RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN
INSTITUT FÜR INFORMATIK III

Bachelor Thesis

A geo-clustering approach for the detection of areas of interest

Author:	**Paul Moosmann**
Email:	**paul.moosmann@uni-bonn.de**
Matriculation number:	**2908663**

First Evaluator:	**Prof. Dr. Jens Lehmann**
Second Evaluator:	**Dr. Damien Graux**

A thesis submitted in fulfilment of the requirements
for the degree of Bachelor of Science
in the

Smart Data Analytics
Computer Science Department

December, 2018

Declaration of Authorship

I declare that this master thesis with the title 'A geo-clustering approach for the detection of areas of interest' and the work presented in it are my own. I conform that:

• This work was done wholly or mainly while in candidature for a research degree at this University.

• Where I have consulted the published work of others, this is always clearly attributed.

• Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.

• I have acknowledged all main sources of help.

Paul Moosmann

Signed: ______________________________

Date: ______________________________

Contents

List of Figures

List of Tables

Abstract

With the availability and increasing accuracy of route navigation services (e.g. Google Maps or OpenStreetMap) the information available about certain places considered relevant for everyday life grows. These points are referred to as points of interest (POIs) To maximize the information that can be gained through this data it can be represented in a structured form, e.g. in the RDF (Resource Description Framework) format. The goal of this work is to gain additional value out of such data (in the context of this work this is a POI database in RDF format). Therefore, an algorithm is proposed which deduces areas of interest (AOI) based on the POI data available. For this purpose, the algorithm uses the descriptive tags which are stored in the database for each POI. Apart from the goal of maximizing the accuracy of the description of each area through an appropriate method of ranking the tags according to their representativeness, another key aspect of the work is the effective determination of dense areas in the database in order to increase the efficiency of the algorithm. The evaluation of the proposed algorithm shows which methods achieve a satisfactory result in regard to these two key aspects.

1 Introduction

The possibility of using points of interest (POIs) in everyday life grows with the increasing coverage of services like Google Maps, Open Street Map or dedicated navigation systems for cars. These services offer an increased information content like coordinates, reviews and descriptive tags for points which have a particular value for specific target groups (e.g. gas stations for car drivers or attractions for tourists). The goal of this work is to implement an algorithm which is able do deduct areas of interest (AOIs) from a given database containing POI data. Here-for it is important to consider that there are various possibilities to define geographical areas. In this work the goal is not to divide the areas by physical characteristics (physical geography) but by their impact on humans visiting or living in this area (human geography). The basis on which these areas are determined can vary. Three major categories are formal, functional and vernacular areas [Duckham et al., 2004]. While the formal areas show administratively connected regions (e.g. cities or countries), functional areas are areas where several regions are interacting with each other and therefore building a new higher-level area (e.g. a large city builds a functional area with its surrounding regions through the attraction of a certain quantity of labour [Farmer and Fotheringham, 2011]). In contrast to formal and functional areas it is not possible to describe vernacular areas on the basis of strictly defined borders or an interaction between existing regions. They are formed through “place awareness or consciousness usually termed “sense of Place””[Shortridge, 1980]. This means the forming of vernacular areas rely on the perception of the people acting within this area. Because this relates very well to the idea of AOIs as “area within an urban environment which attracts people’s attention”[Hu et al., 2015] the concept of vernacular areas is the most important one for this work. These areas cannot be strictly defined, but vary depending of each person’s background. While a local resident of a city might be interested in areas containing supermarkets or public schools, a visitor is likely more interested in areas containing tourist attractions.

In order to identify such AOIs a grid with rectangular tiles is applied on dense areas of our POI database. Each tile is described by the tags of the POIs located in this tile. Part of this work is to identify a tag ranking approach which identifies the relevance of these tags in a way that areas are described in a representative way, without using tags that are too specific to generally describe a whole area. The tags used for this purpose can be determined by a “naïve” approach (i.e. just counting the overall occurrence of the tag in this tile) or by an approach called “term frequency - inverted document frequency” (TF-IDF) weighing the tags according to their relevance in relation to the complete database, thus creating a more differentiated representation of each tile. Tiles with a sufficiently similar representation can then be merged to form larger AOIs. The approach how to do this is described in chapter 4 and an evaluation of said approach can be found in chapter 5.

2 Background

In order to enable a good understanding of the presented work this chapter gives a brief introduction into the used technologies / models. It covers the formal language RDF (**R**esource **D**escription **F**ramework) and the analytics engine Apache Spark. The database used for this work contains 312 385 POIs which are stored in RDF format in order to be able to model information that is implemented in these resources. Since the dataset is quite large, we use Spark to analyse the data because Spark is suitable for large scale data processing.

2.1 RDF

Hitzler et al. introduce RDF in their book **Foundations of Semantic Web Technologies** in the following way:

"The Resource Description Framework RDF is a formal language for describing structured information. The goal of RDF is to enable applications to exchange data on the Web while still preserving their original meaning."[Hitzler et al., 2009]

Apart from being implemented as description of information in web resources, RDF is also used for knowledge management. The RDF model uses so-called triples in order to make statements about resources. A triple consists of the components subject, predicate, and object. Figure 1 shows an example of such a triple.

As figure 1 already indicates a list of triples can be represented as a directed, labelled graph where the subjects and objects are represented by nodes and the predicates are represented by edges. Figure 2 show a graph derived from several triples. This figure also shows that the directedness of the graph is important for the assignment of the nodes to being a subject or an object in a RDF-triple. Such a graph consists of the following different parts which are assigned to be subject, predicate or object:

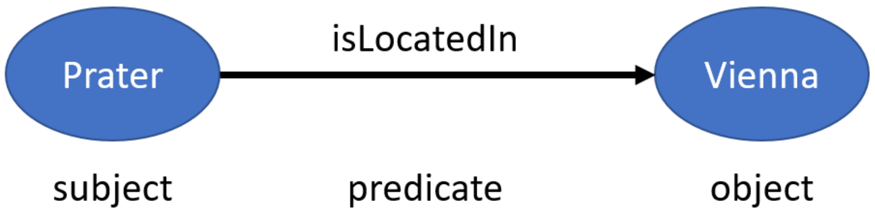


Figure 1: Components of a RDF triple

- **URIs** (**U**niform **R**esource **I**dentifier), which unambiguously reference resources
- **Literals**, which describe data values
- **Blank nodes**, which illustrate an existing object without naming it

Note, that not every one of these parts can be assigned to any part of the RDF-triples. Only the following assignments are possible:

- **Subject:** URI or blank node
- **Predicate:** URI
- **Object:** URI, blank node or literal (in this case the object is represented as a rectangle in the graph form)

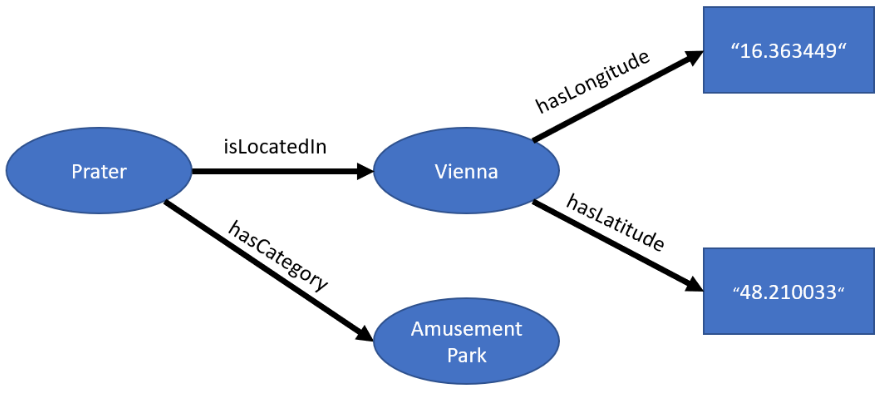


Figure 2: RDF graph

2.2 Apache Spark

Apache Spark is a unified engine for big data processing and machine learning, which is used for the data processing in this work. By exploiting memory computing and further optimizations Spark is a fast and easy to use tool for large scale data processing. It uses a programming model similar to MapReduce and extends it with Resilient Distributed Datasets (RDDs), which are a data sharing abstraction[Zaharia et al., 2016]. Figure 3 (taken from https://spark.apache.org/, 07.12.2018) shows the various high-level libraries Spark can make use of in order to enable support for SQL queries, streaming data, machine learning and graph processing. These can be combined seamlessly in the same application.

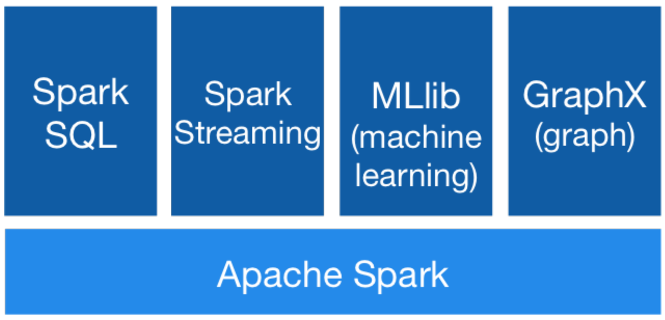


Figure 3: Apache Spark Stack

Spark SQL uses DataFrames, which is a data abstraction to provide support for (semi-)structured data. Spark Streaming performs streaming analytics through dividing the data stream into small batches on which RDD transformations are performed. These DataFrames are also utilized in the data processing part of the algorithm presented in this work. Spark MLlib is a machine learning framework which enables many common machine learning algorithms, e.g. classification, cluster analysis, feature extraction or optimization. GraphX is a graph processing framework which is based on RDDs. Due to RDDs being immutable GraphX is only suited to process graphs which do not have to be updated or altered in any way.

Spark applications can be written in various programming languages, like Scala, Java, Python or R. The algorithm for this work is implemented using Scala.

3 State of the Art

The growing significance of the topic of area of interest extraction has led to numerous scientific publications over the last years. Basis for the extraction of AOIs are various clustering algorithms, which each approach to solve specific problems. Especially research introducing algorithms which work with spatial databases are relevant foundation for this work. In order to achieve meaningful results it is furthermore important to analyse possible approaches on how to achieve a representative description of the derived AOIs.

3.1 AOI Extraction

A survey of clustering algorithms for different data sets and application areas was done by Xu and Wunsch [Xu and Wunsch, 2005] covering the main approaches. Since the application area of AOI extraction requires certain characteristics to the given dataset, the clustering algorithms have to specifically work with spatial datasets. Due to such fields of application Chandra and Amuradha published a survey focusing on clustering algorithms for data in spatial databases [Chandra and Anuradha, 2011]. Of course, the actual implementation of these clustering approaches in order to extract AOIs out of spatial databases has been researched as well. E.g. Liu et al. presented an approach to the discovery of AOIs through analysis of geo-tagged images and check-ins from different websites [Liu et al., 2012]. Further approaches involving deriving data from social media sites were presented by Noulas et al. who applied a clustering algorithm on square areas using Foursquare categories and check-in data [Noulas et al., 2011] or Hollenstein and Purves who used Flickr data to describe city cores on the basis of user-created tags [Hollenstein and Purves, 2010]. Spyrou et al. published a paper titled *A geo-clustering approach for the detection of areas-of-interest and their underlying semantics* which introduces an algorithm for AOI extraction through the exploitation of geo-tagged and user-tagged images from Flickr [Spyrou et al., 2017]. The clustering approach presented there is the basis for this work. The reasons for this are mainly the grid-based approach of this algorithm, which divides the input region into several rectangular tiles and the focus on vernacular geographical regions in contrast to formal or functional geographical regions. The implementation of a grid-based approach supports the idea of including several zooming levels in order to identify dense areas and the main goal of this work is the detection of areas of interest in a vernacular sense.

3.2 Tag ranking

When clustering POIs based on their description / categorisation it is necessary to interpret this description (usually through tags) in an appropriate way. This means e.g. disregarding tags that have a negligible occurrence in regard to the total quantity of tags or eventually weighing more specific tags higher than general ones. Rattenbury and Naaman researched several methods including naïve scan, spatial scan and TagMaps TF-IDF achieve this goal based on user data extracted from Flickr [Rattenbury and Naaman, 2009]. Naïve scan methods compute frequency of data and identify bursts of this data using burst detection methods originating from the field of signal processing [Vlachos et al., 2004]. The spatial scan method uses another burst detection method, namely the spatial scan statistic [Kulldorff, 1999], which test for an abnormal number of occurrences of a specific phenomenon. Finally, the TF-IDF method is a numerical statistic which measures the importance of a given word in a document in relation to a collection of documents. Chaundry and Mackaness presented an approach to identifying the most relevant tags describing geographical areas based on TF-IDF [Chaudhry and Mackaness, 2012]. Since this work aims to identify areas of interest and identify tags describing these areas in a representative way the TF-IDF approach is applied in this work and therefore is described in more detail than the naïve or spatial scan methods in this section.

The idea for a measure of term specificity was first introduced by Spark Jones in 1972 [Sparck Jones, 1972] and is now known as inverse document frequency (IDF). Multiplying this measure with the frequency of the term in the document, also called term frequency (TF), creates the "class of weighting schemes known generically as TF×IDF, which involve multiplying the IDF measure (possibly one of a number of variants) by a TF measure (again possibly one of a number of variants, not just the raw count)"[Robertson, 2004]. Given a collection of N documents and a term t which occurs in n documents, the most common form of the IDF measure is given by

$$idf_t = log(\frac{N}{n}) \tag{3.1}$$

Yet as stated above there is no fixed approach to measure the term frequency or inverted document frequency. Further information on the form of measurement applied in this work can be found in chapter 5.

4 Approach

In this chapter the implemented clustering algorithm is presented. This algorithm is an extended version of the algorithm introduced by Spyrou et al. [Spyrou et al., 2017]. The goal of this algorithm is to find areas of interest through clustering associated POI. The POIs contained in the database used for this work have the following attributes:

- An integer functioning as unique identifier for this POI (in the following referred to as ID)
- Coordinate values (Longitude / Latitude)
- Tags, which describe the Point of Interest (e.g. “Business”, “Company”, “Construction”, etc.)

The clustering is based on the geographic information (coordinates) as well as the description of the POI (tags). In order to enable the aggregation of areas of interest the region covered by the POI database is divided by a grid into rectangular areas. Then two neighbouring areas which are sufficiently similar can be merged into one area. A high-level overview of the structure of the algorithm can be seen in figure 4 below.

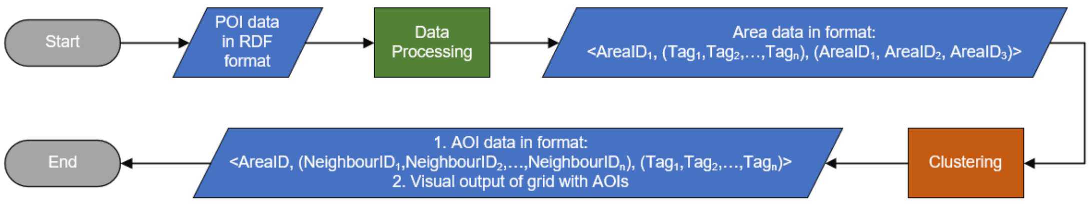

Start
POI data in RDF format
Data Processing
Area data in format: <AreaID$_1$, (Tag$_1$,Tag$_2$,...,Tag$_n$), (AreaID$_1$, AreaID$_2$, AreaID$_3$)>
Clustering
1. AOI data in format: <AreaID, (NeighbourID$_1$,NeighbourID$_2$,...,NeighbourID$_n$), (Tag$_1$,Tag$_2$,...,Tag$_n$)>
2. Visual output of grid with AOIs
End

Figure 4: High level flowchart of the algorithm

4.1 Data Processing

To adjust the available data in a way it can be used by the merging function it is transformed from the initial format:

$$< ID, Longitude, Latitude, (Tag_1, Tag_2, Tag_3, ..., Tag_n) >$$

to the following format:

$$< AreaID_1, (Tag_1, Tag_2, Tag_3, ..., Tag_n), (AreaID_1, AreaID_2, AreaID_3) >$$

To obtain the Area IDs (integer to identify each area) in the resulting format the region covered by the POI dataset is split into rectangular areas by applying a grid. To achieve a more efficient clustering algorithm three different grid sizes are being applied as can be seen in figure 5. Due to enable a less complicated implementation the grid size grows exponentially (5x5, 25x25 and 125x125). To still be able to adjust the influence of the zooming effect for the result of the algorithm, adaptable thresholds are used, which are explained later on in this chapter. These thresholds aim to enable a zooming process which is as "smooth" as possible to avoid missing relevant areas or focusing on areas which might not be relevant enough to be considered. The decision to use 5 as the basis for calculating length and width of the grid on each zoom level was determined experimentally using the given database. This value might have to be adapted for databases with a different geographical coverage or POI density.

Since these grids are placed on a 2-dimensional map the longitude and latitude values are being transformed to a (x, y) tuple describing the POI's position on a coordinate system. The origin of this system is placed in the lower left corner of the grid and originates from the $(Longitude, Latitude)$ values $(9.533743, 46.372712)$. For this transformation geographic information system (GIS) technologies are considered. Maliene et al. describe these as follows: "GIS technologies integrate a range of geographical information into a single analytical model, in which diverse data are "georeferenced" to cartographic projections"[Maliene et al., 2011] and as "a distinctive approach to the spatial analysis of data"[Maliene et al., 2011]. In their work **GIS: An Introduction to Mapping Technologies** McHaffie et al. state that the default way of projecting the geospatial data onto a 2-dimensional surface is to apply the concept of equirectangular projection (also referred to as plate carrée projection)[McHaffie et al., 2018]. Therefore, this concept is also

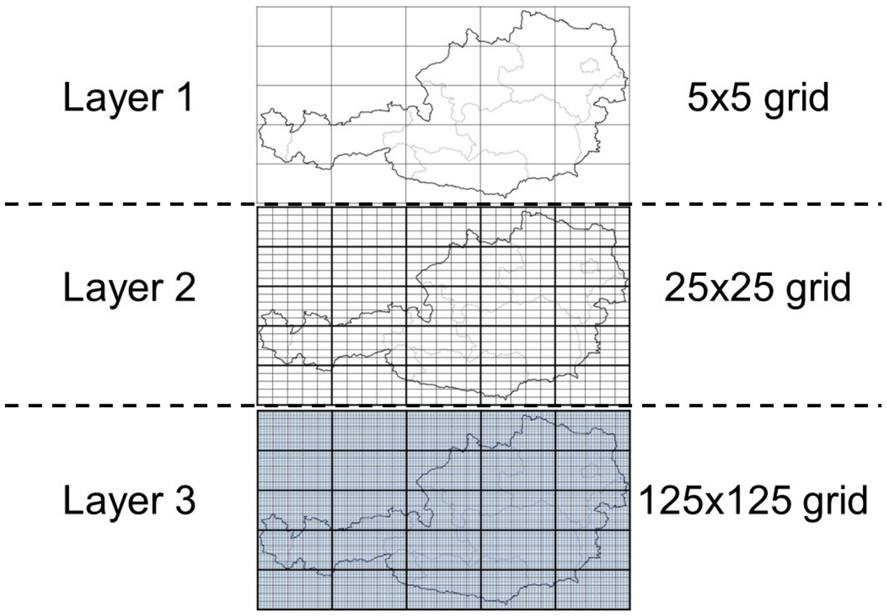


Figure 5: Different grid levels applied on the region of Austria

applied in this work. It is introduced by Snyder[Snyder, 1997] and by using this approach the x and y values can be determined using the following equations:

$$x = ((long\,\frac{\pi}{180}) - (9.533743\,\frac{\pi}{180})) \cdot \cos(46.372712\,\frac{\pi}{180}) \cdot earth\,radius \tag{4.1}$$

$$y = ((lat\,\frac{\pi}{180}) - (46.372712\,\frac{\pi}{180})) \cdot earth\,radius \tag{4.2}$$

Applying this transformation allows us to map an area of each grid level to every POI to describe where this POI is located on the two-dimensional coordinate system. Therefore, each area on each grid level is described by a specific $AreaID$. These $AreaIDs$ describing the location of every POI are added to the data format describing each POI. On the first grid level the area IDs given are ranging from 1 to 25. On the second level every area from the first level is again divided into 25 areas. The enumeration on this level is as follows:

- The area IDs located in $AreaID_1 = 1$ range from 1 to 25
- The area IDs located in $AreaID_1 = 2$ range from 26 to 50

...

- The area IDs located in $AreaID_1 = 25$ range from 601 to 625

The enumeration process of the third and final grid level is performed analogous. The actual order in which the area IDs are being assigned is shown in figure 6 below. The start of the enumeration in the lower left corner was chosen because it resembles the structure of a coordinate system (with the origin being in the lower left corner), but is actually not relevant for the result of the algorithm. Any other corner could be chosen as starting point as well.

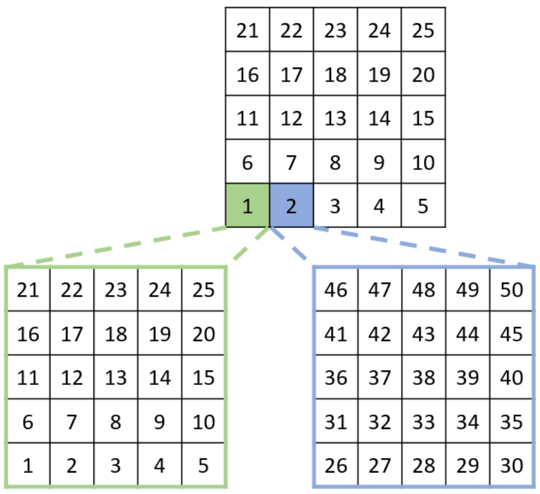


Figure 6: Enumeration of the different grid levels

4.2 Clustering

Figure 7 gives an overview of the main parts of the clustering algorithm. It also brings out the iterative nature of the merging function implemented. A more detailed description of the clustering algorithm is shown in **Algorithm 1: Clustering**

In order to get an efficient computation of the areas of interest, the algorithm concentrates only on “relevant” areas of the input region. The input region consists of all POIs which are currently considered and is separated by the grid into 25 areas. Please note that an input region can also be an area of a bigger input region on a lower grid level. “Relevant” is in this case defined by a sufficient amount of POIs in said area. This quantity of POIs is defined by the variables $POIThreshild_1$ and $POIThreshold_2$ found in lines 1 and 3 of **Algorithm 1:**

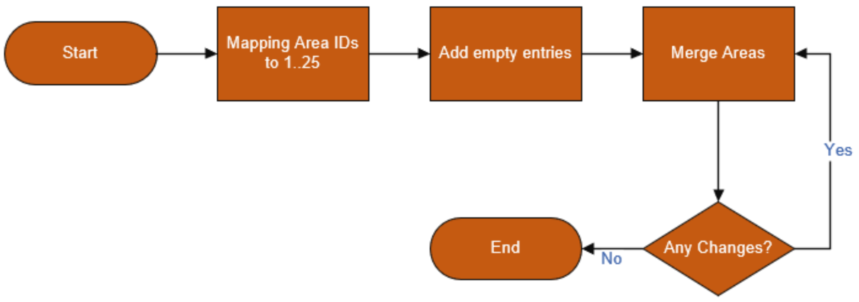


Figure 7: Flowchart of the clustering algorithm

Clustering. The data of the POI lying in relevant areas is the data functioning as input for the merging algorithm. What values are best used to achieve meaningful results is being discussed in chapter 5. In the lines 2 and 4 of the algorithm the enumeration of the new grid level is applied on these dense areas.

Algorithm 1: Clustering

```
Input: POI Data
Output: Areas of interest in dense areas of the input region
 1: $DenseAreas_1 \leftarrow$ Areas on first grid level with $\geq POIThreshhold_1$ POIs
 2: $SGL \leftarrow$ Apply second grid level to partition areas from $DenseAreas_1$
 3: $DenseAreas_2 \leftarrow$ Areas from $SGL$ which have $\geq POIThreshhold_2$ POIs
 4: $TGL \leftarrow$ Apply third grid level to partition areas from $DenseAreas_2$
 5: for all $area \leftarrow DenseAreas_2$ do
 6:     $DenseAreas_3 \leftarrow$ Get all areas from $TGL$ which lie in $area$
 7:     $DenseAreasNewIDs \leftarrow$ Map AreaIDs from $DenseAreas_3$ so that they
        range from 1 to 25 and add empty entries for areaIDs which contain no POI
 8:     $AreaDescription \leftarrow$ Cluster the POIs by their AreaID and determine the
        most relevant tags for each area up to a maximum of 5
 9:     merging($AreaDescription$)
10: end for
```

The last steps before the actual merging function is being applied on the POI data are shown in lines 6 to 8 of **Algorithm 1: Clustering**. Here the POI data for the relevant areas are being extracted and checked for completeness. Since the merging algorithm needs at least one POI in every area of the third grid level one POI is added to empty areas. E.g. if in an given region the area IDs 1,2,...,24 each contain POIs and the area 25 is empty, one POI is added to area 25. The tag describing this POI is set to the string "empty". Then for each dense region the area IDs area being maped to 1,2,...,25. So e.g.:

$$\begin{gathered} 76 \rightarrow 1 \\ 77 \rightarrow 2 \\ ... \\ 100 \rightarrow 25 \end{gathered}$$

In the last step before the merging function all of the POIs geographically located in the same area are being clustered and all tags occurring in this area are being counted and a weighing method is applied to determine the most relevant tags. The used approach is the TF-IDF method, which was introduced in chapter 3. Its metric is given by

$$tf - idf_{t,d} = tf_{t,d} \times idf_t \tag{4.3}$$

where $tf_{t,d}$ measures the frequency of a term t within a document d and idf_t gives a value for the importance of t. In this case $tf_{t,d}$ is the number of occurrences of one tag in the area currently serving as input for the merging algorithm and idf_t is defined as the inverse of the occurrence of said tag in the complete database. That way tags which are very strongly represented get weighed less in the ranking for defining the area-describing tags and we avoid that more general tags are used to describe the majority of the areas.

Another method would be the naïve approach, which simply uses the n most often occurring tags in a specific area as the descriptive tags of this area. In the experiments shown in chapter 5 n is set to 5. Using a smaller number of tags would lead to a lower significance of the description of each area. A larger number would make it harder to find similar areas for the merging algorithm. To bypass this problem the similarity threshold for the merging function could be lowered. But since the naïve approach produced noticeably worse result than the TF-IDF approach, the optimal parameters for the naïve approach were not analysed in depth, but only to the point where a meaningful comparison with the TD-IDF approach could be achieved. The advantages of using the weighed TD-IDF approach are described in chapter 5.

The pseudo code below shows the merging function of the clustering algorithm. In the first step for each area the neighbours are being determined and stored in a list in ascending order. An area is considered as neighbour to another area, if both areas share a border. If they only share one point, meaning the areas lie diagonally to each other, they are not considered neighbours. Figure 8 gives an example of these rules. E.g. the list of neighbours from area 1 would contain the IDs 2 and 6, while the list of neighbours from area 12 would contain the IDs 7, 11, 13, and 17.

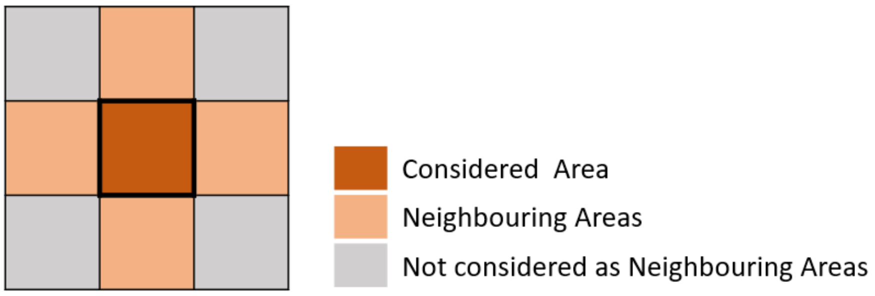


Figure 8: Example of possible neighbouring areas

Algorithm 2: Merging

```
Input: Area Data from a specific region
Output: Areas of interest in this region
  $NeighbourComb \leftarrow$ All possible neighbouring area combinations
  for all $neighbours \leftarrow NeighbourComb$ do
      $jac \leftarrow$ Get Jaccard distance of the describing Tags of the two neighbours
      if $jac \geq jacThreshold$ then
          combine neighbours of the two areas
          combine tags of the two areas and take the five most occurring ones
          remove those instances of $neighbours$ including one of the merged Areas
          update area IDs
      end if
  end for
  return New areas in input region
```

Now for each pair of neighbours it is assessed if the two areas are similar enough to be merged into one area. To determine if two areas are sufficiently similar the Jaccard similarity coefficient [Jaccard, 1912] is applied , which determines the similarity of two sets A, B, given by

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.4}$$

is used. More specifically A, B are sets of the most representative tags in areas a, b. The merging algorithm uses an user defined threshold t and if $J(A, B) \geq t$ then the two areas with the IDs $AreaID_i, AreaID_j$ are merged together. For this the lists containing the neighbouring area IDs N_i, N_j are being merged into a new list

$$N_{new} = (N_i \cup N_j) \setminus \{AreaID_i, AreaID_j\} \tag{4.5}$$

containing all neighbours of the new area, except the IDs of the merged areas. Analogously the tags describing the two areas are being merged and for tags found in both areas the importance values are being summed up. Then up to the five most representative tags are used to describe the new formed area. Five is in this case enough to consider all relevant tags. If less than five tags are evaluated as relevant then this area is described by fewer tags. More information on how to evaluate the relevance of tag is found in chapter 5.

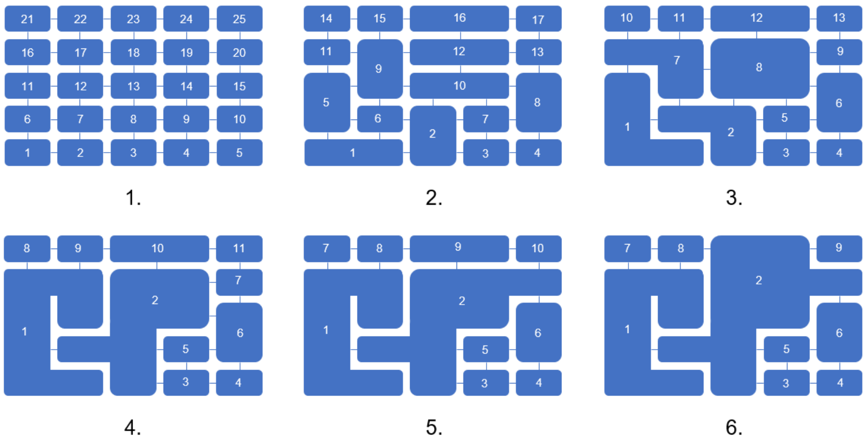


Figure 9: Example: Clustering Areas of Interest in Vienna

It is to be noted that in every merging iteration each area can only be merged with on other area, so after two areas have been merged no further neighbour combinations including these areas are being evaluated. SO e.g. if the areas with the IDs 1 and 2 are merged, then it is not evaluated in this step if the areas with the IDs 2 and 3, or 1 and 6 can be merged. Instead possible mergers of the areas with the IDs 3 and 4 are evaluated. After all possible mergers for this iteration step have been performed, the n newly formed areas are assigned new area IDs ranging from 1 to n. Now the next iteration of the merging function can be conducted. The merging process ends, when no new areas can be created. Figure 9 shows the iteration steps for the region of Vienna as an example of the merging algorithm. In this case the descriptive tags were determined with the naïve approach and threshold for the Jaccard distance was set to 0.8. With the number of descriptive tags per area set to five all of these tags have to be identical for two areas to be merged. This figure only serves as an illustration of the iterative nature of the merging process and not necessarily of a satisfactory result regarding the determined AOIs.

5 Results and Evaluation

In this chapter the results of conducted experiments regarding different aspects of the clustering algorithm are discussed and evaluated. The experiments were run on a machine with:

- Intel Core i5-6200 CPU @ 2.30Ghz, 2401Mhz, 2 Cores
- 8 GB of physical memory
- Microsoft Windows 10 Pro

The database used for this work consists of 312 385 POIs located in Austria and is provided by a manufacturer of route guidance systems for automotives.

5.1 Runtime and Memory Usage

For the conducted experiments regarding runtime and memory usage the database size was adapted in a way such that it only contained 1%, 10% and 50% of the original POI data. For each of these cases the runtime of the data processing part as well as the actual clustering part of the algorithm were examined. Each test was executed multiple times and the results of each test are shown in table 1 and table 2 below.

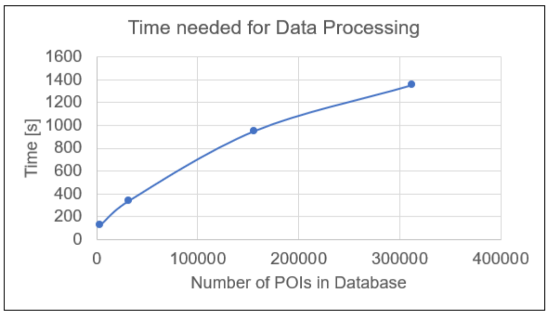
Number of POIs in database	Time needed (1)	Time needed (2)	Time needed (3)	Time needed (Average)
3 124	119s	130s	128s	126s
31 239	330s	327s	358s	338s
156 193	953s	952s	955s	953s
312 385	1 342s	1 376s	1 352s	1 357s

Table 1: Data processing run-time in relation to database size

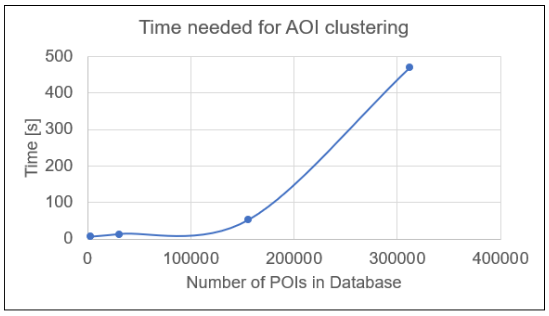
Number of POIs in database	Time needed (1)	Time needed (2)	Time needed (3)	Time needed (Average)
3 124	5s	6s	5s	5s
31 239	12s	13s	13s	13s
156 193	50s	52s	55s	52s
312 385	426s	556s	428s	470s

Table 2: Clustering algorithm run-time in relation to database size

The run-time measurement shows that for the tested sizse of the used databases the data processing part requires most of the run-time, while the AOI clusterings run-time is comparatively short. However, the plotted graphs of the **(a)**data processing and **(b)**AOI clustering in figure 10 show that the graph of the data processing part appears to be of logarithmic nature, while the graph of the AOI clustering seems to be exponential. This means that with increasing size of the database not the data processing, but the clustering part of the algorithm defines the run-time. The values used for these plots are the average values shown in table 1 and table 2.



(a)



(b)

Figure 10: Time needed by the algorithm in relation to the size of the database

Database Size [%]	Increase in memory usage [MB]
1	about 900
10	about 1000
50	about 1000
100	about 1400

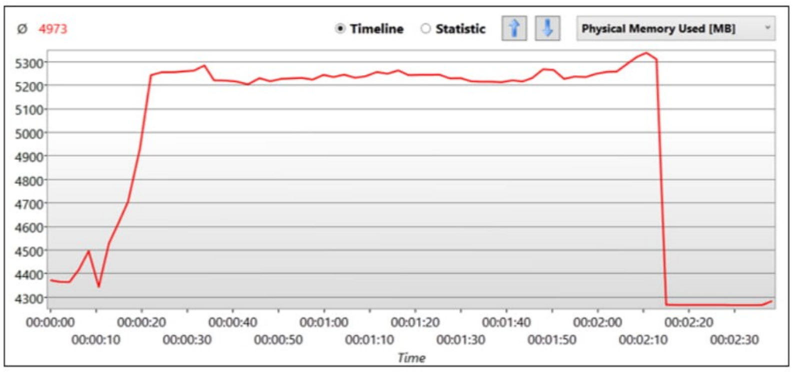
Table 3: Increase of memory usage when starting the algorithm

Figure 11 gives an impression on how much of the physical memory is used depending on the size of the POI database. The usage of virtual memory is not shown in these graphs since its behaviour does not differ greatly from the physical memory. It shows the initial increase in memory usage does not vary a lot between the database size 1%, 10%, and 50%. Only with the 100% database size the increase in memory is clearly higher (see table 3).

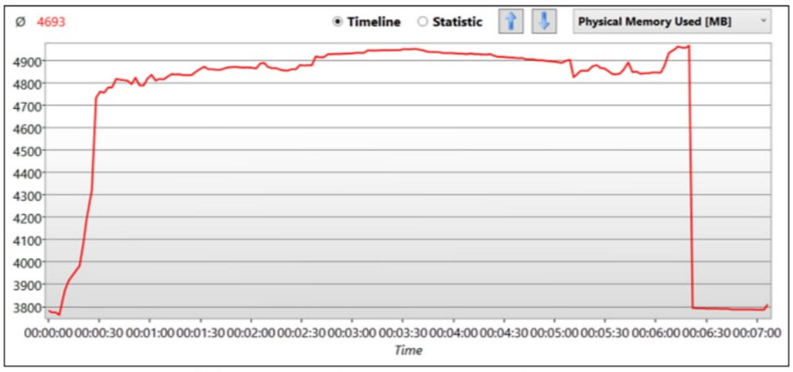
Besides the initial increase in memory usage, a second rise can be observed, when the merging function of the algorithm is executed. The size of this rise in memory usage gets larger the bigger the POI database is. But in contrast to the increasing run-time of the clustering algorithm, here the rise seems to be closer to a linear increase rather than an exponential one (see table 4). Nonetheless this means that the memory usage of the clustering algorithm might be problematic when considering much larger databases, e.g. databases which cover complete Europe instead of only one country.

Database Size [%]	Increase in memory usage [MB]
1	about 100
10	about 200
50	about 600
100	about 1200

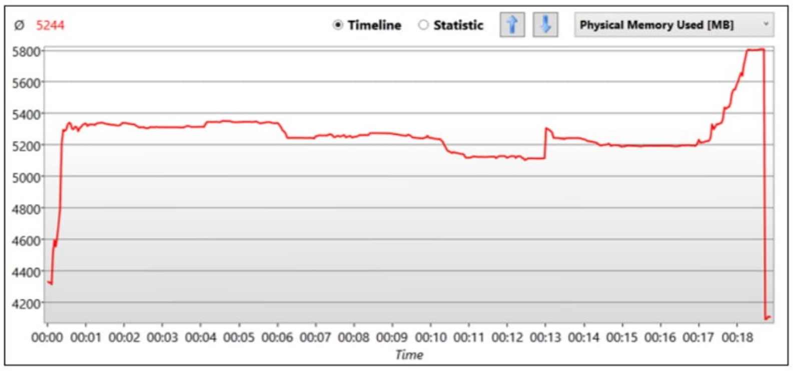
Table 4: Increase of memory usage when running the merging function



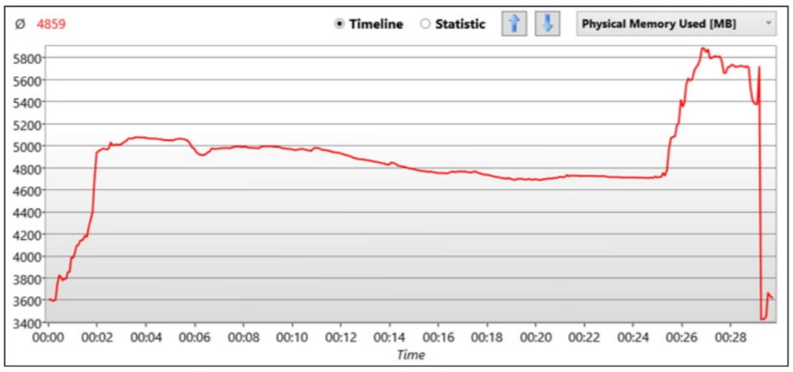
(a) Database with 1% of the available POIs



(b) Database with 10% of the available POIs



(c) Database with 50% of the available POIs



(d) Database with 100% of the available POIs

Figure 11: Physical memory usage during run-time

5.2 Zoom level scaling

As mentioned in chapter 4 the algorithm uses a threshold on the first two grid levels, splitting the input region into several areas. This threshold sets a particular number of POIs to identify dense areas. This threshold varies on the different grid levels, which brings up the question on in what way these thresholds are to be set. For this work two approaches for the scaling of the threshold between the grid levels where examined:

1. linear scaling
2. logarithmic scaling

For the first grid level the threshold is defined by

$$POIThreshold_1 = Number\,of\,POIs\,in\,database/25 \tag{5.1}$$

For the second grid level the following two approaches exist:

$$POIThreshold_2 = POIThreshold_1/25 \tag{5.2}$$

$$POIThreshold_2 = POIThreshold_1/log_2(25) \tag{5.3}$$

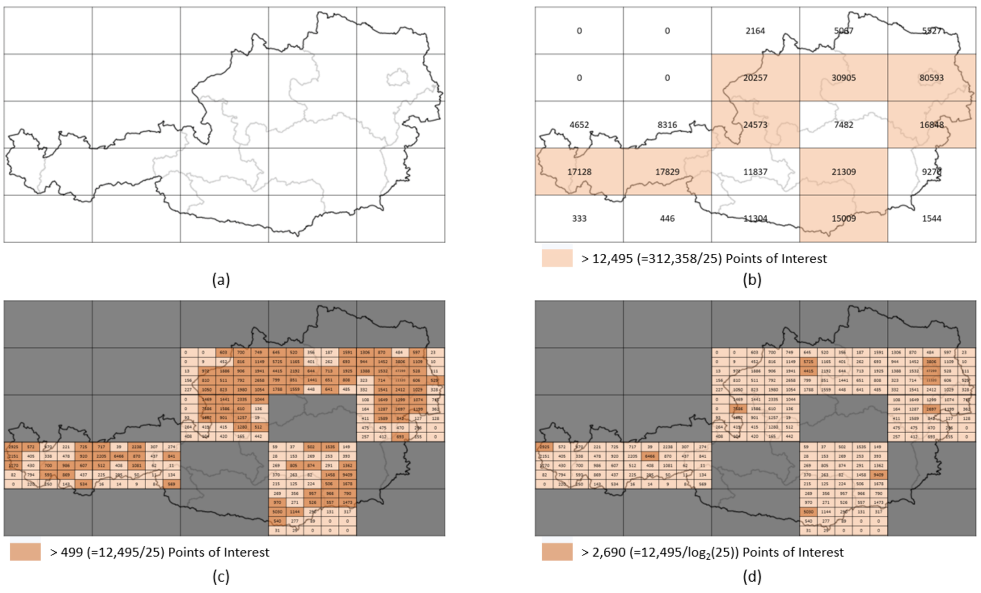

(a)
0
0
2164
5067
5527
0
0
20257
30905
80593
4652
8316
24573
7482
16848
17128
17829
11837
21309
927
333
446
11304
15009
1544
> 12,495 (=312,358/25) Points of Interest
(b)
> 499 (=12,495/25) Points of Interest
(c)
> 2,690 (=12,495/log$_2$(25)) Points of Interest
(d)

Figure 12: Comparison of the results provided by the clustering algorithm with linear and with logarithmic thresholds

Figure 12 shows what dense areas are being identified by these two approaches. Figure 12 (c) shows that on the second grid level 115 of 225 areas are identified as dense areas on which the clustering algorithm is to be run. But since the idea of implementing these thresholds is to identify only really relevant areas in order to avoid running the clustering on too many areas it does not fit its purpose in this case. The threshold is set too low and too many not so relevant areas are wrongly identified as dense areas.

In order to only identify much more relevant areas a logarithmic approach was chosen and the results are shown in figure 12 (d). Now only 12 areas are being identified as dense areas. While this might seem a little low, figure 13 and table 5 (values taken from http://www.citypopulation.de/Oesterreich-Cities_d.html, 07.12.2018) show that all major cities, as well as some smaller ones are correctly being identified as dense areas. Since the goal is to cluster areas of interest in the major cities, this result is satisfying and the logarithmic approach was chosen for setting the threshold.

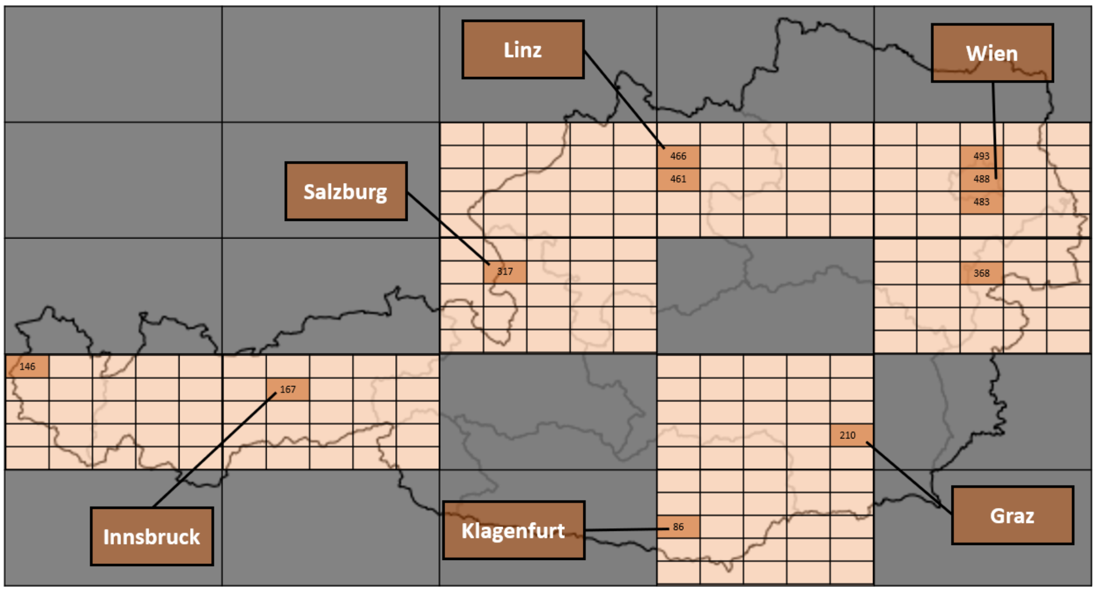


Figure 13: Identification of dense areas

City	Inhabitants
Vienna	1 888 776
Graz	286 292
Linz	204 846
Salzburg	153 377
Innsbruck	132 493
Klagenfurt	100 369

Table 5: Austrian cities with over 100 000 Inhabitants in 2018

5.3 Tag ranking

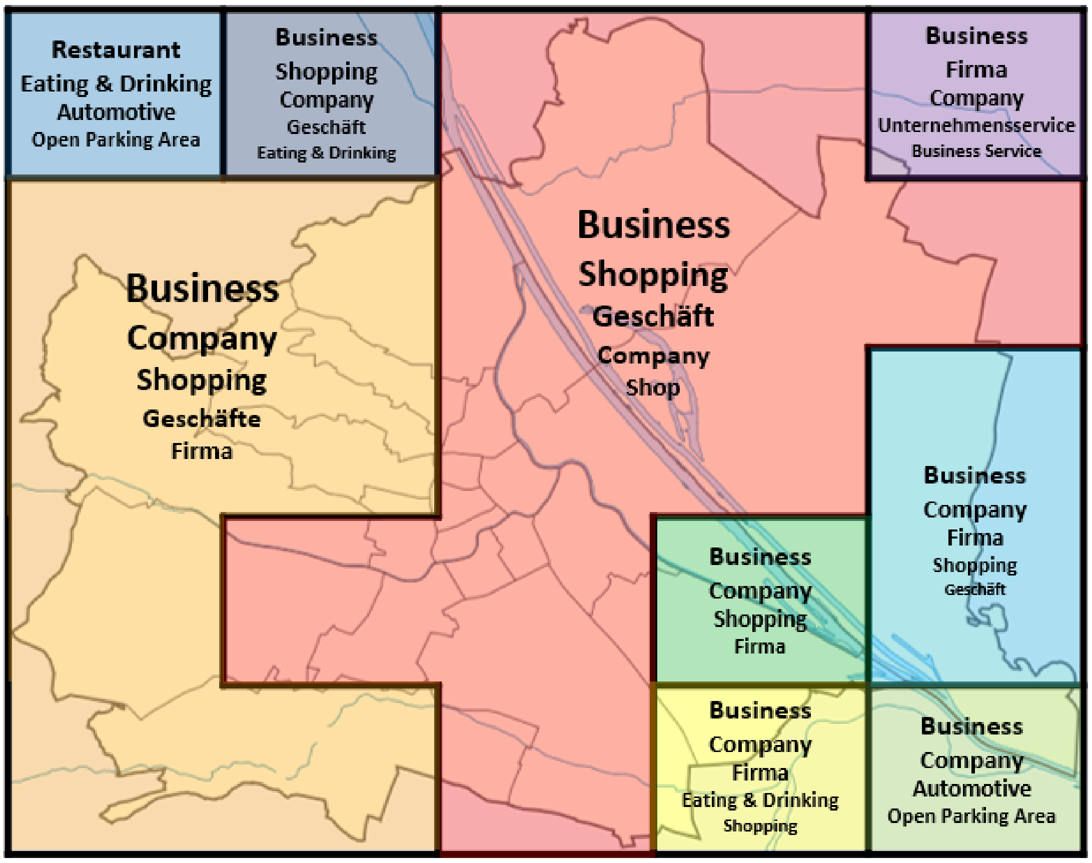


Figure 14: Areas of Interest and their corresponding tags in Vienna

After identifying the dense areas, the algorithm determining the areas of interest in said areas can be applied. Figure 14 shows an visualisation of the outcome of this algorithm being applied on the Vienna area. The visualisation shows that nearly all of the found AOIs have the “Business” tag, as their most occurring tag. Also, the other occurring tags do not vary a lot between the AOIs. As table 6 shows this is the result of the focus of the database on these kinds of POIs.

Tag	Number of occurrences in database
Business	91 495
Shopping	88 466
Shop	83 490
Geschäft	83 490
Company	80 794

Table 6: The five most occurring tags in the POI database

In order to nonetheless get a more diverse description of the AOIs, the TF-IDF approach introduced in chapter 4 was used as tag ranking method. Instead of counting the absolute number of tag occurrences in one AOI, it takes the overall on importance of a tag in the whole database into account. Figure 15 shows that the representative tags for each area are much more specific and diverse than in the result shown before in figure 14. E.g. the tag “Business” is not found in almost every area any more, but in the border areas of the city, which is a more satisfying result.

Since the definition of relevance in regard to the descriptive tags is highly depended on level of granularity desired the identification of a satisfying parameter setting is heuristic based. This makes it necessary to make some adjustments compared to the “naïve” approach of using the absolute number of occurrences to rank the tags. These adjustments lie in the nature of trying to evaluate the importance of a tag. With the idf_t used a tag occurring only once in the whole database would evaluate with an importance of 1, which is the maximal achievable importance. Obviously, a tag occurring only once in the whole database is not suited to be used as a representation of a whole AOI. Therefore, we ignore uncommon tags while using the TD-IDF approach. The threshold here-fore is dependent on the number of POIs located in the currently computed area. Tags which to not surpass this threshold are being weighed with 0. All tags weighed with 0 are not further considered which leads to areas being represented by fewer than five tags. Furthermore, the already existing parameters like the Jaccard Distance threshold have to be adjusted. A value which is too low leads

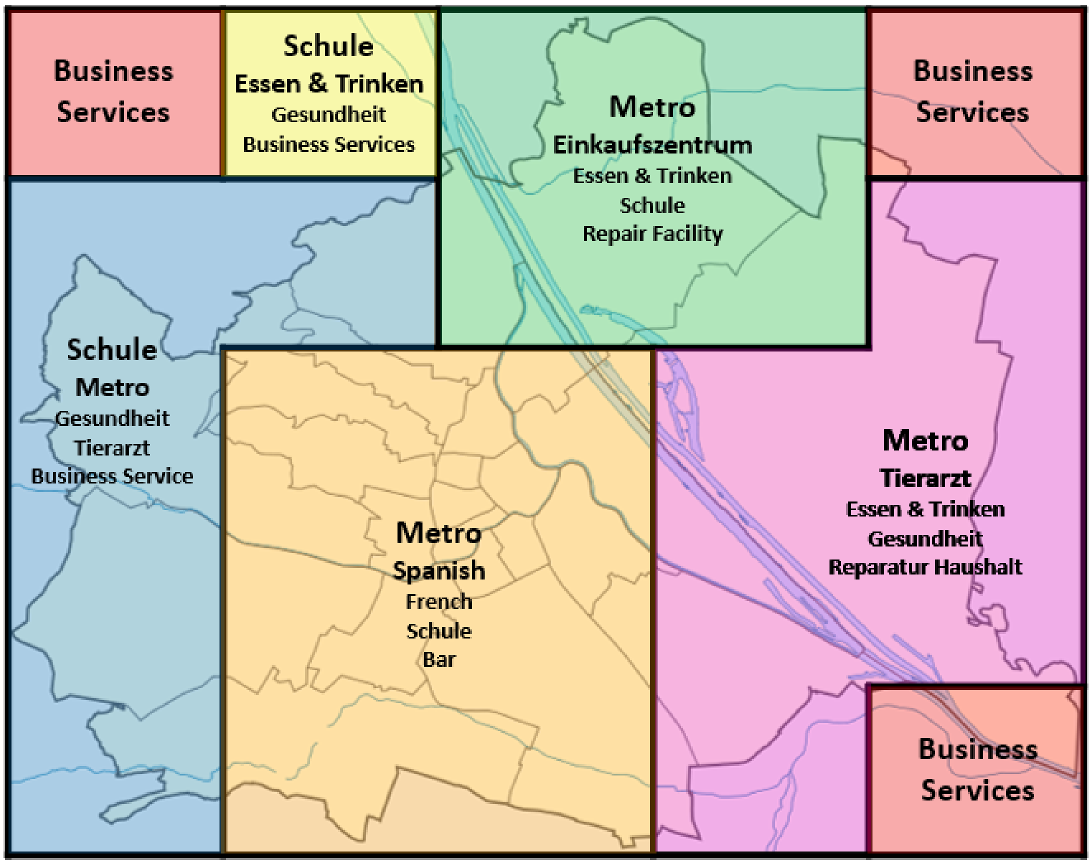


Figure 15: Areas of Interest and their corresponding tags in Vienna using the TD-IDF approach

to counter-productive merging of areas while increasing the Jaccard Distance Threshold leads to an more and more specific representation of the AOIs which simultaneously leads to smaller AOIs to the point where no merging at all takes place. Again the determination of an appropriate value is heuristic-based and can vary depending on the use-case.

The weighing scheme for the idf_t weight as presented in chapter 4 vary from the most commonly used form (logarithmic approach) presented in chapter 3. It is to be noted, that the logarithmic approach for the idf_t weight was also tested, but the results did not bring much improvement compared to the naïve approach of the tag ranking.

6 Conclusions

The algorithm presented in this works aims to aggregate areas of interest from a given points of interest database. In opposition to the algorithm presented by Spyrou et al. [Spyrou et al., 2017] which builds the basis for this work the algorithm does not work "in-the-wild" but needs prior knowledge in the form of descriptive tags for the points of interest in the database. These tags can be more general, e.g. "Business" or more specific, e.g. "Veterinarian".

6.1 Discussion of the results

Since this algorithm gets a database as input which include points of interest of complete Austria and not only bigger cities, a determination of dense areas is performed. On these dense areas the experimental results of the algorithm are satisfactory, but nonetheless they are not optimal for every determined dense area. Therefore, a lot of trial-and-error experiments had to be conducted in order to create overall satisfactory results. The hereby determined values for the various used parameters emerged are based on the used database in the work and may vary for other databases. This inevitably results from the type of descriptive tags used for the POIs, and the possible focus on specific POIs in a database. E.g. the database used in this work is taken from a navigation system for cars and therefore, focuses on POIs which are easily accessible by car. Also, the level of diversity of the descriptive tags is relevant to conclude a suitable tag ranking method limiting the effect of the databases' bias towards a specific type of POIs.

6.2 Future work

As stated in the section above the descriptive tags used in the POI database have a large impact on the result of the algorithm. To improve the variety of tags future work could include enhancing the database with more tags describing each POI. These could e.g. be achieved through tag exploitation from user-generated tags from geo-tagged pictures or reviews shared on social media, e.g. Flickr or Yelp.

For the grid-based zooming approach of the algorithm further experiments could be constructive, in order to determine an “smoother” transition between each zoom level, such that a good detection of dense areas is not so reliable on the assignment of suitable threshold parameters. Also, the optimal number of zooming levels could be further examined. One idea would be to adjust the scale of zooming and the number of zooming levels in a way that they match with the parameters of possible further sources for databases (e.g. OpenStreetMap).

A final idea for possible upcoming work on this algorithm would be to already assign descriptive tags for AOIs on “zoomed-out” levels and not only on the highest zoom level. These could be less detailed tags describing a larger area in a general way.

Bibliography

[Chandra and Anuradha, 2011] Chandra, E. and Anuradha, V. (2011). A survey on clustering algorithms for data in spatial database management systems. *International Journal of Computer Applications*, 24(9):19–26.

[Chaudhry and Mackaness, 2012] Chaudhry, O. and Mackaness, W. (2012). Automated extraction and geographical structuring of flickr tags. *Proc. of GEO-Processing*.

[Duckham et al., 2004] Duckham, M., Goodchild, M. F., and Worboys, M. (2004). *Foundations of geographic information science*. CRC Press.

[Farmer and Fotheringham, 2011] Farmer, C. J. and Fotheringham, A. S. (2011). Network-based functional regions. *Environment and Planning A*, 43(11):2723–2741.

[Hitzler et al., 2009] Hitzler, P., Krotzsch, M., and Rudolph, S. (2009). *Foundations of semantic web technologies*. Chapman and Hall/CRC.

[Hollenstein and Purves, 2010] Hollenstein, L. and Purves, R. (2010). Exploring place through user-generated content: Using flickr tags to describe city cores. *Journal of Spatial Information Science*, 2010(1):21–48.

[Hu et al., 2015] Hu, Y., Gao, S., Janowicz, K., Yu, B., Li, W., and Prasad, S. (2015). Extracting and understanding urban areas of interest using geotagged photos. *Computers, Environment and Urban Systems*, 54:240–254.

[Jaccard, 1912] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50.

[Kulldorff, 1999] Kulldorff, M. (1999). Spatial scan statistics: models, calculations, and applications. In *Scan statistics and applications*, pages 303–322. Springer.

[Liu et al., 2012] Liu, J., Huang, Z., Chen, L., Shen, H. T., and Yan, Z. (2012). Discovering areas of interest with geo-tagged images and check-ins. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 589–598. ACM.

[Maliene et al., 2011] Maliene, V., Grigonis, V., Palevičius, V., and Griffiths, S. (2011). Geographic information system: Old principles with new capabilities. *Urban Design International*, 16(1):1–6.

[McHaffie et al., 2018] McHaffie, P., Hwang, S., Yang, B., and Follett, C. (2018). Gis: An introduction to mapping technologies.

[Noulas et al., 2011] Noulas, A., Scellato, S., Mascolo, C., and Pontil, M. (2011). Exploiting semantic annotations for clustering geographic areas and users in location-based social networks. *The social mobile web*, 11(2).

[Rattenbury and Naaman, 2009] Rattenbury, T. and Naaman, M. (2009). Methods for extracting place semantics from flickr tags. *ACM Transactions on the Web (TWEB)*, 3(1):1.

[Robertson, 2004] Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520.

[Shortridge, 1980] Shortridge, J. R. (1980). Vernacular regions in kansas. *American Studies*, 21(1):73–94.

[Snyder, 1997] Snyder, J. P. (1997). *Flattening the earth: two thousand years of map projections*, pages 5–8. University of Chicago Press.

[Sparck Jones, 1972] Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

[Spyrou et al., 2017] Spyrou, E., Korakakis, M., Charalampidis, V., Psallas, A., and Mylonas, P. (2017). A geo-clustering approach for the detection of areas-of-interest and their underlying semantics. *Algorithms*, 10(1):35.

[Vlachos et al., 2004] Vlachos, M., Meek, C., Vagena, Z., and Gunopulos, D. (2004). Identifying similarities, periodicities and bursts for online search queries. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 131–142. ACM.

[Xu and Wunsch, 2005] Xu, R. and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678.

[Zaharia et al., 2016] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., et al. (2016). Apache spark: a unified engine for big data processing. *Communications of the ACM*, 59(11):56–65.