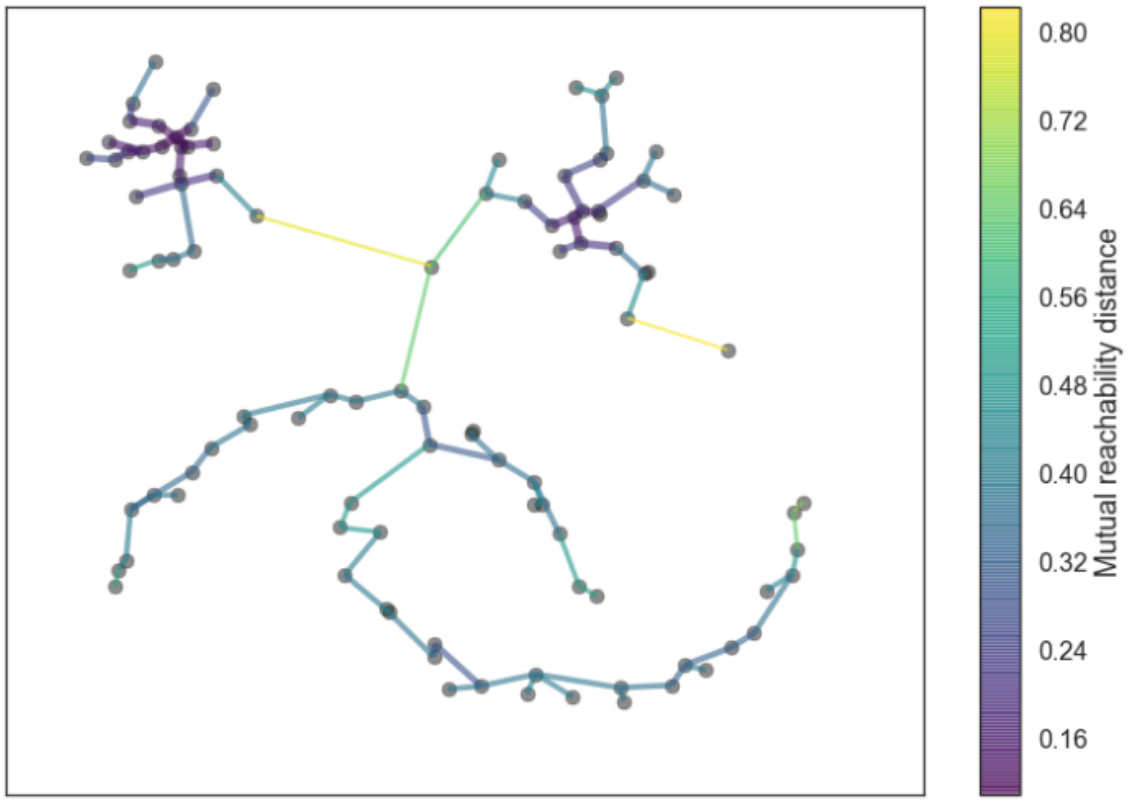

Figure 5. HDBSCAN – Minimal Spanning Tree with regards to mutual reachability distance. Source: [31]

To finally create the HDBSCAN hierarchy (dendrogram), the tree needs to be iterated, decreasing threshold weight between vertexes each time. As a result of each iteration, new hierarchy level is created containing different connected partitions of the data, which are formed by disconnecting vertexes that do not exceed the threshold weight. Hierarchies of clusters starting from the largest single cluster up to large amounts of the smallest possible clusters are formed. [30]

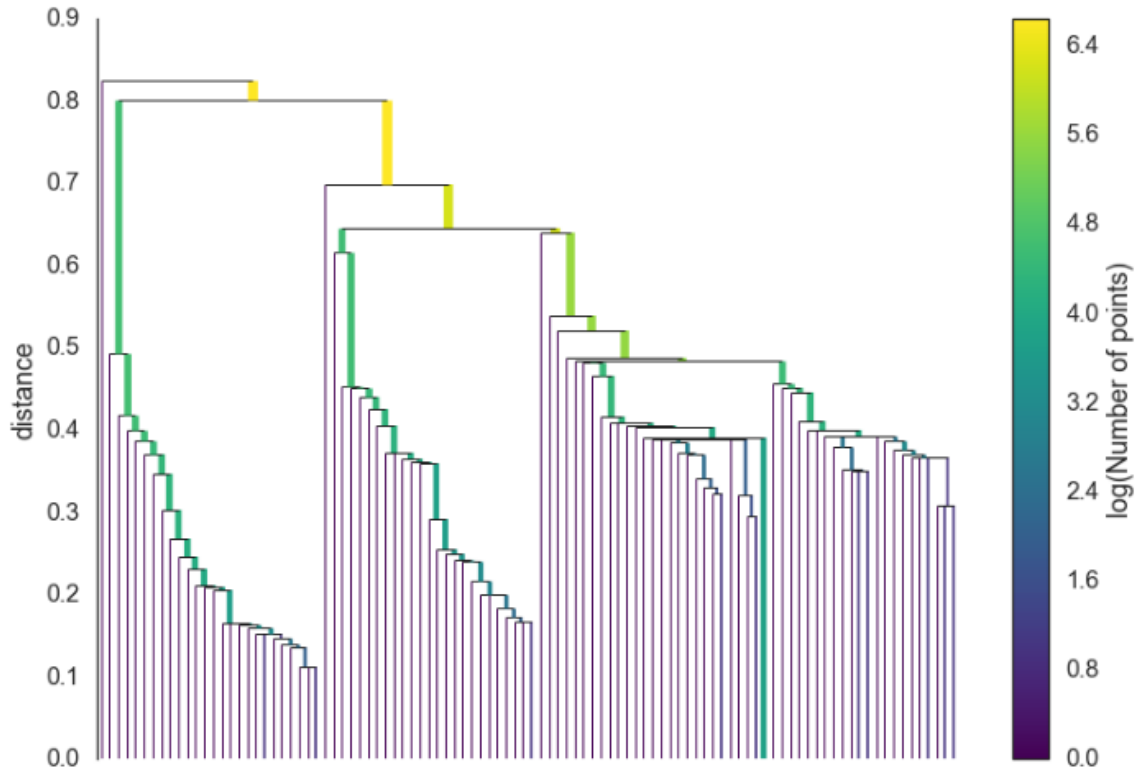

Figure 6. HDBSCAN – dendrogram illustrating a HDBSCAN hierarchy.

Source: [31]

The computational and space complexity of creating the MST depends on the available input. If it is given the dataset X**,** it can run in $O(dn^2)$, where d is the number of data attributes, and requires $O(dn)$ space. However, if the distance matrix D is given, the computational complexity reduces to $O(n^2)$, but the space complexity increases to $O(n^2)$. [30]

In practice, it is difficult to work with hierarchies that contain large amounts of data. To tackle that problem, Campello and others propose a method for simplifying the HDSBCSAN hierarchy. The general idea behind it is setting a minimum cluster size m_{clSize}, so that all clusters that have less data points than m_{clSize} are disregarded as noise. This can greatly reduce the size of the hierarchy. Usually, $m_{clSize} = m_{pts}$, so a single input parameter is retained, but if needed, it can be set independently as well. [30]

To extract a flat, single layered representation of the most significant clusters in the HDBSCAN hierarchy, Campello and others propose a globally optimal solution to this problem. McInnes and others explain that to extract flat clusters, we need to pick

partitions with long lifespan. For this purpose we need to find each cluster's *stability*, formally:

$$\sum_{p \in cluster} (\lambda_p - \lambda_{birth})$$

Where p is the data point in a cluster, $\lambda = \frac{1}{distance}$; λ_{birth} – λ value when the cluster with point p split off from a larger cluster and formed a cluster on its own; λ_p – λ value when the point p was removed from the cluster. Moving from bottom of the tree to the root of tree, we need to compare the stability of the cluster to the sum of the stabilities of its children. If the latter is greater than the stability of the parent node, the parent node gets assigned the sum, otherwise, the cluster gets selected and all descendants are declared unselected. [31]

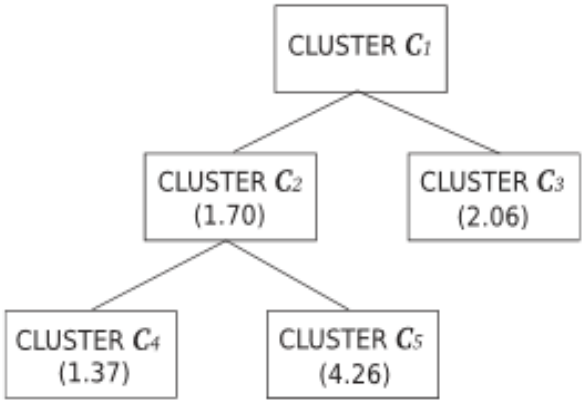

Figure 7. HDBSCAN – cluster hierarchy with cluster *stability*.
Source: [32]

Figure 7 is an example of a HDBSCAN cluster hierarchy C where each node has its stability calculated. When applying previously mentioned method to this tree, cluster C_2 would be assigned a new stability $stability(C_4) + stability(C_5)$, because the sum of the children is greater than the parent node's. Cluster C_3 would get selected, because its stability exceeds children's (in this case there are no children).

After traversing the tree, we have arrived at the flat representation of the extracted clusters. Figure 8 displays the clusters with separate colours generated in this example.

Figure 8. HDBSCAN – generated clusters.

Source: [31]

3.8 Comparative study of clustering algorithms

This section will present reasoning behind choosing HDBSCAN as the core algorithm for ROI generation. There are very many different clustering methods available, however, grid-based methods are disregarded by the author, because the cluster shapes are limited to grid cells, which does not suit our purpose. Three common clustering methods where considered: partitioning-based, density-based and hierarchical methods. Main disadvantages of each method will be brought out by concrete implementations of each: partitioning algorithm k-means, density-based DBSCAN and OPTICS and finally a novel hierarchical clustering algorithm HDBSCAN. For each, shortcomings of the subject methods will be brought out and illustrated. Ways how HDBSCAN tackles these issues and why it was chosen will be expressed.

3.8.1 K-Means

K-means is one of the most mature and popular clustering algorithms available. It is in most cases a fast algorithm, having time complexity of $O(I * K * N * M)$, where: I – number of iterations, M – vector distance computation and K*N is the number of distances computed, it is easy to spot that all parameters have linear relationship [33].

There are several reasons why k-means is not ideal for generating ROIs out of spatial data. The most obvious being the fact that the number of clusters need to be known beforehand. While this problem can be solved for a concrete dataset (the town of Budapest) by parameter tuning, it is out of question for the whole of the world or even a country – it can never be generalised. While there do exist methods for optimising K value, one of which was proposed by Hamerly and Elkan [34], but it adds complexity and does not eliminate other problems.

The second issue with k-means, as pointed out by Raviya and Dhinoja [35], is that k-means is sensitive to outlier data points – noise. This means that k-means is not able to find and filter outlier points. Every data point is always assigned to a (single) cluster, which creates unwanted effects in our use case – photos that should be excluded from ROIs get assigned anyway.

Another known issue with k-means is that it does not work well with complex shapes (clusters) that are not spherical in nature. Sightseeing objects, however, tend to vary a lot in the matter of shape and size which suggests that k-means is not suitable for our purpose. This and the previous issue are illustrated by Figure 9.

Figure 9. Regions of interest generated by K-Means, with regards to k=1000.

With HDBSCAN the number of clusters do not need to be known before execution, instead, the minimum number of points m_{pts} in each cluster needs to specified. In my opinion, it is far more intuitive and far less sensitive in regards of the final result. It is

also able to detect noisy data points and remove them from the final result. Furthermore, clusters' shapes and sizes are not in any way limited – an important condition for correct and meaningful ROIs.

3.8.2 DBSCAN

DBSCAN improves k-means in the sense that the number of generated clusters is varying and therefore do not need to be specified before execution. Also, the clusters can have non-spherical shapes [26]. This comes at a cost – the average time complexity is $O(n * logn)$ and the worst case complexity is $O(n^2)$ [18].

DBSCAN depends on parameters m_{pts} and ε and the end result is highly sensitive to these two, but especially to ε. Similarly with k-means, with some experimental parameter tuning, meaningful ROIs with densely distributed data points can be extracted from a minimal dataset, but a global configuration is not feasible.

The main issue with DBSCAN is that having clusters with different densities is not possible [35]. In our example most popular and densely populated ROIs are found without any problems, but more sparse areas are completely ignored. This does not suit our needs well because data points on squares, parks, lakes, etc. tend to be more scattered.

By experimenting with different m_{pts} and ε parameter values, I was able to get visually best results with $m_{pts} = 15$ and $\varepsilon = 0.0001$(m). Figure 10 perfectly illustrates how DBSCAN is able to pick up only the densest ROIs.

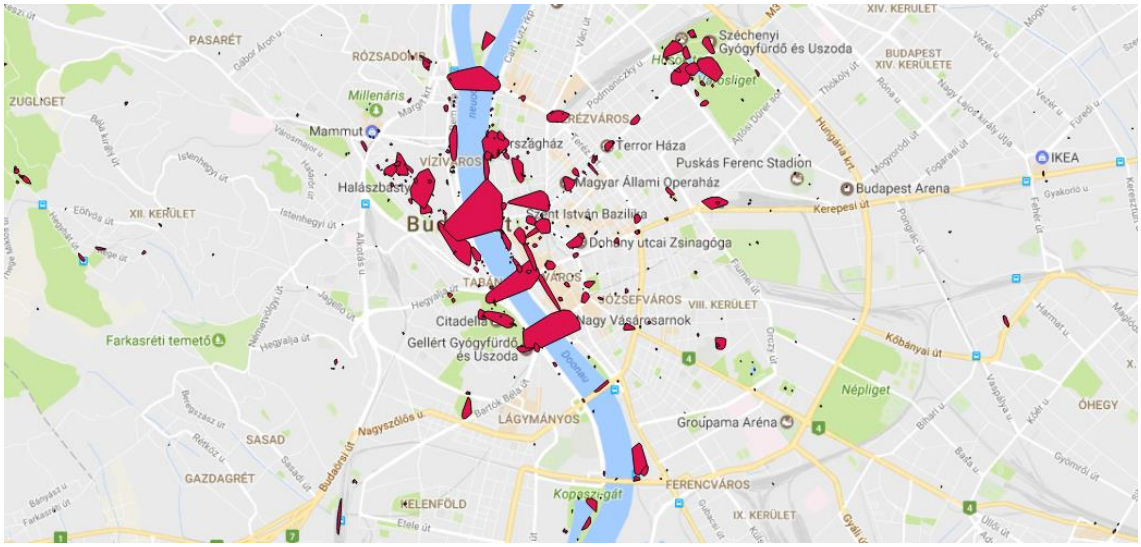

Figure 10. Regions of interest generated by DBSCAN, with regards to $m_{pts} = 15$ and $\varepsilon = 0.0001$.

HDBSCAN is superior to DBSCAN in the way that it does not require a global ε value, a hierarchy with $\varepsilon \in [0, \infty]$ is built instead and flat clusters carefully selected from the hierarchy [32]. Furthermore, hierarchical clustering enables HDBSCAN to generate clusters to more sparse areas as well.

3.8.3 OPTICS

OPTICS has the same input parameters as DBSCAN – m_{pts} and ε, but it is possible to reduce parameters to one by setting $\varepsilon = \infty$. This is a real improvement in regards to parameter selection. OPTICS runs with $O(n^2)$ runtime complexity, however, with spatial indexing, the complexity is reduced to $O(n * logn)$ [19].

Campello and others point out that OPTICS uses a *global cut threshold* when creating flat clusters out if its hierarchy, which may not result in most significant partitions. Inspired by this limitation, HDBSCAN eliminates it by introducing a method that uses *local cut threshold* instead. [30]

Furthermore, OPTICS has performance issues – the program execution time is far longer than all of the other clustering algorithm execution times as can be seen in Figure 12.

3.8.4 HDBSCAN

HDBSCAN exceeds all previously mentioned methods in several ways that suit the needs and goal of this thesis. To give a comparison, Figure 11 demonstrates the clusters generated by HDBSCAN.

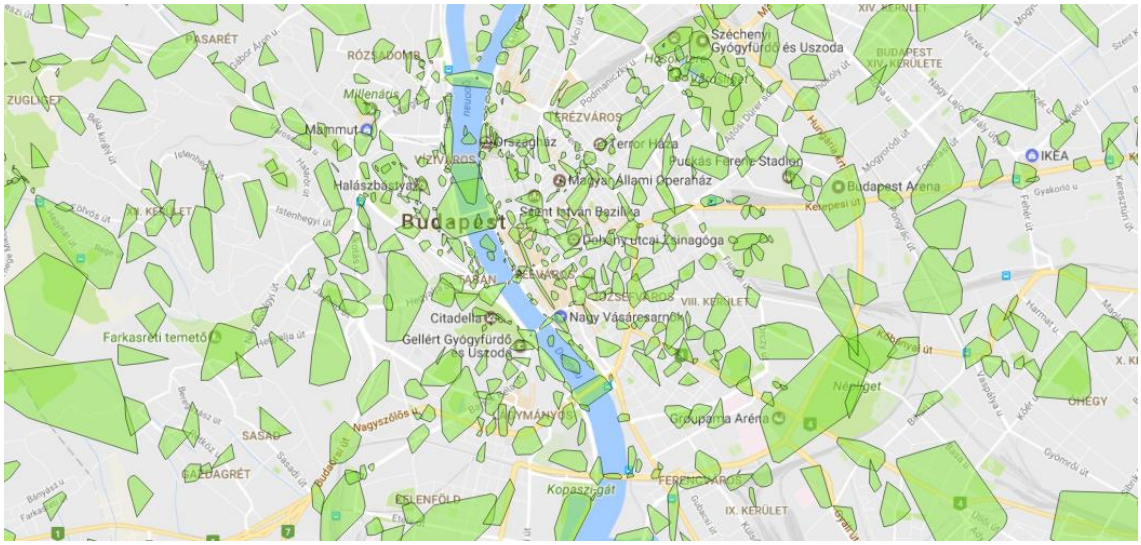

Figure 11. Regions of interest generated by HDBSCAN, with regards to $m_{pts} = 15$.

Only drawback that HDBSCAN poses is high run time complexity, which can reach up to $O(dn^2)$. This has a significant effect on the time the clustering takes, but is unfortunately inevitable as with higher precision, often more complex calculations are needed. Keeping that in mind, for our purpose, HDBSCAN is efficient enough. For larger datasets, data could be divided into grids and parsed separateley and in parallel. Figure 12 demonstrates the average time each algorithm takes on our sample dataset with regards to $n = 177498$.

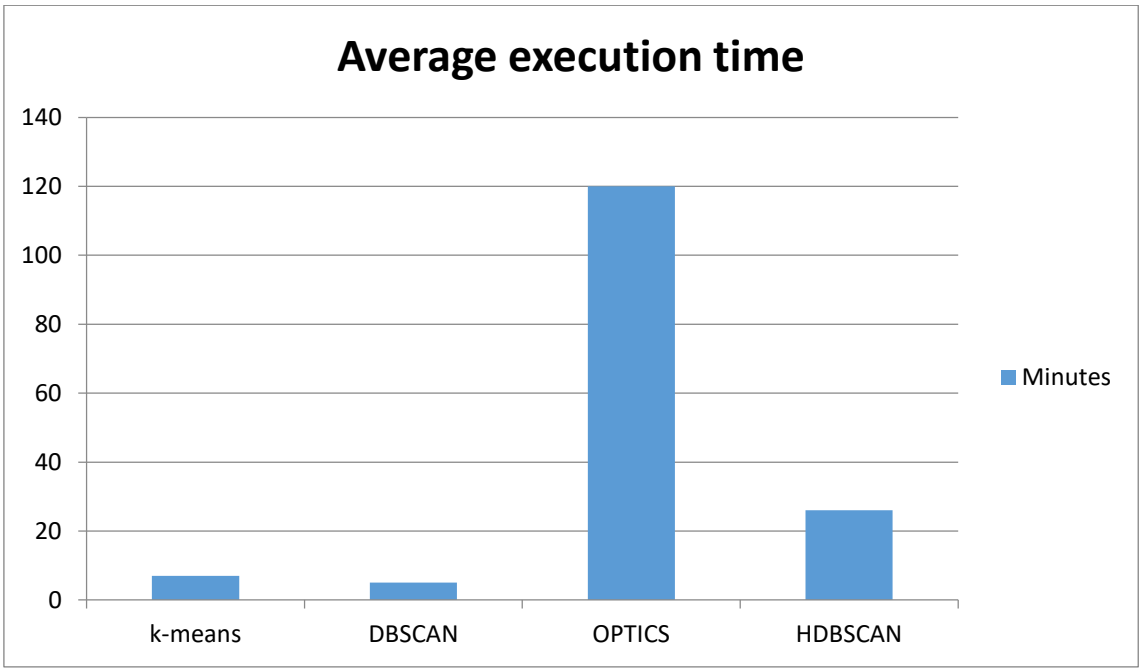

Figure 12. Execution time of different clustering methods.

4 Method

In this section the author proposes a novel method for generating ROIs and assigning an appropriate label to it. The method consists of three main parts:

1. Cluster generation,
2. Label extraction,
3. Cluster merging.

Each part improves the ROI generation accordingly. They will be discussed in the upcoming subsections. The final subsection contains the pseudocode for this method for reference.

4.1 Cluster generation

The core data clustering algorithm used for generating clusters from photos is HDBSCAN. The Reasoning behind this choice was explained in section Comparative study of clustering algorithms earlier and will not be discussed further.

HDBSCAN takes a single input parameter m_{pts}, which denotes the minimum number of points each cluster can have and is also used for finding the reach of the cluster with regards to m_{pts}. Even though the m_{pts} parameter is not as sensitive as the parameters in alternative methods, the value should still be chosen carefully because otherwise the clusters generated might be undesirable. In the proposed method, the parameter is assigned a value

$$m_{pts} = 15,$$

which has proven (by experiment) to give optimal results. Any cluster with fewer points than 15 is just not densely populated enough to be considered popular and restricting ROIs to much higher populations than 15 is too limiting – many of the desired clusters are not formed.

It is demonstrated in Figure 13 that too large values ($m_{pts} = 100$) result in bloated ROIs. In the image, pink areas are ROIs with $m_{pts} = 100$ and green ROIs are the ones proposed by the author. Some examples include the ROIs in the South-East corner of the image.

Figure 13. Regions of interest generated by HDBSCAN, with regards to $m_{pts} = 100$.

Figure 14 shows the opposite effect of a too little m_{pts} value, where most of the ROIs are too splintered across map. Red areas are HDBSCAN clusters with regards to $m_{pts} = 5$ and green clusters with regards to $m_{pts} = 15$ as proposed by the author.

Figure 14. Regions of interest generated by HDBSCAN, with regards to $m_{pts} = 5$.

In the proposed method, a convex hull of each of the generated cluster is calculated and expanded by 1.5 times the length from the centroid to the edge. Each point $p(x, y)$ in the convex polygon representation of the cluster are shifted to a point $\bar{p}(\bar{x}, \bar{y})$ using formulas: :

$$x \to \bar{x} = x + \frac{(x - x_c)}{d_{xy}} * \frac{1}{2} d_{xy}$$

$$y \to \bar{y} = y + \frac{(y - y_c)}{d_{xy}} * \frac{1}{2} d_{xy}$$

In the formula, x_c, y_c represent the coordinates of the centroid point p_c of the polygon and d_{xy} is the Euclidean distance between p_c and the original point p.

$$d_{xy} = \sqrt{(x - x_c)^2 + (y - y_c)^2}$$

The expansion is illustrated by Figure 15, where dark green clusters are the original and purple clusters are the extended clusters. As can be seen, the expanded clusters keep the original shape.

Figure 15. Regions of interest – extended.

The expanded clusters are then added to the the initial set of ROIs.

4.2 Label extraction

It is difficult to achieve any practical value for ROIs if they are not in any way semantically enriched. This section we will propose a method for extracting and assigning labels to each ROI generated in the previous part.

Firstly, since tourists tend to be from countries with a language other than the language used in the country the are visiting, the photo titles are also quite often in very many languages. To reduce the data variance in the initial dataset, the photo titles can be translated. However, it is an optional part of the method. The framework implemented by the author uses Yandex Translation service because it offers the most number of free translations.

Secondly, before the actual label generation can begin, all the photo titles need to be stripped of all the special characters, like punctuation marks, digits and conjuctions, like “a”, “the”, etc. in the English language. This helps to reduce differences between titles that are purely semantic. For example, similarities between titles “Budapest, Hungary!” and “Budapest Hungary at 15:55, 01.01.2012”, which clearly describe the same entity, would otherwise be missed.

When data is prepared, all the titles are iterated and n-grams of all the titles are parsed with the limitation of *n* being between $[2,5]$. For example, all the n-grams of a title “budapest river danube” would be:

- “budapest river” ($n = 2$),
- “river danube” ($n = 2$),
- “budapest river danube” ($n = 3$).

This helps to extract significant words (in a way explained in the next paragraph) from over descriptive titles like “This photo was taken on budapest river danube”, which include, in this context, unimportant words.

All n-grams of all the titles are then counted and the most frequently occuring n-gram is selected as the title of the ROI. In case of a tie, the n-gram with most words (the highest *n* value) among the most popular n-grams is selected. If the data includes a person or a trajectory id, then only unique titles of the person/trajectory are counted to lessen the impact in cases when a person has uploaded many images with the same title.

4.3 Cluster merging

Some of the ROIs generated have the same title. If these clusters appear relatively close to each other, there is a good chance that they represent the same real life entity. The author states that these clusters should be merged together into a single cluster, we shall call it *exact match merging*. Figure 16 illustrates this phenomenon.

Figure 16. Merging clusters together – exact match merging.

4.3.1 Exact match merging

With exact match merging, all clusters with the exact same label should be aggregated and collected. Each aggregate is parsed individually by iterating all the clusters in it. A pivot cluster is selected and its distance from other clusters is computed. If any clusters are within 50m to the pivot cluster, all points of the selected clusters are merged into a single one. As demonstrated in Figure 16, clusters with the same name can be situated sequantially and all clusters that need to be merged, do not get merged with the first iteration, so the merging is recursive in nature.

For example, if the blue cluster in Figure 16 was selected as the initial pivot, it would merge with 2 purple clusters nearby, but not with any other. If the previously merged cluster was set as the pivot, it would further merge with orange clusters and finally merge with the pink cluster and a single green cluster is formed.

4.3.2 Fuzzy match merging

Some clusters are named slightly differently, but are semantically the same. Previously introduced exact match merging does not take these examples into account. To overcome this shortcoming, *fuzzy match merging* is introduced by the author.

Fuzzy match merging matches clusters that are geographically close to each other (within 50m) and have **similar** titles. In order to do so, all clusters are iterated and with

each iteration a pivot cluster is selected. The pivot cluster's title is compared against all remaining clusters' titles using Jaro-Winkler's string similarity algorithm [36]. A cluster is considered to be a fuzzy match if the distance d_{JW} is:

$$d_{JW} \geq 0.9, d_{JW} \in [0,1]$$

It is important to note that unlike exact match merging, fuzzy match merging does not occur recursively, because the mutation of the titles in the tree of matches can become too large – each tree node allows a minimal mutation but combining *n* mutations could lead to undesirable merging of the clusters.

Jaro-Winkler algorithm was chosen because of its several desired properties for this use case. Firstly, as tested by Cohen and others [37], Jaro-Winkler algorithm is a top-tier algorithm when it comes to its runtime performance, which is important because fuzzy match merging is computationally complex. Secondly, Jaro-Winkler algorithm uses a prefix scale, meaning that strings that match in the beginning get a higher similarity value [38]. This is helpful in examples such as:

- "Váci u." – "Váci utca" – 90%,

but are also helpful finding typos like:

- "károl mihály" – "károlyi mihály" – 92%.

Finally, Jaro-Winkler is a mature algorithm with many out-of-the-box thoroughly tested implementations available, making it easy to integrate it into the framework.

If two clusters are subject to merging, then the question which title to choose arises (since they are not exactly the same). The author resolves this by selecting the title that occurs most frequently within all the generated ROIs. If there happens to be a tie, then the longest title is chosen for the merged cluster.

4.4 Pseudocode

Cluster generation

```
1. var points = READ data points
2. var clusterArr[ ] = HDBSCAN.cluster(points)
3. clusterArr = expandClusters(convex(clusterArr))
```

Label extraction

```
4. FOR cluster IN clusterArr:
5.    var strippedPoints = stripPointTitles(cluster.points)
6.    var bagOfWords{};

7.    FOR point IN strippedPoints:
8.          var ngrams[] = generateNGrams(point.title)
9.          bagOfWords.put(<point.title, point.personId>)

10.   cluster.label = bagOfWords.maxCountWord()
11.   SAVE cluster
```

Exact match merging

```
12. var aggregateArr = READ clusters aggregated by label
13. FOR aggregate IN aggregateArr
14. FOR clusterA IN aggregate:
15.   FOR clusterB IN aggregate:
16.         IF geoDistance(clusterA, clusterB) <= 50
17.               var clusterC = merge(clusterA, clusterB)
18.               DELETE clusterA, clusterB
19.               SAVE clusterC
20.               UPDATE aggregate(clusterC)
21.               JUMP 14.

22. FOR cluster IN aggregate
23.   SAVE cluster
```

Fuzzy match merging

```
24. var clusterArr = READ all clusters
25. FOR clusterA IN clusterArr:
26.   FOR clusterB IN clusterArr:
27.         var titleA = clusterA.title
28.         var titleB = clusterB.title
27.         IF jaroWinklerDistance(titleA,titleB) >= 0.9
28.               IF geoDistance(clusterA, clusterB) <= 50
29.                     var clusterC = fuzzyMerge(clusterA, clusterB)
30.                     DELETE clusterA, clusterB
31.                     SAVE clusterC
```

5 Experimental Setup

In order to evaluate and verify the results with the proposed methods, several experiments will be conducted. This experiment will test the quantatitive attributes of the method proposed in this thesis.

5.1 Main goal

In this experiment we will mainly focus on measuring the accuracy of the generated ROIs, which is the main contribution in this paper. The ROIs will be evaluated in regards to the position of the ROI in the geographical space and whether the title that has been assigned to it is correct: describes the ROI in a way that clearly identifies it.

5.2 Ground truth

Since this is an unresolved problem that is being developed rapidly in research and also because entities in the world are changing very fast, the ground truth does not really exist for this experiment. Furthermore, this approach is in theory able to find interesting places that are not yet identified by existing systems, diminishing the quality of these even further. However, in this experiment, Google Maps will be used to identify whether ROIs are correct or not, since it is most widely used and is able to give information with, in most cases, good quality. Furthermore, QGIS, an open source geographic information system, will be used for visualising Google Maps as the base layer and all the generated ROIs as the main data layer. All ROIs will be evaluated manually using methods described in the ‘Evaluation method’ section.

5.3 Data

The data used for this paper is scraped from a photo sharing social media site, previously popular among tourists, called Panoramio. The dataset includes 177498 unique geo-tagged high-resolution photos taken in Budapest, Hungary in JSON format and 98% of the photos are titled.

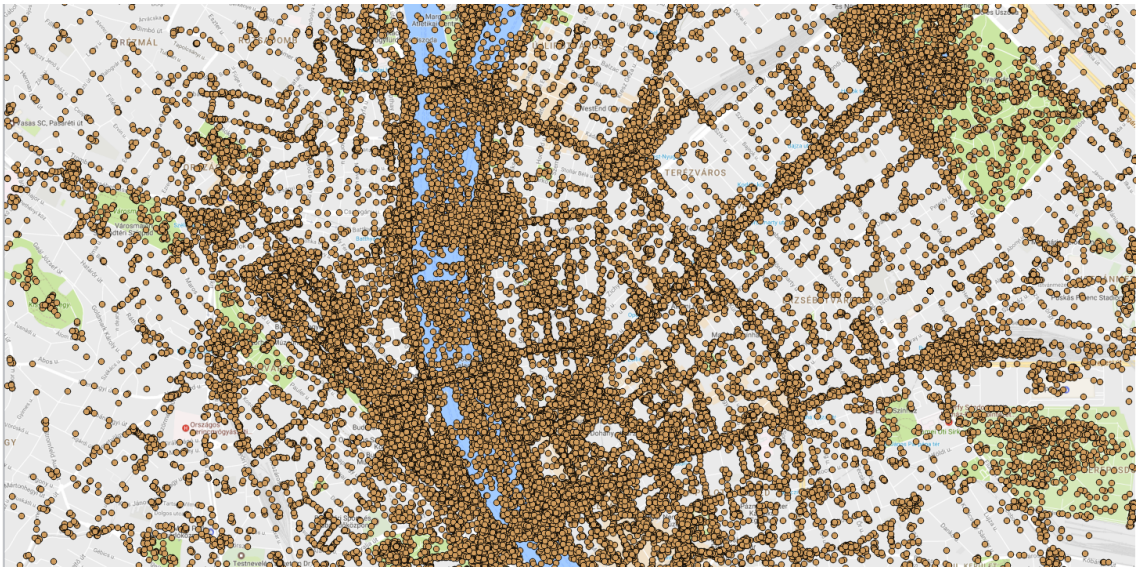

Figure 17. Panoramio's geo-tagged photos.

The quality of the titles varies greatly: while some titles are named precisely, like 'Parliament' or 'Buda Castle', some are too vague, like 'Hungary' or 'Budapest' or the combination of two, others fail to give any meaning at all, like a date of the image or the photographer's name. Each data instance is also annotated with the id of the person who uploaded it. Data can be described as a vector: <*point*, *title*, *personId*> where point *p* is is a coordinate pair in a geograpical space:

$$p(x_1, x_2)$$

5.4 Evaluation method

Because of the lack of an automatic way to evaluate generated ROIs, a clear set of rules will be established in this section. These rules allow evaluation to stay as objective as possible throughout the experiment.

A. ROI is correct when the convex area of the ROI fully covers an object (Figure 18).

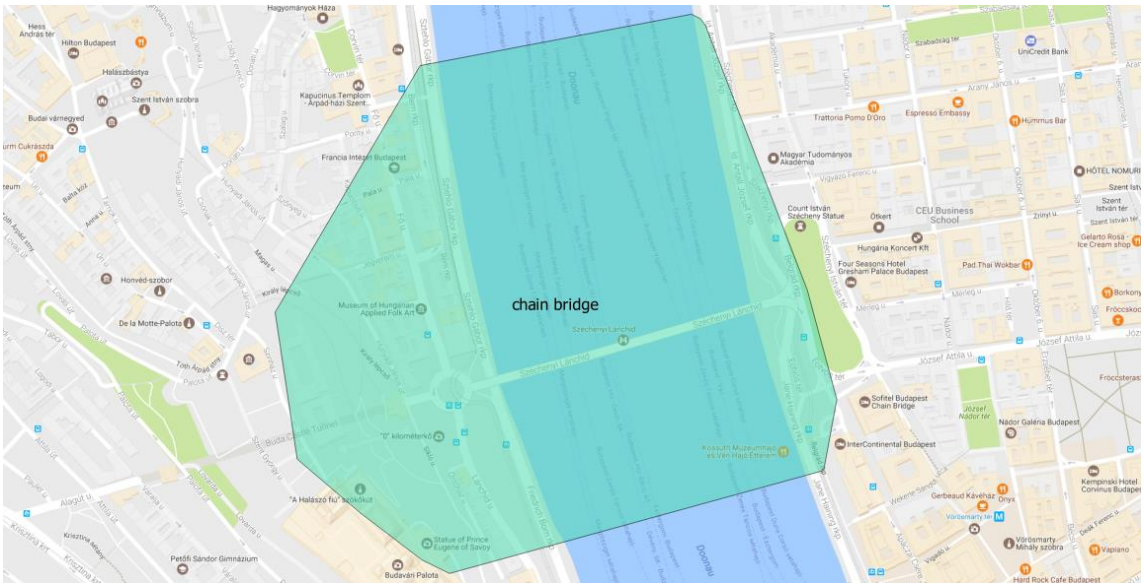

Figure 18. Region of interest – fully covered.

B. ROI is correct when it partially covers an object (Figure 19).

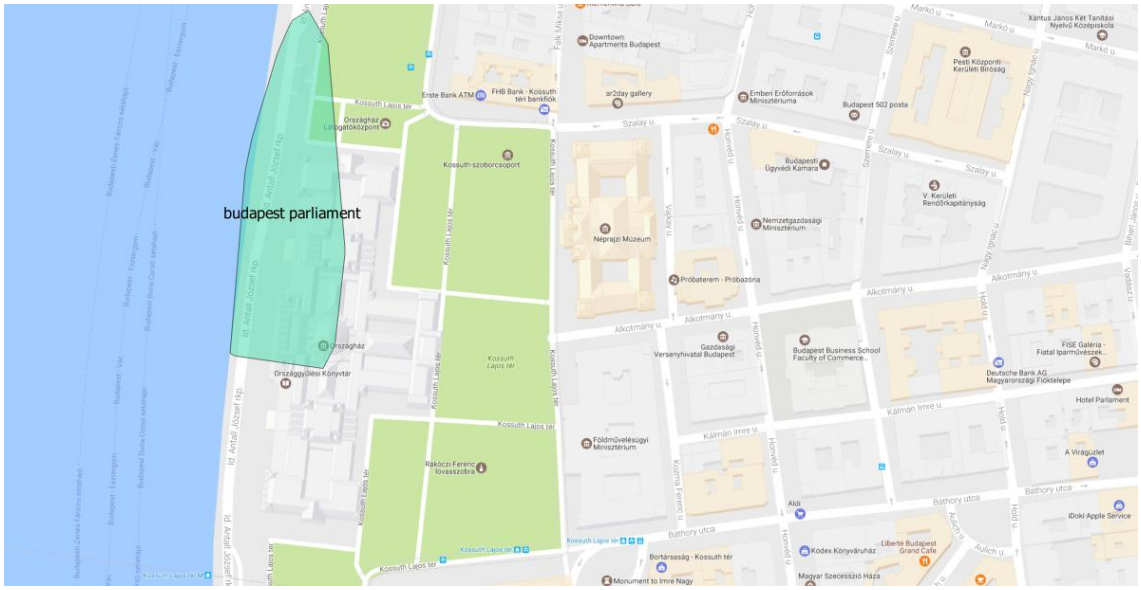

Figure 19. Region of interest – partially covered.

C. ROI is correct when it is located very near to the object itself (Figure 20), i.e. in front of a building.

Figure 20. Region of interest – near object.

D. ROI is considered to be a spot for the observation of the object from distance if the probability of the accuracy of the label is high. The probability is considered high if the count γ of similar labels in the ROI is equal to or exceeds 3: $\gamma \geq 3$ or if the spot is within reasonable viewing distance and environment. Figure 21 illustrates how a ROI can cover the object itself, but a popular spot for viewing from distance over the lake is generated as well.

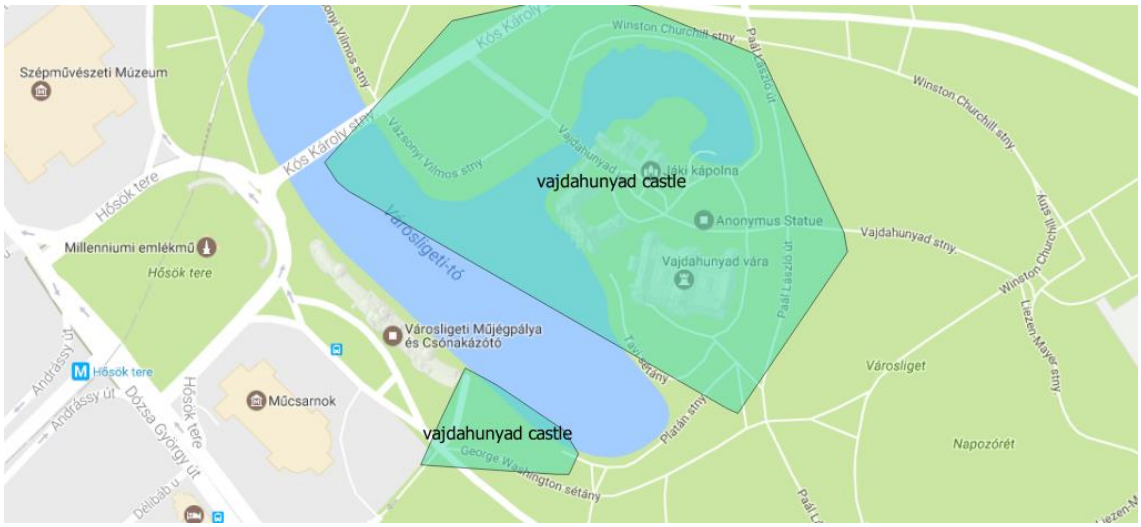

Figure 21. Region of interest – from distance.

E. A label is considered to be correct if it clearly expresses the object the ROI covers by rules 1–4. Examples include Figure 18, Figure 19, Figure 20 and Figure 21.

F. A label is considered to be correct with a translation error if the original – not translated label – clearly indicates the object under examination, but the

translated version has lost its meaning due to a false translation or a word-to-word translation. Examples:

1. “big sure” – “Nagytarcsa”
2. “matthias earth” – “Mátyásföldi”
3. “original air” – “Káposztásmegyeri”

G. A ROI and its label are considered to be both incorrect if neither of the rules 1–6 apply.

5.5 Results

Table 1 lists coded numeric values for classification of all the evaluated ROIs tied to its meaning and connection with evaluation rules.

Table 1. Region of interest evaluation classification.

Type	Meaning	Rules
0	ROI misplaced or in any other way not meaningful; Label is false or missing.	∅
1	ROI misplaced, but is a popular spot for observation of the object.	D,E
2	ROI is correct; Label false or missing	A,B,C
3	**ROI is correct; Label is correct**	**A,B,C,E**
4	ROI is correct; Original label is correct, final translation distorts the meaning – is false.	A,B,C,F

In upcoming subsections results of both DBSCAN and HDBSCAN evaluations are given. All of the ROIs were randomly chosen using PostgreSQL built-in function *random()*.

5.5.1 DBSCAN

In total 694 ROIs where generated using the DBSCAN algorithm. 100 ROIs where evaluated using translated data, making the evaluation rate 14.4%. Table 2 illustrates the results.

Table 2. DBSCAN results, with regards to n=100.

Type	**Count**	**%**
0	38	38%
1	7	7%
2	9	9%
3	**43**	**43%**
4	3	3%

43% of the ROIs where extracted and labelled correctly (type 3), in total 50% of the ROIs can be considered correct if popular observation points (type 1) are considered correct as well. The main reason for such a high percentage of incorrect ROIs (type 0), is that 46.5% of labels generated for DBSCAN ROIs have less than 3 occurrences of the same label, making it highly likely to be out of context.

5.5.2 HDBSCAN

In total 150 ROIs where evaluated with translated data out of total 1257, making the evaluation rate 11.9%.

In the **first** iteration 100 ROIs where chosen, the results are in Table 3:

Table 3. HDBSCAN results, with regards to n=100.

Type	**Count**	**%**
0	11	11%
1	7	7%
2	8	8%
3	**60**	**60%**
4	14	14%

In the **second** iteration **150 ROIs** where chosen, the results are in Table 4:

Table 4. HDBSCAN results, with regards to n=150.

Type	**Count**	**%**
0	19	12.7%
1	11	7.3%
2	10	6.7%
3	**93**	**62.0%**
4	17	11.3%

Because of very little change in the results between two datasets we can assume 150 ROIs is enough to make objective conclusions from it. That said, we can say that roughly 60%–62% of all ROIs are generated correctly (type 3). Furthermore, taking into account popular observation spots (type 1), which can also be considered to be of importance, roughly 67%–69% percent of ROIs are indeed correct.

Some of the ROIs have labels assigned that have a count of less than 3. When the ROIs that do not exceed the confidence threshold are removed from the final set, the percentage of correct ROIs rises to 67.7% and the percentage of popular observation spots stays almost the same at 6.7%. In total we can say that the overall percantage of correct ROIs would be 74.4%. However, in the test set, 30% of ROIs have labels with count less than 3, meaning that almost a third of ROIs would be removed from the total set while only gaining 5% in correctness.

It is clear from the results that HDBSCAN is superior to DBSCAN in its performance: it is able to generate twice as many ROIs while greatly improving the accuracy of the ROIs labelling using the method proposed by the author of this paper.

6 Conclusions

With the rapid growth of GPS enabled devices and the number of geo-tagged photos uploaded to social media every day, mining data from these sources is becoming more relevant than ever and can be used for different novel applications like tour guides for tourists or recommending alternative sightseeing near you.

There already exist plenty of methods and databases for points of interest. However, a point of interest does not give much information about the object it represents. Regions of interest, on the other hand, are able to give a lot more context: for example, the shape and the size of the popular object. Applications can use this data to make more precise calculations of nearby sightseeing resulting in more satisfactory user experience.

A method for generating regions of interest from geo-tagged photos was proposed in this paper. A comparative study of different clustering methods and their popular implementations was conducted and the clustering algorithm with the best properties was chosen for the purpose. The method proposed by the author was able to extract the geographical areas of popular objects from geo-tagged photos and assign an appropriate label to it.

An experiment was conducted by the author to evaluate to correctness of regions of interest and the author concluded that 69% of the regions of interest generated with the proposed method are indeed correct. Furthermore, the correctness can be boosted up to 74.4% if roughly 1/3 of the regions of interest are removed from the final set that do not exceed the confidence threshold.

The main limitation of the proposed method is the overall high complexity. High computational complexity limits the dataset size – the regions of interests for the whole world cannot be generated within tolerable time. However, it is possible to overcome this limitation by dividing the map into grids and computing in parallel.

This thesis is a part of a larger research effort which aims to predict human behavior for recommending sightseeing and venues. Future work may include collaboration in this subject.

References

[1] Yiyang Yang, Zhiguo Gong, and Leong Hou U, "Identifying points of interest by self-tuning clustering," in *SIGIR '11 Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, Beijing, 2011, pp. 883-892.

[2] Jitao Sang, Tao Mei, Jian-Tao Sun, Changsheng Xu, and Shipeng Li, "Probabilistic sequential POIs recommendation via check-in data," in *SIGSPATIAL '12 Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, Redondo Beach, 2012, pp. 402-405.

[3] Tanel Tammet, Ago Luberg, and Priit Järv, "Sightsmap: crowd-sourced popularity of the world places," in *Information and Communication Technologies in Tourism 2013*, Berlin, 2013, pp. 314-325.

[4] Zhenni Feng and Yanmin Zhu, "A Survey on Trajectory Data Mining: Techniques and Applications," *IEEE Access*, pp. 2056-2067, 2016.

[5] Fosca Giannotti, Mirco Nanni, Fabio Pinelli, and Dino Pedreschi, "Trajectory pattern mining," in *KDD '07 Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, San Jose, 2007, pp. 330-339.

[6] Sonia Khetarpaul, Rashmi Chauhan, S K Grupta, L Venkata Subramaniam, and Ullas Nambia, "Mining GPS Data to Determine Interesting Locations," in *IIWeb '11 Proceedings of the 8th International Workshop on Information Integration on the Web: in conjunction with WWW 2011*, Hyderabad, 2011, p. 8.

[7] Jinfeng Ni and Chinia V. Ravishankar, "Pointwise-Dense Region Queries in Spatio-temporal Databases," in *2007 IEEE 23rd International Conference on Data Engineering*, 2007, pp. 1066-1075.

[8] Md Reaz Uddin, Chinya Ravishankar, and Vassilis J. Tsotras, "Finding Regions of Interest from Trajectory Data," in *IEEE MDM 2011 - 12th IEEE International Conference on Mobile Data Management*, Lulea, 2011, pp. 39-48.

[9] Markus M. Breunig, Hans-Peter Kriegel, Raymong T. Ng, and Jörg Sander, "LOF: identifying density-based local outliers," in *SIGMOD '00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, Dallas, 2000, pp. 93-104.

[10] Xihui Chen, Jun Pang, and Ran Xue, "Constructing and comparing user mobility profiles for location-based services," in *SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing*, Coimbra, 2013, pp. 261-266.

[11] Luis Otavio Alvares, Vania Bogorny, Bart Kuijpers, Jose A F de M B Moelans, and Alejandro Vaisman, "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*, Seattle, 2007, p. 22.

[12] Andrey T Palma, Vania Bogorny, Bart Kuijpers, and Luis Otavio Alvares, "A clustering-based approach for discovering interesting places in trajectories," in *SAC*

'08 Proceedings of the 2008 ACM symposium on Applied computing, Fortaleza, 2008, pp. 863-868.

[13] Jiajun Liu, Zi Huang, Lei Chen, Heng Tao Shen, and Zhixian Yan, "Discovering areas of interest with geo-tagged images and check-ins," in *MM '12 Proceedings of the 20th ACM international conference on Multimedia*, Nara, 2012, pp. 589-598.

[14] C. Fraley and A. E. Raftery, "How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis," *Comput J*, pp. 578-588, 1998.

[15] Jiawei Han and Micheline Kamber, *Data Mining: Concepts and Techniques (The Morgan Kaufmann Series in Data Management Systems)*.: Morgan Kaufman Publishers, 2000.

[16] M. Emre Celebi, Hassan A. Kingravi, and Patricio A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Systems with Applications*, pp. 200-210, 2012.

[17] Dmitry Laptev, Alexey Tikhonov, Pavel Serdyukov, and Gleb Gusev, "Parameter-free discovery and recommendation of areas-of-interest," in *SIGSPATIAL '14 Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Dallas, 2014, pp. 113-122.

[18] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu, "Density-based spatial clustering of applications with noise," in *Int. Conf. Knowledge Discovery and Data Mining*, 1996.

[19] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander, "OPTICS: ordering points to identify the clustering structure," in *SIGMOD '99 Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, Philadelphia, 1999, pp. 49-60.

[20] Quannan Li et al., "Mining User Similarity Based on Location History," in *GIS '08 Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information system*, Irvine, 2008, p. 34.

[21] Slava Kisilevich, Florian Mansmann, and Daniel Keim, "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*, Washington, 2010, p. 38.

[22] Guilherme Andrade et al., "G-DBSCAN: A GPU Accelerated Algorithm for Density-based Clustering," *Procedia Computer Science*, pp. 369-378, 2013.

[23] Jian Hou, Huijun Gao, and Xuelong Li, "DSets-DBSCAN: A Parameter-Free Clustering Algorithm," *IEEE Transactions on Image Processing*, pp. 3182-3193, 2016.

[24] Rokach Lior and Maimon Oded, "Clustering Methods," in *Data Mining and Knowledge Discovery Handbook*.: Springer US, 2005, pp. 321-352.

[25] S. Inderjit Dhillon and S. Dharmendra Modha, "Concept Decompositions for Large Sparse Text Data Using Clustering," *Machine Learning*, pp. 143–175, 2001.

[26] Jiawei Han, Jian Pei, and Micheline Kamber, *Data Mining: Concepts and Techniques*.: Morgan Kaufman Publishers, 2012.

[27] STHDA - Statistical tools for high-troughput data analysis. [Online]. http://www.sthda.com/english/wiki/dbscan-density-based-clustering-for-discovering-clusters-in-large-datasets-with-noise-unsupervised-machine-learning

[28] Carmelo Cassisi. (2011) DBStrata Data Mining Tool. [Online].

http://www.dmi.unict.it/~cassisi/DBStrata/help/methods.html

[29] Roderick Urquhart, "Graph theoretical clustering based on limited neighbourhood sets," *Pattern Recognition*, pp. 173-187, 1982.

[30] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander, "Density-Based Clustering Based on Hierarchical Density Estimates," in *PAKDD 2013: Advances in Knowledge Discovery and Data Mining*, Berlin, 2013, pp. 160-172.

[31] Leland McInnes, John Healy, and Steve Astels. (2016) hdbscan. [Online]. http://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

[32] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander, "Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2015.

[33] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, "K-means," in *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2009, pp. 360-365.

[34] Greg Hamerly and Charles Elkan, "Learning the k in k-means," *NIPS*, pp. 281-288, 2003.

[35] Kaushik H. Raviya and Kunjan Dhinoja, "An Empirical Comparison of K-Means and DBSCAN Clustering Algorithm," *PARIPEX Indian Journal of Research*, pp. 153-155, 2012.

[36] William E. Winkler, The State of Record Linkage and Current Research Problems, 1999.

[37] William W. Cohen, Pradeep Ravikumar, and Stephen E. Fienberg, A Comparison of String Metrics for Matching Names and Records, 2003.

[38] (2017, Apr.) Wikipedia. [Online]. https://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance

Appendix 1 – Source code for the framework

Git repository:

https://bitbucket.org/martentall/magister.git