Victor de Graaff was born in Rotterdam, The Netherlands, on August 16, 1985. In 2008, he received his Bachelor degree in Electrical Engineering from the Technical University of Delft. He graduated from the University of Twente in Computer Science in 2009. From 2011 until 2015 he pursued his PhD at the University of Twente, which resulted in this dissertation on geosocial recommender systems. These recommender systems, which are based on trajectory data and social media profiles, are used for the recommendation of location-bound objects, such as holiday homes or real estate. In this dissertation, an architecture for a geosocial recommender system is laid out, and its components are explored, validated and discussed in detail.
Geosocial Recommender Systems
Victor de Graaff
SIKS Dissertation Series No. 2015-34
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

DOI 10.3990/1.9789036539777

Cover design: Victor de Graaff
Copyright © Victor de Graaff
GEOSOCIAL RECOMMENDER SYSTEMS

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
Prof.dr. H. Brinksma,
on account of the decision of the graduation committee,
to be publicly defended
on Thursday December 10th, 2015 at 16:45

by

Victor de Graaff

born on the 16th of August, 1985
in Rotterdam, The Netherlands
This dissertation has been approved by:

Supervisor:
prof.dr. P.M.G. Apers

Co-Supervisors:
dr.ir. M. van Keulen
dr.ir. R.A. de By
Acknowledgments

PhD thesis acknowledgements these days have become almost a list of names of people that authors met during their time as a PhD candidate. I would like to break with this tradition, and dedicate this space to those people that had the most significant impact on both my research and my well-being throughout the entire four years of my PhD research.

First of all, there is Maurice. Maurice, you are a wonderful person who combines a motivating attitude with subtle coaching. I always enjoyed the freedom you gave me in choosing the direction of my research, but also felt the gentle virtual pushes against my shoulder when I started heading in a wrong direction. In the first three months, I had to switch from a pragmatic getting-things-done attitude to a dream-big perspective. The time you took to coach me in this, especially during our early coffees on the third floor of the Zilverling building, have made this transition as smooth as possible. Also the small meetings-after-the-meeting, where you made sure that I kept faith in my own possibilities, despite the prior discussion, were a strong support throughout my entire PhD. And while the educational load on your shoulders was growing, all the PhD’s from our group have been able to count on you for putting in the effort to supervise us, while keeping our backs free from spending too much time on re-inventing the education wheel. I could not have wished for a better first supervisor.

Secondly, there is Rolf. Rolf, you and I have quite a bit in common, and the most important piece of common ground is our strong opinion. I have to admit: often you were right, and I was almost right. Your input into my thesis has provided a huge boost of quality. Countless were the times where you pointed out where improvements could, should, or must be made, in order to remove repeating repetitive double ‘duck talk,’ \LaTeX ‘thingies,’ and false ordering of interpunction, often with a touch of humor. By the way, did you notice the usage of plural in the first sentence of these acknowledgements? And more than once, the opinions of reviewers reflected what you had already warned me for. But most importantly, you know one thing about me: sometimes, just sometimes, I need a wake-up call. A wake-up call to put in a little more time, and a little less hurry.
seemed like you had no problem being that alarm clock every now and then. Thank you for being my alarm clock!

I would also like to thank both of you for the trust you had in me and the freedom you gave me to explore the numerous committees and organizations where I wanted to play a role throughout my PhD trajectory. I believe the skills that I developed there are much more valuable than what I could have learned from writing one more paper, and I am very grateful for this opportunity.

Number three on this list is my coffee buddy Jan. In the first year, you and I worked together on the data harvester, but while the cooperation reduced over the following years, the joint coffee, tea, and chocolate milk consumption increased. I am curious about the coffee bill statistics of our research group over the past months, as it is hard to imagine that my departure would go unnoticed there. Jan, we always find something to talk about, whether it is the next holiday destination, cycling, polar bears, trips to Amsterdam, the latest news from PNN and the university council, or restaurant recommendations in The Hague (sorry about that), we’ve always known what is keeping the other busy. These talks have always been a great start of the day, and as the end of my PhD came in sight, at any given time of the day. I always came to work with a smile, and our coffee chats were definitely one of the major reasons for it.

Since one coffee buddy just did not suffice to get me through the day, number four and the last one on this list is my other coffee buddy: Mohammad. The first months of our joint office time, I thought you were a relatively closed person, but as we shared an office longer, and I talked you into being my successor as the P-NUT secretary, we grew closer and closer, and in fact, I realized you are not a closed person at all. My second coffee of the day became timed to match your office starting time, and we saw each other outside work frequently, making you more a friend that happened to work in the same group, than a colleague. Maybe it was only for the better that we got separated during our move to the second floor, as for both of us the end is in sight now. Moh, good luck with the last bits, and I’m glad to be around to distract you some more!

I would also like to thank my long-time office mate Mena and my closest friend since I live in the Enschede area Adriana for enduring my complaining when it was time to blow off some steam, and bringing me back to a positive mood. Your pleasant company has played a major role in my happiness during this PhD.

And to conclude, I would like to thank those who funded my research, because for the last time: this publication was supported by the Dutch national program COMMIT/.
Contents

1 Introduction 1
  1.1 Inspiration scenario ................. 2
  1.2 Geosocial recommender system architecture ....... 4
  1.3 Trajectory analysis ................ 5
  1.4 UGC quality assessment ......... 7
  1.5 Recommendation selection .......... 8
  1.6 Thesis scope & structure ........ 8

2 Architecture 11
  2.1 Introduction ..................... 11
  2.2 Related work .................... 12
  2.3 Information collection ............ 13
  2.4 Information enrichment .......... 16
  2.5 Recommendation selection ........ 17
  2.6 Architecture overview .......... 19
  2.7 Conclusion ..................... 20

3 Point-of-interest collection 21
  3.1 Introduction .................... 21
  3.2 Related work .................... 23
  3.3 NeoGeo scraper .................. 24
  3.4 POI scraping ..................... 33
  3.5 Conclusion ..................... 36

4 POI to POLOI conversion 37
  4.1 Introduction .................... 37
  4.2 Related work .................... 39
  4.3 Used data sources ............... 40
  4.4 Terminology ..................... 42
  4.5 Approximation approaches ....... 43
## Contents

4.6 Validation .............................................. 47  
4.7 Privacy preservation ................................. 50  
4.8 Conclusion ............................................. 51  

5 Automated semantic trajectory annotation ............... 53  
5.1 Introduction ............................................ 53  
5.2 Related work ............................................ 56  
5.3 Approach ................................................ 58  
5.4 Validation ............................................... 62  
5.5 Privacy considerations ............................... 68  
5.6 Conclusion ............................................. 69  

6 Spatiotemporal profiling for UGC quality assessment .... 71  
6.1 Introduction ............................................ 71  
6.2 Related work ............................................ 73  
6.3 Case study, data collection & data pre-processing ....... 74  
6.4 Spatiotemporal behavior analysis ..................... 76  
6.5 Behavior prediction .................................... 82  
6.6 Conclusions ............................................. 87  

7 Knowledge-based recommendation selection ............... 89  
7.1 Introduction ............................................ 89  
7.2 Related work ............................................ 91  
7.3 Motivation ............................................... 93  
7.4 Concept & technology ................................ 94  
7.5 Validation ............................................... 101  
7.6 Conclusion ............................................. 106  

8 Conclusion ............................................... 109  
8.1 Research questions revisited ......................... 109  
8.2 Future research directions .......................... 112  
8.3 Concluding remarks .................................. 113  

Bibliography .................................................. 115  

SIKS Dissertation list ....................................... 125  

Summary ...................................................... 137  

Samenvatting ................................................ 139
Introduction

Daily life is full of location-related decisions: where to go on vacation, which house to buy, which job to apply for, which route to take for the Saturday shopping. These decisions are not only influenced by the characteristics of this holiday home, house, or company, but also by the region it is located in. What is the distance to the beach, the nearest train station, or the schools and kindergartens? Another important aspect in this decision is the person making it. Is the decision maker interested in cultural events or looking for a relaxing vacation on the beach? Is he looking for a job in a big city, or more a country-side type of person? But when a person starts looking for a place to go or where to live, the number of options is overwhelming. For example, over 200,000 houses are currently for sale in The Netherlands [33]. In this thesis, we show how recommender systems can help users to narrow down the options based on the surroundings of a location-bound object (LBO). An LBO is an object that cannot be seen separately from its location, such as a house or a holiday home.

Computer systems are suitable to search through thousands of potential matching locations based on the wishes of a user. However, first we need to know what the user wants to do, and where he can do these things, before we can recommend him where to go. What a person wants to do on a vacation is strongly related to his interests, his mobility, and whether or not this person will be travelling alone, in a group, or with children. A person who is moving to a new house uses similar criteria, as a car owner will value a nearby train station less than a commuter does, and a parent is more likely to look for a neighborhood with a school. To describe such locations we use the concept of a point-of-interest (POI): a location where goods and services are provided, geometrically described using a point, and semantically enriched with at least an interest category. An interest category provides information about the type of services or goods that can be expected at the location, such as restaurant, supermarket, or airport. In trajectory analysis, a point representation often does not suffice. Therefore, we also introduce the two-dimensional counterpart of a POI, its polygon-of-interest (POLOI): a location where goods and services are provided, geometrically described using a polygon, and
1. Introduction

semantically enriched with at least an interest category. The number of nearby POIs and their categories influence the attractiveness of a neighborhood for a specific purpose.

Recommender systems are widely spread to recommend books, movies, music and many other types of consumer products to users. In recommender systems, a user profile captures information that can be used to compare the profiled user with other users, and is used to predict what this user may like or need. User profiles can be populated from many input sources. In this thesis, we focus on input from trajectory data, and social media profiles. A person who visits a gas station on a regular basis probably owns a car, and a person who visits a school every morning for five minutes probably has a child there. We use this information to populate a user profile to describe those characteristics of a person that are relevant for the recommendation process. Similarly, a geoprofile can be used to compare neighborhoods with each other. A geoprofile contains those characteristics of a region that are relevant to the decision to be made. Examples of such characteristics are nearby POIs, climate, and spoken languages: a geoprofile for real estate contains information such as the distance to the nearest train station or nearby schools, while a geoprofile for a holiday home is more focused on nearby touristic attractions. The types of user interest we use in recommendations, therefore always have a relation with the type of object we are trying to recommend. When recommending holiday homes, it is useful to know if a person is more interested in a cultural or a sportive vacation, while in the recommendation process of real estate, it is more important to know where this person works or spends free time.

In this thesis, we introduce a geo-social recommender system (GRS). A GRS is a system that is specifically designed to use social media content and geospatial data to discover a user’s interests and to create a match with LBOs. For our system, called GeoSoRS (GeoSocial Recommender System), we provide an architecture and in-depth elaboration of its components, by answering the four research questions that are introduced in the remainder of this chapter.

1.1 Inspiration scenario

The work in this thesis is inspired by the holiday home portal of EuroCottage. EuroCottage is a small Dutch company that allows its clients to search and book approximately 200,000 holiday homes throughout Europe. In this section, we introduce a vision of how user-generated content (UGC), and especially volunteered geographic information (VGI), can be used to improve the recommendations on this holiday home portal.

For the selection of a holiday home, a person takes several considerations into account. Besides price, size, availability, and the holiday home
aesthetics, the location of the holiday home plays a large role in the decision-making process. While price, size and availability are already available as criteria to narrow down the search in current holiday home search engines, the options to search for locations are rather limited. One can choose to search in specific countries, or nearby the sea, but one cannot for example search on the vicinity of cultural places, or amusement parks. This information however, is already present in some of the descriptions to a certain extent, albeit as free text input. In this thesis, we explore the possibilities to extract this information from these descriptions, using existing techniques, and explore new possibilities to match them with a user. For the holiday home domain, one could ultimately think of the geoprofile schema example that is given in Figure 1.1, consisting of:

1. the POIs within a reasonable distance;
2. the (un)organized events within a reasonable distance;
3. the socio-demographical backgrounds of the neighborhood;
4. the geographic properties of the location.

While the holiday home descriptions form a good starting point for interest detection, existing customer experiences may even be more suitable. If several renters of a holiday home visited the same POI during their stay, this POI may be worth visiting for potential visitors as well. Similarly, recommendations by locals, or even holiday home owners, can be of value while creating recommendations. This is illustrated in Figure 1.2. The reviews of nearby POIs and automatically detected visits of places through VGI content around the cottage can be used to inform or recommend the holiday home to new renters. This UGC and VGI can be generated by former renters, local businesses and/or holiday home owners. The UGC on items in this area does not necessarily come from the company’s own platform, but can also be collected from social media. Social media profiles of potential renters are also useful to use as an information source for the interests of that renter, to find a matching holiday home for them. The gray
Figure 1.2: Holiday home domain: input can be provided by multiple sources, and detected from multiple channels, such as social media or VGI

mini-networks on the outside of Figure 1.2 illustrate that these people may be connected to other holiday homes, or cottages, as well.

1.2 Geosocial recommender system architecture

Recommendation selection in our GRS is a step-by-step process of detecting a user’s interests, and discovering characteristics of all LBOs. Information on users and objects needs to be collected, combined and enriched before it can be used in any recommender system. In the exploratory phase of this research, we focused on which components are necessary for LBO recommendations. To create a solid foundation for our GRS, we answer the following research question:

**RQ1. Which software components can contribute to LBO recommendation based on the LBO’s geographic embedding and a user’s interests?**
1.3 Trajectory analysis

Chaw et al. already proposed an architecture for a social network with geotagged recommendations in [21], just as Papadimitriou et al. did in [72]. While their architectures are based on the creation of a new social network, throughout this thesis we will focus on the usage of information that is already available in external social media, such as Facebook. For each of the components from the architecture we present in Chapter 2, we give a direction of its implementation. These components formed a basis for the next phases of the research, and led to the consecutive research questions below.

1.3.1 Point-of-interest collection

Cadastral data, with its two-dimensional descriptions of building and property outlines, forms a good starting point for trajectory analysis. However, this data is often expensive, or even impossible to obtain. On the web, information on POIs is often freely available, but: (1) each information source has its own structure, and (2) information is scattered over many websites. To cope with these challenges, we attempt to answer the following subquestion:

RQ2a. How can existing POIs be collected from the web using minimal resources, with respect to both human effort and computation power?
1. Introduction

![Figure 1.3: POIs with their respective POLOIs. POIs are represented by the red dots, their POLOIs by the blue polygons directly surrounding them.](image)

We attempt to answer this subquestion in Chapter 3, where we introduce a scraping method based on *scraper workflows*, rather than rigid scraping configurations. This part of our work is inspired by wrapper induction, as it was introduced by Kushmeric in [55]. We show how combinations of induced wrappers can be used in general, and for POI collection specifically.

1.3.2 Polygon-of-interest estimation

A third problem arises when it comes to POI collection from the web: information on POIs is often limited to an address, or a coordinate pair at best. Some POIs however, such as an airport, are very large, while others, such as a flower shop, are relatively small. This requires us to describe a POI as more than just a point object, if this information is to be used for trajectory analysis. We address this problem through the next subquestion:

RQ2b. *How can the size and shape of a parcel related to a POI be estimated?*

In Chapter 4, we attempt to find an answer to this question, by transforming POIs into POLOIs, as illustrated in Figure 1.3. Gianotti et al. acknowledge the need for POLOI detection in [34], but they assume that the set of POLOIs is provided as an input to their approach. In Chapter 4, we introduce six different approaches for POLOI estimation, and compare them.
1.3.3 POI visit detection

The final subquestion for interest detection from trajectory data follows from the necessity to cope with typical problems with trajectory data: this data is noisy and incomplete, due to the influences of, amongst others, low sensor quality, signal multi-path and loss of signal inside buildings. This leads to the final subquestion:

RQ2c. How can POI visits be detected from mobile trajectory data?

Several attempts have been undertaken to extract POI visits from trajectory data. There are generally two ways to do this. The most common way is to detect POIs from slow movement over a longer period of time, and defining the locations at which this happens regularly as the POI set. This has been done for example by Ashbrook et al. [6] or Zheng et al. [97]. A less common way is to match trajectories with a given POI set, such as the one by Alvares et al. [4]. The advantage of the latter approach, is that more information on the matched POI may already be available, such as a name, address, website, and POI category. In Chapter 5, we show the drawbacks of existing approaches, and introduce and validate a new approach.

1.4 UGC quality assessment

UGC is by nature imprecise, and sometimes conflicting. One person reviews a restaurant positively, while someone else’s experiences were quite negative. Especially when the number of reviews per item is relatively low, it is important to filter out low quality content. With the following research question, we attempt to find a solution to do so, based on trajectory data:

RQ3. Which methods can be used to assess the quality of UGC, based on trajectory data?

Several attempts for behavior pattern detection from trajectory data have been undertaken, such as the ones by Giannotti et al. [34], Zheng et al. [98], and Spaccapietra et al. [81]. All these approaches for behavior detection from movement data are useful for several applications, but with the method we introduce in Chapter 6, we attempt to find new ways to detect patterns both from geometric trajectory data, as well as from preprocessed data that already contains some semantics.
1. Introduction

1.5 Recommendation selection

With the increased social media usage over the past decade, we have a new, semi-structured, way to access knowledge about a person’s interests. On Facebook users have the option to like pages about places, organizations, products or famous people, on Twitter users can follow companies or famous people, and so on. Therefore, social media content forms a good starting point to build a user profile based on their interests. In the recommendation selection process, we focused on the structured interest information from social media profiles to answer the final research question:

RQ4. How can interests be extracted and used to select recommendations from a set of LBOs?

To find an answer to this research question, we propose a method based on external social media and generic knowledge-bases. The potential for recommendations based on these sources has also been described by Passant and Raimand in [75] or Bostandjiev et al. in [15], for example. The Interest-Based Recommender System (IBRS) we propose as a component for our GRS is not only suitable for LBOs, but is even applicable for other domains, as we demonstrate in Chapter 7.

1.6 Thesis scope & structure

With respect to the problem we laid out at the beginning of this chapter, in this thesis we aim to provide possibilities to collect POIs, both for interest extraction and to construct geoprofiles. We also provide ways to detect interests from both trajectory data, through POI visit detection and behavioral patterns detection, and from social media. With the IBRS recommendation component, we match these interests with interesting locations detected from holiday home descriptions. For each of the components resulting from RQ2a, RQ2b, RQ2c, RQ3 and RQ4, we validate our answer separately, for which the results are discussed in its respective chapter. Furthermore, since we are working with data on privacy sensitive information, such as location information, we also provide hints and considerations on how to use our solutions in a privacy respecting way throughout the thesis.

To conclude this introduction, we provide you the overview of the thesis structure in Figure 1.4 as a reading guide. Each of the above research (sub)questions is discussed in a separate chapter. Chapter 2 discusses the architecture resulting from RQ1. In Chapter 3, we describe the web harvester from RQ2a. In Chapter 4, we present the POLOI estimation algorithm from RQ2b. RQ2c is answered through the trajectory analysis approach in Chapter 5. Chapter 6 presents the behavior detection method from RQ3.
1.6. Thesis scope & structure

Geo-social recommender system (RQ1/CH2)

Scraping (RQ2a/CH3)

POI-to-POLOI Conversion (RQ2b/CH4)

Visited POI Detection (RQ2c/CH5)

Behavior Pattern Detection (RQ3/CH6)

User profiles

Location profiles

Recommendation Selection (RQ4/CH7)

Location-Bound Objects

Figure 1.4: Global overview of the components of a GRS discussed in this thesis. Each research question (RQ), represented by a box (including the dashed one) is discussed in a separate chapter (CH).

In Chapter 7 we discuss the recommendation selection techniques of RQ4. Chapter 8, finally, contains the conclusions and provides directions for future work.
Architecture

Abstract

In this chapter, we propose the GeoSoRS architecture that is designed to create personal LBO recommendations based on a person’s whereabouts and social network profiles. In this architecture, information is used from multiple sources to extract suitable recommendations: authoritative data, knowledge bases, internal UGC/VGI, web content, social media content, and the already available LBO product database. In three steps, this information is collected from the corresponding source, enriched and combined, and, finally, used for LBO recommendation extraction. For each of these steps, the components used in GeoSoRS are introduced, and for each component, we discuss existing work, propose a solution, and provide pointers to where this component is discussed in more detail in the remainder of this thesis.

This chapter is based on [28]

2.1 Introduction

Large online stores have access to massive amounts of customer data, such as past browsing and purchasing behavior. This data can be used to support customers in their decision-making for future purchases. Collaborative filtering-based RSs have been designed and proven their effectiveness over the past decades [78]. The main drawbacks of these systems are their vulnerability to data sparsity and the cold-start problem [2]. The first problem is especially the case for the recommendation of a large item set, to a relatively small group of users, as is the case for the holiday home broker from our running example. For this scenario, knowledge-based RSs are a more suitable alternative [13]. A knowledge-based RS is a RS that uses a combination of knowledge on the item set with knowledge on the users to extract recommendations, and does not rely on (browsing and/or purchasing) behavior from other customers. In a knowledge-based RS for holiday homes, for example, a user with a known preference for France gets holiday home recommenda-
2. Architecture

Current GRSs, such as GeoLife 2.0 by Zheng et al. [95], or GeoSocialDB by Chow et al. [21], provide location-based recommendations for nearby POIs. This deviates from our goal, as we intend to provide recommendations for remote locations, based on a collection of interesting or useful POIs. Furthermore, in GeoSoRS, we assume the starting point that the only available information is an LBO product database, and there is no requirement for interactions between users, as is the case for example in GeoLife. To facilitate this, we propose the GeoSoRS architecture in this chapter, that is designed to:

1. extract knowledge about a user from his trajectory data and existing social media profile;
2. extract knowledge about the region an LBO is located in, and;
3. combine this knowledge to extract recommendations.

In the GeoSoRS architecture, information is processed in three steps: (1) data collection, (2) data enrichment, and (3) recommendation extraction. For each of these steps, we introduce the required components, and for each component, existing work is discussed and pointers are provided to where this component is discussed in more detail in the remainder of this thesis.

This chapter is further structured as follows: related work is discussed in Section 2.2, information collection is discussed in Section 2.3, information enrichment is discussed in Section 2.4, the selection of recommendations is discussed in Section 2.5, an overview of the designed architecture is given in Section 2.6 and Section 2.7 presents the conclusion.

2.2 Related work

GRS research is at the intersection of three research areas: geographic information science, online communities, and recommender systems. Several GRS architectures have readily been introduced, each with its own strengths and drawbacks. Zheng et al. presented the architecture for GeoLife 2.0 in [95]. GeoLife 2.0 is a GPS-data-driven social networking service where people can share life experiences and connect to each other with their location histories. Their architecture is based on collaborative filtering, and has components for user similarity detection, trajectory analysis, and recommendation selection. The social aspect in GeoLife is the possibility to connect with other users. GeoLife’s architecture uses GPS traces only to create a user profile. This user profile is used both for location recommendation and to connect with similar users. In GeoLife’s architecture, no components are available for information collection from external sources, and the availability of a filled POI database is therefore a prerequisite. In collaboration with Bao and
Mokbel, Zheng proposed another architecture in [10], where they attempt to overcome the data sparsity problem. This architecture however, does not provide a solution for data collection, and requires users to check-in to locations.

Chaw et al. introduced GeoSocialDB: a holistic system providing three location-based social networking services, namely, location-based news feed, location-based news ranking, and location-based POI recommendation [21]. The latter refers to POI recommendation, potentially based on reviews by connected users. While the services of GeoSocialDB are thought to be implemented as query operators inside a database engine, the proposed system is rich enough to be considered a complete recommendation engine. GeoSocialDB extracts recommended news items and places based on geo-tagged news messages, user profiles, and POI ratings. A user can submit new news messages, user profile updates or POI ratings. The user profile contains personal information, e.g., identity and contact information, a list of friends, and preferences for the location-based news ranking service, and is maintained by the user. Just as is the case for GeoLife, users in GeoSocialDB have the option to connect with each other.

Papadimitriou et al. introduced a GRS called GeoSocial Recommender System, where users can get recommendations on friends, locations and activities [72]. In Papadimitriou’s system, a user profile consists of check-ins and friends. The three types of recommendations are extracted through a tensor reduction of a 3-order tensor, containing the user, location and past activities. As this is a form of collaborative filtering, Papadimitriou et al. reported that data sparsity became a problem upon evaluation.

Gupta et al. finally, proposed MobiSoC: middleware that enables mobile social computing application development [40]. In contrast to the aforementioned approaches, its architecture contains components for data collection of trajectories, POIs and information on users. An API is proposed to facilitate the development of mobile social applications in a generic way, using calls such as getCommonSocialContacts() to get the mutual friends of two people or getPeopleAtPlace() to get the people currently present at a specified location. The MobiSoC architecture is designed to provide support for the creation of an a recommendation engine, but does not contain one.

2.3 Information collection

Information is collected in GeoSoRS from five information source types: authoritative data, knowledge-bases, internal UGC, web content, and social media. Which data is collected, depends on the contents of the sixth information source: the product database. The geoprofile and user profile schemas determine for a large part which information is relevant to collect from other information sources, while certain components in the informa-
tion enrichment phase also require additional data to be collected for their analysis.

2.3.1 Authoritative data collectors

Authoritative data can be used as an information source of (typically) high quality on POIs or regional characteristics, such as demographics or regional climate. In GeoSoRS, we use authoritative data for enriching data collected through web harvesting, as discussed in Section 4.3, where we use an authoritative web feature service (WFS) to geocode collected POIs. This detailed information leads to more accurate insight into the POI visit behavior of users, as described in Chapter 5 and Chapter 6.

2.3.2 Knowledge base connectors

A public knowledge-base, such as OpenStreetMap, DBpedia, Freebase, GeoNames or YAGO, forms another important data source type for GeoSoRS. OpenStreetMap can be used as a starting point for POI sets, although inspection of this data set showed that this information is rather scarce, and not always up-to-date. However, its building polygons, that are less prone to be out-of-date, can be used for trajectory analysis, as we show in Chapter 4. DBpedia also contains many geographic references, but this information only contains items with a certain historical value, and typically not information on local businesses. A more useful application of the DBpedia data for GeoSoRS is the possibility to find all entities that are related to another, specified, entity, as we show with our query expansion technique in Section 7.4.

2.3.3 User-generated content collector

Internal UGC can be explicitly provided, through a POI rating & review system, or implicitly, through the ordering of products or booking of services. While the latter does not provide information on the user’s experience, it does give an indication for the likeliness of other people choosing that or a similar product or service. A special type of UGC is VGI. The frequency of visits and/or time spent at certain locations says something about the preferences and needs of a person. While we consider the challenges of collecting accurate trajectory data efficiently outside the scope of this thesis, automated trajectory analysis and enrichment is discussed in Section 2.4.1 and Chapter 5 and Chapter 6.
2.3. Information collection

2.3.4 Web data harvesters

Collecting public information from the web minimizes both costs and dependencies on data suppliers. Web data extraction systems are designed to facilitate this and are defined by Baumgartner et al. as software extracting, automatically and repeatedly, data from web pages with changing contents, and that delivers extracted data to a database or some other application [12]. For a recent survey on this subject, please refer to the work of Ferrara et al. [31]. In this thesis, we will focus on the concept described by Ferrara et al. as a web wrapper: a procedure, (..), that seeks and finds data required by a human user, extracting them from unstructured (or semi-structured) web sources, and transforming them into structured data, merging and unifying this information for further processing, in a semi-automatic or fully automatic way. According to Ferrara et al. a web wrapper typically goes through a life-cycle of: (1) generation, (2) execution, and (3) maintenance. In Chapter 3, we introduce a scraper that is specifically designed to minimize the effort of exactly these three phases, especially for a large number of websites.

2.3.5 Social media information collectors

Social media form a special subcategory of web content, as this content type also provides detailed information on the preferences of a single user, while regular web content is typically not limited to a specific user. The pages people like on Facebook, and all similar behavior on other social media, tell us something about what people want to be associated with. On social media, users can explicitly or implicitly review an item. We define a review as an indication of preference of an object, in a textual, numeric or boolean way, or a combination thereof. On Facebook for example, people can review an organization with a combination of a star rating and (optionally) a description of their experience, or they can like their Facebook page. We consider the first example an explicit review: a review with clear intent to inform other users, and the second example an implicit review: a review that is derived from content that was (most likely) not intended to inform other users. The latter often occurs when people simply want to share their recent activities with their social media peers, through for example a picture or a status update. A more advanced way of collecting reviews from a social media account, is to analyze the user’s messages on his or someone else’s bulletin board (such as Facebook’s timeline). This involves information retrieval techniques, to extract POIs from these messages, such as described in [42], as well as the detection of the corresponding sentiment, similar to [83].

A second type of information from social media that can be used in GeoSoRS, are the social medium pages on a place or an organization located near to one or more LBOs. A social medium page is a reference to a real-world entity on social media, often intended to inform about and/or increase the popularity
of this entity. A social medium page on a place or an organization often has a reference to its address or even geographic coordinates. While the information on places or organizations used to be very limited, the current trend of awareness for the importance of a positive online presence has led to an increase in detail and accuracy of provided information over the past years. This is especially the case for mainstream social media, such as Facebook. By now, this information could also be useful for POI collection, and thus for populating geoprofiles of LBOs.

Social medium profiles have been used in both generic and domain-specific RSs. Examples of generic social media-based RSs are Fijalkowski and Zatoka’s (unnamed) e-commerce architecture [32], Guy et al.’s Lotus Connections-based people-based recommender [41], and He et al.’s social network-based recommender system. Examples of domain-specific social media-based RSs are Bu et al.’s music RS [16] and Bonhard and Sasse’s Facebook-based movie RS [14]. The first step in using social media data in a RS is to connect his user profile in the RS to his social medium profile, and to extract the relevant content, such as liked pages, visited places, and timeline posts that potentially say something about a person’s interests. In Chapter 7, where we introduce our implicit review-based recommendation engine, we briefly touch upon the collection of social medium profile data. Since privacy naturally plays a large role when dealing with social medium profile data, we recommend the discussion of Zimmer et al. on this topic [100].

2.4 Information enrichment

Information enrichment is used to automatically analyse and combine available information. When necessary, these components can interact with information collection components to initiate additional information collection. The analysis approaches we use in GeoSoRS are trajectory analysis, quality assessment, and social graph analysis.

2.4.1 Trajectory analyzer

The trajectory analyzer is used to extract semantically meaningful information on the preferences and needs of a user from the VGI output of the UGC collector component. The places that a person visits, provide information on which LBO a person might be interested in. For example: a person visiting a school every morning probably drops off a child there, and is thus more likely to book a child-friendly holiday home, or to buy a house near a school. Extracting visited POIs is a challenging task, especially due to impreciseness of trajectories and loss of signal inside buildings. This topic has been researched already for example by Alvares et al. [4], Palma et al.
2.5 Recommendation selection

Their work and more is described in detail in Section 5.2. As discussed in Chapter 1, in GeoSoRS we attempt to solve this problem in three steps: (1) collection of POIs from the web, (2) conversion of POIs to POLOIs, and (3) matching trajectories with POLOIs. The trajectory analyzer has both the role of the second step, discussed in Chapter 4, and the third step, discussed in Chapter 5.

2.4.2 Quality inspector

The goal of the quality inspector is to filter out irrelevant, imprecise, untrusted or outdated content. UGC is known to be regularly imprecise, and many online reviews lack usefulness, or even trustworthiness. Coping with the possibility of imprecise content on elsewhere unmentioned items can be done by rating the accuracy of content sources. Such a rating can for example be based on a comparison of the source’s content with other sources, and the type of source (e.g. the website of a restaurant chain is more likely to provide more accurate information on their restaurants than the yellow pages). Chai et al. provided an overview of purely UGC-based quality assessment systems in [19]. In this thesis, we focus on quality assessment based on trajectory data: in Chapter 6, we show how behavioral patterns can be extracted from trajectory data, and how these patterns can be used to predict the UGC quality.

2.4.3 Social graph analyzer

Social networks can be represented as a labeled graph, called the social graph [54, 77]. Hidden relations can be derived from the social graph using regular graph theory, such as the approach by Roth et al. [77]. Clusters of friends can be detected from this as well, as shown by Cazabet et al. in [18] for example. Konstas et al. used the social graph to extract recommendations based on the opinions of connected social medium users in [54]. Although we acknowledge potential in propagating interests between users, especially with a suitable user similarity function, we limit ourselves in this thesis to the interests extracted directly from the user itself. In Chapter 7, we use the relations between a social medium user and his liked social media pages to extract interests.

2.5 Recommendation selection

The recommendation selection performed in GeoSoRS is based on finding a match between a user profile and a geoprofile. Other characteristics of the LBO, such as price and size, are then used for filtering.
2.5.1 Profile matching

Profile matching, as we use it in GeoSoRS, is the process of linking a user profile to a product profile, based on common grounds. The profile matching idea is illustrated at a high level in Figure 2.1: a user has interests that are potentially met by an LBO.

![Diagram of profile matching](image)

Figure 2.1: Profile matching based on common interests. Interest sources can be trajectory collections or social media accounts.

In Figure 2.2, we give an example of what types of information could be present in a geoprofile of a holiday home in Greece, and how this information can be used to relate it to a specific user. Multiple paths lead from the user to the holiday home, through matching elements in the user profile and the geoprofile.

![Diagram of profile matching between user and holiday home](image)

Figure 2.2: Profile matching based on common interests between the geoprofile of a Greek holiday home and a user with a Facebook account.

Formally, the set of recommended products (i.e. LBOs) based on profile matching are given as:
\[ R_{pm}(u, I, P) = \{ p \mid i \in I, p \in P \land \text{hasInterest}(u, i) \land \text{meetsInterest}(i, p) \} \]

where \( u \) is the user, \( I \) the set of interests, and \( P \) the set of products. \( \text{hasInterest} \) and \( \text{meetsInterest} \) are functions that are based on the available information in the user profile and geoprofile (contained in the LBO profile). A ranking of the matches can be based on characteristics of this graph, such as the number of paths or number of common interests, or even by creating an aggregate function on a weighted graph.

2.5.2 Filtering

Filtering is applied to make a selection of the products of the user’s interest, based on conditions supplied by the user inputs, or known search behavior from the past. The filtering function can be defined using boolean algebra or probabilistic logic. In the case of the holiday home broker, a user-filled filter contains for example the start date and end date of the vacation. The resulting set of recommendations \( R_f \) for a user \( u \) under filtering condition \( f \) as the intersection between the products for which both the filtering condition and the profile matching function hold:

\[ R_f(u, I, P) = \{ p \mid p \in R_{pm}(u, I, P) \land \text{matches}_f(p) \} , \]

where \( \text{matches}_f \) is a function that is defined by filtering condition \( f \).

2.6 Architecture overview

Combining the discussed information collection, information enrichment, and recommendation selection phases leads to the GeoSoRS architecture in Figure 2.3. The six arrows on the left represent the sources of data: the
product database, authoritative data, knowledge-bases, internal UGC/VGI, web content, and external social media content. In the information collection phase, this information is transformed to basic user profiles and geoprofiles for the LBOs. In the information enrichment phase, additional analysis is done to create enriched profiles, with a semantic meaning that is suitable for the recommendation selection phase. In that last phase, recommendations are extracted based on the discussed profile matching and product filtering approach.

2.7 Conclusion

This chapter presents the architecture of GeoSoRS, consisting of three phases: information collection, information enrichment, and recommendation selection. For each of these components, existing approaches are discussed and possible (alternative) solutions are proposed and/or pointers are given to a more detailed discussion elsewhere in this thesis. Contrary to existing approaches, the GeoSoRS architecture contains both data collection and recommendation components. Furthermore, it contains advanced components for travelling behavior analysis and UGC quality analysis, that are discussed in the remainder of this thesis.
Abstract

In this thesis, we show how the interests of users can be collected and used for recommendations, from trajectory data and social media profiles. In this chapter, we focus on the first step of interest collection from trajectory data: POI data collection. Rather than buying expensive POI data, in GeoSoRS, POI data is collected from the web. We discuss the construction of a web scraper that collects information, without the need for reconfiguration in response to changes of the HTML structure on these websites. We present the NeoGeo scraper, which wraps both existing and novel algorithms in components to let an application developer build scraping workflows. The scraper allows the concatenation of components based on recent web site inspection algorithms to build scraping workflows that are robust against interface changes of the information sources. The Dutch Yellow Pages are used as an example of such an information source to demonstrate how information retrieval (IR) algorithms can be combined to automatically and robustly collect POIs.

3.1 Introduction

In this chapter, we lay the foundation for interest collection based on trajectories. Where people go, says a lot about their preferences and needs. To obtain knowledge about where people like or need to go, the places visited by the mobile device owner need to be recognized. As discussed in Chapter 2, most conventional trajectory analysis approaches (e.g. [97]), use the trajectory data to detect where POIs are located, by finding those locations where a significant amount of time is spent. In GeoSoRS, we take a different approach: we collect trajectories on one hand, POIs on the other hand, and use the algorithm discussed in Chapter 5 to match these. Using this approach for POI collection, in contrast to detecting POIs from trajectory data, gives us knowledge on what kind of human activity or interest the location can be associated. In this chapter, we focus on POI collection, using the NeoGeo scraper, that is introduced here. The contributions discussed in this
chapter are therefore two-fold: (1) the introduction of the NeoGeo scraper technology, and (2) a demonstration of its application to scraping POIs from the web, using minimal resources and limited configuration efforts.

Ferrara et al. identify five main challenges in web data extraction in [31]. We focus on the first and fifth (which from here on we will call our second challenge) challenge they identify. The first is providing a high degree of automation by reducing human efforts as much as possible, and the second challenge is the fact that a web data extraction tool has to routinely extract data from a web data source which can evolve over time. Currently commercially available scrapers, such as Visual Web Ripper [87], rely on XPath configurations that are used to detect where the relevant information is located. These XPaths are entered manually, by a developer, or through a visual inspection of the page by a developer, that is translated by the scraper to an XPath configuration. The main drawback of this type of configuration, is that they are often outdated as soon as the website interface is changed. This makes this type of configuration time-consuming, especially with an increasing number of scraped websites.

We address these challenges through the usage and creation of web inspection algorithms, such as the Search Result Finder by Trieschnigg et al. [86]. The use of such techniques that mimic human cognitive skills helps us to find the relevant content on a web page automatically. The only configuration that is still required is the definition of the steps the website consumer (in this case the scraper) is supposed to take, which is defined as the scraper workflow. A scraper workflow is a concatenation of such algorithms that step-by-step leads the scraper to the relevant content on the website. The intention of scraper workflows is that they are robust to interface changes, and also are re-usable for different websites that follow a similar flow.

Another challenge discussed by Ferrara et al. is the volume of data that has to be processed in a relatively small amount of time. While we do not focus on the throughput speed, we do put emphasis on the potential to scrape multiple sources simultaneously, by reducing the required resources for a single web extraction. For scraping the Dutch Yellow Pages for the mid-sized city of Enschede, as we did for the validation of the NeoGeo scraper, a total of 33,623 pages were accessed. Scraping this website for a larger number of towns and/or scraping multiple sources will increase this number rapidly. Therefore, we introduce a pipeline mechanism. The pipeline mechanisms allows scraping components to interact in such a way that a depth-first crawl is performed. This ensures that a minimal amount of pages is in memory, while downloading and parsing pages only once, thus reducing resource requirements. Other solutions, such as efficient querying of web forms as discussed by Nelson et al. [68] and Khelghati et al. [53], can be used to add feedback to the pipeline, but in this chapter we focus on the general pipeline principle.
The NeoGeo scraper is built in Java, and uses the HTMLUnit and Hibernate libraries for web and database interaction, respectively. The project is set up as a Maven project, and is released open source under the FreeBSD license on GitHub.

This chapter is further structured as follows: related work is discussed in Section 3.2. In Section 3.3 the NeoGeo scraper is presented, and in Section 3.4 we explain how the NeoGeo scraper can be used for POI scraping. Section 3.5 finally contains our conclusion and gives directions for future work in this field.

### 3.2 Related work

As the amount of public information on the web increased from the mid 1990s onwards, the amount of scrapers to collect and bundle this information increased with it. In 1997, Kushmeric defined a wrapper as: a procedure for extracting a particular resource’s content [55]. He also introduced the concept of wrapper induction, a technique for automatically constructing wrappers. In his thesis, he explains how to create and select useful wrappers in three steps. In our work, we aim to avoid having a single wrapper for each information resource, but rather we aim to re-use scraper workflows for different resources. The encountered problems are very similar though. Ashsish and Knoblock introduced an alternative approach to the wrapper generation problem in [7] that required less knowledge about the structure of the scraped objects. Gruser et al. presented a toolkit in [38] that uses a wrapper capability table to determine which URL constructor and HTML extractor shall be used for each input and output type combination. This is similar to our approach of specifying a workflow for each website, but assumes that only one single wrapper inducer is used per source. For a complete overview of these and other web data extraction methods up to 2002, we recommend the survey by Laender and Ribeiro-Neto [56].

In 2003, Wang and Lochovsky presented DeLa, a system to extract information from the deep web that uses a wrapper inducer to detect and annotate the retrieved data [88]. At the same time, Liu et al. demonstrated in [61] that their method using the HTML tree to detect the fields of one or more objects on one page was much more accurate than existing techniques. All of these techniques laid a strong foundation for data extraction from the web, while proving the usefulness of wrapper inducers several times. Baumgartner et al. provide an overview of the evolution of web scrapers in [12], and define the following five tasks of a web data extraction system:

1. web interaction;
2. support for wrapper generation and execution;
3. Point-of-interest collection

3. scheduling for repeated application of previously generated wrappers;
4. data transformation, and;
5. delivering the resulting structured data to external applications.

The most recent survey of web data extraction techniques is the one by Ferrara et al. which was mentioned already in the introduction of this chapter [31]. In this survey, techniques, existing systems, and applications of web data extraction systems are discussed up to 2014. Many of the techniques that are discussed, are suitable to be implemented as components of our workflow-based scraper. An example of such an implementation of an existing technique is the SEARCH RESULT DETECTION component discussed in Section 3.3, which is based on the work of Trieschnigg et al. [86].

3.3 NeoGeo scraper

Information on the web is presented to users in an interface easy to interpret by humans. Users follow a path to the information they need, by filling out forms or clicking on the proper links. For example, when a user searches for a local business in the Yellow Pages, he/she is required to fill out a search field with two forms: (1) the category of the business, and (2) the name of the town, as illustrated in Figure 3.1a. The user is then presented with a list of results located in or nearby the specified town in that category, as illustrated in Figure 3.1b. Then, the user selects one of the items, and is presented with a page filled with more detailed information on that specific business, as illustrated in Figure 3.1c. If we abstract this to a higher level of user interactions, we obtain the user workflow in Figure 3.2. The user starts at the search page, performs a search action, and lands on a result page. After possible pagination through such pages, the user selects an item to view a detail page.

Since scrapers intend to visit all detail pages related to the search goal of the scraper, typical scraper behavior deviates slightly from that of users: (1) scrapers perform multiple searches, (2) scrapers visit all pagination pages, (3) scrapers visit all detail pages, and (4) post-processing of the information (such as data structuring and storage) is required. This alternative interaction is depicted in the workflow of Figure 3.3.

To facilitate web scraping, several algorithms are required to inspect and/or navigate the websites. For the search result page for example, two inspections are necessary: (1) the detection of search results, and (2) the detection of the pagination links. The search results then still need to be inspected further, to detect the link to the detail page within that search result. In the NeoGeo scraper each such inspection or navigation algorithm
is wrapped in a ‘scrapelet’. In the sequel of this chapter, we will use the term ‘scrapelet’ to stand for a scraper component process that is generic and re-usable as part of a scraper workflow. We will identify a number of scrapelet classes, each of which typifies the function/responsibility that a scrapelet has in the workflow that it is part of. Scrapelets have the characteristic that they are easy to recombine, and thus to re-use. Each scrapelet either: (1) inspects the document object model (DOM) tree of the HTML page, (2) selects from this tree, (3) returns a new HTML page, or (4) performs post-processing on the collected information, that is not directly related to the HTML structure anymore (such as geocoding an address, or storing the information in the database). The goal is to create a chain of scrapelets that
becomes a *scraper workflow*. If the methods used in the scrapelets for page inspection and content extraction are generic enough, scraper workflows are reusable for websites with similar workflows. Using automated scrapelets to find XPaths, rather than finding these manually, ensures that the scraper workflow (1) does not need to be updated upon each website update that
changes the XPath, and (2) can be reused for similar sources. Two websites have similar workflows when the same components, in the same order, both lead to the desired scraping result.

A typical example of a generic scrapelet for XPath detection is **SEARCH RESULT DETECTION**, for which the result is illustrated in Figure 3.4. This scrapelet is based on the **SearchResultFinder** by Trieschnigg et al. [86], and mimicks human cognitive skills to inspect a web page that contains search results based on visual and data clues: first it generates a list of candidate XPaths. For each of these, it calculates a score, based on several features, such as pixel area and the grid that the resulting elements are located in, as illustrated in Figure 3.5 and uses this to rank the candidates. The red blocks from Figure 3.4 represent the result for the top ranked XPath.

Besides the (reusable) scraper workflow, the scraper needs to be configured for each information source (i.e. website) with a starting URL and a model object class, which is to be filled with the found information. The webpage corresponding to the starting URL is fed as input to the first scrapelet.

### 3.3.1 Input and output typing of scrapelets

To ensure proper concatenation of scraping components, each component has a fixed input type and output type. Possible input and output types are: **HTML PAGE**, **DOM NODE**, and **MODEL OBJECT**. In the first scraping step of the Yellow Pages example, we perform a search action on the search page. The input is the page containing the search form, and the output is the search result page. In this case, both the input and output are of the type **HTML PAGE**. Further in the workflow, the search results are extracted from the search result page. This step, called **SEARCH RESULT EXTRACTION**, has

![Figure 3.5: Ranked candidate XPaths for the SEARCH RESULT DETECTION scrapelet, based on scores.](image)
3. Point-of-interest collection

Figure 3.6: NeoGeo scraper workflow example; scrapelets with matching input and output types can be concatenated.

HTML PAGE as the input type, and DOM NODE as the output type. In the NeoGeo scrapelets, input and output types are always single elements, not collections, as will be explained later.

3.3.2 Scraper workflows

Some websites require search actions to be taken, others readily provide a list with all the information, and yet other websites provide pagination on the search result pages. To reuse scraper functionality at a high abstraction level, scraper workflows are defined as concatenations of scrapelets to perform exactly those tasks relevant to that specific website. Since websites often have similar workflows, the set of workflows is typically considerably smaller than the set of websites to be scraped, especially when the focus is on a single domain only.

In Figure 3.6, the workflow of scraping the Dutch Yellow Pages is shown. In this Figure, the input and output types are illustrated by different symbols. The HTML PAGE type is represented by a straight line on the respective input or output side, the DOM NODE type by a single triangle, and a MODEL OBJECT by a double triangle. The steps of Figure 3.3 can be recognized in the scraper workflow by the separating dashed lines. For each node of Figure 3.3, a sequence of scrapelets is introduced in the scraper workflow of Figure 3.6. To illustrate the possibility to reuse this workflow: it contains no elements specifically developed for the Yellow Pages, and can be used to scrape other sources of local business information (e.g. restaurant review sites) just as well. We have tested each developed component (through JUnit unit tests) on multiple websites, all of which contained information on POIs, such as the IKEA store finder and the Dutch restaurant review website iens.nl.
3.3.3 Messages

Throughout the workflow, all output elements are constructed as a SCRAPER MESSAGE object. Each message consists of a body (corresponding to the output type of the preceding scrapelet), and a collection of properties. Using a message as a wrapper allows a scrapelet to add properties describing the output of its inspection. These properties can be used by scrapelets later in the workflow. The SEARCH RESULT DETECTION and PAGINATION DETECTION blocks from Figure 3.6 for example add the respective XPaths as properties to the scraper message. The SEARCH RESULT EXTRACTION block uses both these properties to extract the next search result.

3.3.4 Scrapelet types

In the NeoGeo scraper, we distinguish between the following scrapelet types:

**Initiator**

An *initiator* starts the workflow. For each data source the URL is read from the web source configuration (which can be loaded from the database or programmatically using the WEB RESOURCE class), and the initial web request is placed.

**Annotator**

An *annotator* inspects (a fraction of) a page, and adds at least one property to the scraper message. An example is the SEARCH RESULT DETECTION task in Figure 3.6. This scrapelet takes an HTML PAGE as input, detects the search results on the page, and annotates the message with an XPath expression leading to the search results. The scrapelet output is still the entire HTML PAGE, to allow other scrapelets to carry out similar detection tasks, such as PAGINATION DETECTION to detect any clickable elements (e.g., buttons, links) referring to other search result pages.

**Extractor**

An *extractor* uses the information of prior annotators to extract the relevant pieces of content. An example is the DETAILED INFORMATION EXTRACTOR, which uses the annotation of the DETAILED INFORMATION DETECTION task. Separating the extractor scrapelet from the annotator allows us to carry out several annotation tasks, before a selection is made.
3. Point-of-interest collection

**Buffer**

A buffer is a special kind of extractor, that has the task to pass one piece of content at a time, while receiving multiple pieces at once. An example is the SEARCH RESULT EXTRACTION task, which gets a page with multiple search results, and possibly a pagination button, and then passes on one result at a time to the DETAIL LINK DETECTION task.

**Transformer**

A transformer is used to map one structure onto another. In the example of Figure 3.6, the MODEL MAPPING task transforms a DOM NODE into a MODEL OBJECT.

**Enricher**

An enricher carries out a task after the information is collected from the web, but before it is stored in the database of the consuming application. An example is the GEOCODER scrapelet. An enricher outputs the input object with some additional information. This allows enrichers to be used consecutively on the same object.

**Persister**

A persister stores the collected and enriched data in the database or another type of storage, and is typically the last element of the workflow. The STORAGE scrapelet at the end of Figure 3.6 is an example of a persister block.

The types above represent information handling abstractions of currently available scrapelets, but supporting more sources of content, such as multimedia, will lead to more scrapelet types.

### 3.3.5 Pipeline

Dynamic web pages like the Yellow Pages contain information on large numbers of items, and the amount of possible search result pages, including pagination, is in the millions. Since it is not feasible to load all these pages resulting from intermediate steps in main memory of an ordinary server, and it is equally undesirable to load or parse pages multiple times, a pipeline system is introduced. In this pipeline mechanism, inspired by database architectures, each scrapelet requests only the next output item.

---

2In the process of scraping the Dutch Yellow Pages for the mid-sized city of Enschede, we encountered around 6,000 search result pages including pagination. Extrapolating this to search for all Dutch cities (754 after eliminating suburbs) leads to around 4.5 million search result pages.
from the preceding scrapelet, as illustrated in the sequence diagram of Figure 3.7. Furthermore, visited pages are cached in the database, and a list of visited pages are kept in main memory during each scraper run, to avoid accessing the same page through multiple routes.

![Figure 3.7: Pipeline mechanism: each scrapelet requests the next item from the preceding one.](image)

![Figure 3.8: Pipeline mechanism with buffering scrapelet B': not every request results in a request to the preceding scrapelet.](image)

In Figure 3.7, a workflow consisting of three scraping components, is concatenated as A → B → C. The initial information request is placed by the last component, C. It requests the next output item from B. Since B does not have any information yet, B requests the next output item from A. A now determines its next output item, which is used by B to determine its own. This item is now passed on to C that performs the final step. This is repeated until A returns a NULL value, indicating end of output. This causes B to return NULL, and so on. If A is for example the SEARCH ACTION task from Figure 3.6, this means that only one search action is carried out at a time, rather than performing all searches at once, and then moving on to step B, which would be the processing of all search result pages. Because of this pipeline mechanism, in the example workflow the scraper has in main memory at any point in time at most: one search page, one search result page (containing the search result DOM NODE), and one MODEL OBJECT.
In the special case of a buffering scrapelet, the sequence diagram slightly deviates, due to the built-in queuing mechanism. If we replace scrapelet \( B \) with a buffering scrapelet \( B' \), we obtain the sequence diagram of Figure 3.8.

### 3.3.6 Architecture

Combining all the described features and components leads to the architecture presented in Figure 3.9. The top layer is formed by the scrapelets. Ideally, an application developer who uses the NeoGeo scraper, only needs to add components specific for his scraping domain at this level. An application developer interested in multimedia content could for example add a scrapelet to download such content, or even a text-to-speech scrapelet. All scrapelets rely on the supporting infrastructure, and are dependent on the features provided by the pipeline system, and the web source configurations. Also, a page rendering service is provided for algorithms that rely on the dimensions of rendered web page content, such as the pixel area feature in the \( \text{SEARCH\ RESULT\ DETECTION} \) scrapelet. These services are built on top of the NeoGeo web client, which is an extension of the \( \text{HTMLUNIT} \) web client, to add features like caching, a proxy pool and netiquette compliance.
(e.g. adhering to ROBOTS.TXT instructions and limiting load on the scraped server). The foundation is formed by HTMLUNIT for the HTML to Java conversion and Hibernate finally for the object-relational mapping (ORM) layer for database communication.

Figure 3.10: Processes can be started and stopped using the NeoGeo scraper admin panel.

3.4 POI scraping

In GeoSoRS, we are interested in POI scraping. To facilitate this, we need a scraper configuration, containing (1) a starting URL, (2) a scraper workflow and (3) a MODEL OBJECT class. We use the Dutch Yellow Pages homepage as the starting URL [36], and the workflow from Figure 3.6. All the example scrapelets in the scraper workflow from Figure 3.6 are readily available in the NeoGeo scraper. The scraper is then initiated through the NeoGeo scraper admin panel, illustrated in Figure 3.10. The starting point configuration is used by the Initiator, the SEARCH ACTION. The MODEL OBJECT class configuration finally, is read by the MODEL MAPPING scrapelet. This uses code reflection to automatically detect the field names and types of all variables. Annotations are used for aliases of field names. In Table 3.1 we show the attributes of the MODEL OBJECT class for POI scraping. Each field has a name, aliases, and a data type. For the field latitude, this would translate to the following Java code snippet in the MODEL OBJECT class:
3. Point-of-interest collection

```java
@Alias("lat")
private double latitude;

public double getLatitude() {
    return this.latitude;
}

public void setLatitude(double latitude) {
    this.latitude = latitude;
}
```

<table>
<thead>
<tr>
<th>Field name</th>
<th>Aliases</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>title</td>
<td>String</td>
</tr>
<tr>
<td>streetName</td>
<td>address,street</td>
<td>String</td>
</tr>
<tr>
<td>postalCode</td>
<td>zipcode</td>
<td>String</td>
</tr>
<tr>
<td>town</td>
<td>townName</td>
<td>String</td>
</tr>
<tr>
<td>phoneNumber</td>
<td>telephone,tel,phone</td>
<td>String</td>
</tr>
<tr>
<td>latitude</td>
<td>lat</td>
<td>Double</td>
</tr>
<tr>
<td>longitude</td>
<td>lon,long</td>
<td>Double</td>
</tr>
<tr>
<td>url</td>
<td>website</td>
<td>URL</td>
</tr>
<tr>
<td>imageUrl</td>
<td>logo</td>
<td>URL</td>
</tr>
<tr>
<td>category</td>
<td>type</td>
<td>String</td>
</tr>
</tbody>
</table>

Table 3.1: Data model for POI scraping

The model mapping block then inspects the detailed information DOM NODE for occurrences of the field names or aliases in DOM NODE attributes like id, class and itemprop. As an example, we give a simplified and translated snippet of the detailed information DOM NODE from the Yellow Pages, that forms the input of the MODEL MAPPING scrapelet:

```html
<div id="businessInfo" class="component">
    <h1 class="test_h1_detail boxHeader">
        <span itemprop="name">Hu’s Garden Restaurant</span> in Enschede</h1>
    <li class="phone customerContent">
        Call: <span itemprop="telephone">053-4333678</span>
    </li>
    <li id="fullAddress">Oldenzaalsestraat 266, 7523AG, Enschede</li>
    <li class="category">
        Category: </li>
</div>
```

34
The built-in MODEL MAPPING scrapelet is especially suitable for scraping websites that use HTML attributes extensively. Alternative model mapping scrapelets could be introduced to support also websites that make less use of HTML attributes. Jundt and Van Keulen for example used a manually annotated training set to detect the respective XPaths in [51].

Figure 3.11: POI scraping result for the centre of Enschede, The Netherlands. The colored dots represent the respective POIs, the grey polygons with black outlines the cadastral building data (normally unavailable, as discussed in Chapter 4). The actual scraped categories are more detailed, but mapped onto these four generic categories for illustration purposes.

We used NeoGeo for a POI scraping task of 603 POI categories for the city of Enschede. Our scraping experiment resulted in 27,311 POIs in 1,731 (Yellow Pages) categories, after inspecting a total of 33,209 pages. The thus obtained POIs not only have a name, an address and a coordinate pair,
but also a POI category that it belongs to, as illustrated in Figure 3.11. In this figure, we mapped the POI categories onto four generic groups for illustration purposes, but actually these are more detailed categories, such as shoe store, bakery, or movie theater.

3.5 Conclusion

Our objective in this chapter was to address the challenges of reducing the required effort for scraper development and maintenance. We introduced the NeoGeo scraper, a scraper that is based on workflows of wrapper inducers. The configuration effort for this scraper is limited to (1) a (potentially reusable) scraper workflow, (2) a starting point URL, and (3) a model object class. Furthermore, a pipeline mechanism plays a central role in the NeoGeo scraper to reduce the resource requirements. We showed an example workflow and model definition for POI scraping from the Dutch Yellow pages, and the results for running the NeoGeo scraper for the city of Enschede, The Netherlands. This POI set is used in the remainder of this thesis for the creation and validation of other GeoSoRS components.
POI to POLOI conversion

Abstract

Trajectories obtained through a mobile device, such as a smartphone, contain a vast amount of implicit information on the interests of the owner of the device. Collections of trajectories even provide insight into the popularity of locations, and the time spent at those locations. To obtain this information, the places visited by the owner of the device on such a trajectory need to be recognized. However, the location information on a POI as harvested from the web is normally limited to an address and a coordinate pair, rather than a region object describing its boundaries. To create a match with a trajectory, a region object representing this place, a polygon-of-interest (POLOI), is needed. We define a POLOI as “the area that is included in the boundaries of the object represented by the POI.” In the absence of expensive and hard to obtain detailed spatial data like cadastral data, we need to approximate this POLOI. In this chapter, we bridge this gap by presenting several approaches to approximate the size and shape of POLOIs, by integrating data from multiple public sources, and we validate these approximations against the cadastral data of the city of Enschede, The Netherlands.

This chapter is based on [25]

4.1 Introduction

In Figure 4.1 we show how a trajectory intersects with two POLOIs. Given the spatial and temporal overlap with the yellow polygon, it is our goal to automatically infer that the yellow polygon was visited, while the red one was not (i.e. a false positive in a purely intersection-based approach). To do so, we need to carry out both a spatial and a temporal inspection. The temporal inspection illustrated in Figure 4.2 uses temporal smoothing to remove noise from the trajectory, caused by for example GPS signal reflection, usage of a parking facility belonging to another property, etc. In this chapter, however, we focus on the spatial inspection, and explicitly
on how POLOIs, illustrated by the blue polygons in Figure 4.1, can be generated from a given POI set, such as the one resulting from the scraper in the previous chapter, and illustrated by the red dots in Figure 4.1. Cadastral data, provides us with such POLOIs, but this data is unfortunately often expensive or even impossible to obtain. This gap could be bridged with an approach to approximate such data. These approximations shall roughly describe the POLOI and be good enough to be used for trajectory matching. In this chapter, we present six novel approaches to approximate POLOIs, and validate these. We have chosen three publicly available data sources to facilitate the creation and validation of such approximation approaches:

1. a POI source;
2. a geocoding webservice;
3. a map data source.

Our contributions in this chapter are:

1. several approaches to approximate the shape and size of the POLOI;
2. a validation technique for the approaches;
3. a validation of these approaches against the cadastral data of the city of Enschede;
4. an interaction scheme to preserve the privacy of users, while allowing for the creation of personalized services based on trajectory analysis.

Figure 4.1: Trajectory matching based purely on POLOI intersection: the yellow polygon was actually visited, while the red one was not. Red dots represent POIs, blue polygons represent cadastral data for all nearby objects.
4.2 Related work

Existing methods for detection of visited locations in trajectories all rely on the availability of a set of trajectories for the region in which the POIs are located. Zheng et al. [98] introduced an approach to detect visited locations by analyzing the trajectories and discovering ‘stay points’. This approach assumes no prior knowledge about existing POIs. The coordinate pairs resulting from that process however, do not provide us with any contextual information on the stay points. Therefore, their algorithms can be used to detect the candidate coordinates from the trajectories for a trajectory annotator, while the POLOIs that result from the approaches in this chapter can be used to match these coordinate pairs with POIs for which contextual information is available.

Similar to the work of Zheng is the work of Ashbrook and Starner [6], whose approach for discovery of significant locations in trajectories is also based on the availability of trajectories. Contrary to the approaches presented in this chapter, theirs requires additional user input to obtain information on the semantics of the location. Another limitation of their approach is that the radius of the clustering circle has to be configured, rather than letting the algorithm automatically detect a suitable value from the data.

Gianotti et al. describe trajectory pattern mining in [34]. Their focus is on the combination of visited places. They acknowledge the need for
4. POI to POLOI conversion

Figure 4.3: Cadastral data with corresponding POIs. Red dots represent POIs, blue polygons represent cadastral data for all nearby objects.

POLOI detection, but assume that the set of POLOIs is provided as an input to their approach. The result of the approaches described in this chapter can therefore be seen as an input for their pattern mining algorithms.

A fully different approach to detect visited locations is presented by Greenfeld et al. [37], but their focus is on matching with road networks, rather than on matching with POIs.

Yan et al. [90] created SeMiTri, a framework for semantic annotation of trajectories, for both fast moving objects and people’s trajectories. Their POI annotation provides an uncertain answer to the question which POI was visited by analyzing nearby POIs, and then uses the categories of those POIs to detect a sequence of performed activities.

4.3 Used data sources

In our techniques, we make use of three data sources for the approximation of POLOIs: 1) a POI source, 2) a geocoding service and 3) a map data source. In our experiment for the city of Enschede, The Netherlands, we used the Dutch Yellow Pages [36] as the source for POIs, the Dutch geocoding service from the “Nationaal Georegister” [67], and OpenStreetMap [69] for the map data.

Yellow Pages

The Dutch Yellow Pages website [36] contains information on all companies in the Netherlands, such as name, address, zip code, town, phone number, a
4.3. Used data sources

coordinate pair, opening hours, and the POI category (e.g. dentist, hospital, flowerist). This dataset is the one that resulted from the application example of the NeoGeo scraper from Chapter 3.

The Yellow Pages dataset demonstrates typical weaknesses of POI datasets:

1. POIs are point referenced, with no indication of the size of the object;
2. the latitude and longitude of the POIs are sufficient for display on a map, but lack the spatial accuracy to allow sensible proximity analysis, or even the 1:1 correspondence to parcel objects.

Geocoding Service

The “Nationaal Georegister” is the central Dutch geoportal that functions as a catalog to Dutch geodata. Amongst others, it offers an address geocoding webservice that geocodes addresses with high spatial accuracy. This precision provides a solid basis for the creation of POLOIs, and therefore this service was used to pre-process the POI data. The results of this geocoding process are illustrated in Figure 4.4, in which the green dots represent the coordinates from the Yellow Pages, and the red dots represent the POIs after the regeocoding process, and the blue lines match these two for each POI.

OpenStreetMap

The OpenStreetMap (OSM) initiative provides volunteered geographic information (VGI) for the entire world, with variable levels of detail geographically.
4. POI to POLOI conversion

Since this dataset roughly describes the shape of roads and buildings, it can be used to approximate the boundaries of the POLOIs. The PostgreSQL database resulting from the OSM2PGSQL tool for the region of Enschede contains four tables:

1. PLANET_OSM_LINE, containing line segments, such as unclassified ‘highway’ segments, cycleways, motorways, forests, and farms;
2. PLANET_OSM_ROADS, containing road segments (primary, secondary, motorway, etc.), which is a subset of PLANET_OSM_LINE;
3. PLANET_OSM_POINT, containing POIs of various kinds, such as bus stops, shops, traffic signals, and playgrounds;
4. PLANET_OSM_POLYGON, containing polygons of various kinds, such as buildings and forest segments (mostly outside the city).

The weaknesses of the OSM dataset for our purpose are:

1. the entities represented by the polygons in PLANET_OSM_POLYGON are different from our intended POLOIs. The buildings from this dataset frequently contain several POLOIs;
2. the scarcity of data in PLANET_OSM_POINT: the OSM dataset contains for example only 8 restaurants in Enschede, whereas the dataset of the Yellow Pages contains 269 restaurants in 14 different categories;
3. the information is frequently out-of-date. To illustrate this: the used OSM dataset from 2013 still contained bus stops in Enschede that were taken out of service in 2011.

4.4 Terminology

Approximating data by definition leads to successes and errors. To interpret the quality of our results, we use a few well-known concepts from the field of information retrieval (IR). The quality of search results, amongst others, plays an important role in the field of IR.

To compare search results, four categories of success and error are used: true positives, false positives, true negatives and false negatives. Analogous to this, we define four areas in the region around our POIs, when comparing our approximation results for a single POLOI ($POLOI_a$) with the cadastral data for that POLOI ($POLOI_c$):

<table>
<thead>
<tr>
<th>Intersecting $POLOI_a$</th>
<th>Not intersecting $POLOI_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intersecting $POLOI_c$</td>
<td>True positive (TP)</td>
</tr>
<tr>
<td>Not intersecting $POLOI_a$</td>
<td>False positive (FP)</td>
</tr>
<tr>
<td>False negative (FN)</td>
<td>True negative (TN)</td>
</tr>
</tbody>
</table>

To judge the quality of a search engine in IR, two metrics are commonly used to validate the search results: precision and recall [63]. Precision is
defined as “the fraction of retrieved documents that are relevant,” and recall is defined as “the fraction of relevant documents that are retrieved.” This can be translated to our domain, by defining precision as “the fraction of the approximated area which is correct,” and recall as “the fraction of the actual area which is correctly approximated.”

To formalize this, we define the true positive area of a single feature (i.e. POLOI) \( f \) as:

\[
TP(f) = POLOI_a(f) \cap POLOI_c(f)
\]

Precision and recall are then formalized for a single feature \( f \) as:

\[
\text{Precision}(f) = \frac{\text{area}(TP(f))}{\text{area}(POLOI_a(f))},
\]

\[
\text{Recall}(f) = \frac{\text{area}(TP(f))}{\text{area}(POLOI_c(f))}.
\]

A POLOI approximation that is large compared to the actual POLOI therefore has a high recall and a low precision, while a small POLOI approximation has exactly the opposite. The challenge is now to obtain good results for the measure(s) relevant for the application. To rank the results, these two numbers are often combined in IR using a harmonic mean called the F measure. In case precision and recall are equally important, the F measure for a feature \( f \) is formally given as [63]:

\[
F(f) = \frac{2 \times \text{Precision}(f) \times \text{Recall}(f)}{\text{Precision}(f) + \text{Recall}(f)}
\]

**4.5 Approximation approaches**

In this section, we propose six different approaches for POLOI approximation. For each of the approaches, we have created a visualization for the city center of Enschede. Blue areas indicate true positive parts of the approximations, yellow areas indicate the false positive parts, red shaded areas indicate the false negative parts, and white areas indicate true negative parts. Yellow areas with red stripes are a combination of a false positive for one approximated POLOI, and a false negative for another one. A visualization with many yellow areas therefore indicates low precision, one with many red shaded areas indicates low recall, and one with many blue and white areas indicates both high precision and recall. The white dots on the maps are the POIs after regeocoding.
4. POI to POLOI conversion

(a) VRD: Variable radius discs
(b) VRDB: Variable radius discs with boundaries
(c) VD (50): Voronoi diagrams
(d) VD (10): VD with size limitations
(e) VD+ (50): Voronoi diagrams with additional addresses (illustrated as white points without POLOI)
(f) Legend of the maps

Figure 4.5: POI to POLOI conversion approaches (1/2)
4.5. Approximation approaches

(a) VOSM: Voronoi diagrams of OSM polygons
(b) VOSM+: Voronoi diagrams of OSM polygons with additional geocoding (illustrated as white points without POLOI)

Figure 4.6: POI to POLOI conversion approaches (2/2)

4.5.1 Variable radius discs (VRD)

The Variable radius discs approach, illustrated in Figure [4.5a], approximates the shape of each POI as a disc around its geo-reference. The radius of the disc equals half the distance between the POI and its nearest neighbor POI. This approach is intended as a (rather naive) baseline approach, to put the results of other approaches into perspective.

4.5.2 Variable radius discs with Boundaries (VRDB)

The Variable radius discs with boundaries, illustrated in Figure [4.5b], is similar to VRD, but in addition, the line segments of the OSM PLANET_OSM_LINE table are used as natural delineations of the extents of the discs. This way, the overshoot of a VRD approximation to the other side of the street is removed. The objective for this approach is to improve precision, compared to VRD, while only suffering from a marginal decrease of recall.

4.5.3 Voronoi Diagrams (VD)

Voronoi diagrams are frequently used to divide space into regions based on a set of points [8]. This approach, illustrated in Figure [4.5c], divides the space into Voronoi regions by assigning each point in the space to the POLOI belonging to the nearest POI. Different metrics can be used to define which point is ‘nearest’, but we use Euclidean distance.

Since this approach creates a tessellation of the entire space, the approximated POLOIs are large compared to the actual POLOIs. This also causes the overlap between those two to be large, which results in high recall, but
also low precision. This extreme result can be moderated by limiting the extents of the polygons, to a maximum radius of, for example, 10 meters from the POI, as illustrated in Figure 4.5d. For our experiments, we have used the values 10, 25, and 50 meters. Since this reduces the overshoot, it will typically increase precision (as the ratio of correctly estimated area increases), while decreasing recall (as a limit to the radius also limits the maximum distance between the POI and the POLOI extent that can still be correctly estimated), compared to a regular Voronoi diagram. However, the chosen value is arbitrary, and any value fits better with certain classes of POI than other classes.

4.5.4 Voronoi diagrams with additional geocoding (VD+)

A problem with the creation of Voronoi diagrams is the scarcity of POIs, which causes POLOI approximations to be too large in areas with a low POI density. Voronoi diagrams with additional geocoding overcome this problem by geocoding all addresses that can be derived from the streetnames in the OSM dataset. For each streetname in OSM, we geocoded ‘all’ house numbers.1 We used these addresses with their respective coordinate pair as additional points while creating the Voronoi diagram, and then discarded their POLOIs. The results of this approach are illustrated in Figure 4.5e. Just as for VD, we limited the extents of the polygons to 10, 25, and 50 meters for our experiments. The objective of the addition of addresses is to have a strong positive effect on precision, while the negative impact on recall is small. Due to the high number of points and the associated increase of complexity of the creation of the Voronoi diagram, this approach is the most time-consuming, even when omitting the time necessary to geocode the additional addresses.

4.5.5 Voronoi diagrams of OSM polygons (VOSM)

OSM polygons contain buildings, which frequently contain several POIs. Voronoi diagrams of OSM polygons split these polygons into smaller ones, as illustrated in Figure 4.6a. Using the OSM polygons as a starting point for the POLOI approximation provides us with good approximations of outside walls of the building. The Voronoi technique for the creation of the boundaries inside the OSM polygons results in decent boundaries, especially for OSM polygons with many known POIs. Due to this combination, we expect VOSM to have both a fairly good precision and recall.

1We geocoded addresses for the odd and even sequence of each street separately, until a gap of 100 non-existing house numbers was found.
4.6 Validation

In this section, we present the validation results for all approaches of Section 4.5 using cadastral data from the city of Enschede, The Netherlands, as ground truth. This cadastral data contains two tables relevant for this purpose: one that contains the buildings and one that contains the parcels. A parcel may contain several buildings, as illustrated in Figure 4.7a, and a building may be built over multiple parcels, as illustrated in Figure 4.7b. Since the ultimate goal of these approaches is to provide a POLOI to recognize visited places from trajectories, we have chosen to use the parcel-building intersection as ground truth for our validation.

For this validation, we use the concepts of precision, recall, and F measure, defined in Section 4.4. The results for the entire city of Enschede are presented in Table 4.1. The F measure is calculated for each estimation.
4. POI to POLOI conversion

separately, and is therefore sometimes lower than the precision or recall individually.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VRD</td>
<td>0.321</td>
<td>0.811</td>
<td>0.303</td>
<td>0.187</td>
<td>1.000</td>
<td>0.271</td>
</tr>
<tr>
<td>VRDB</td>
<td>0.333</td>
<td>0.809</td>
<td>0.318</td>
<td>0.209</td>
<td>1.000</td>
<td>0.296</td>
</tr>
<tr>
<td>VD (10)</td>
<td>0.493</td>
<td>0.685</td>
<td>0.444</td>
<td>0.425</td>
<td>0.783</td>
<td>0.429</td>
</tr>
<tr>
<td>VD (25)</td>
<td>0.193</td>
<td>0.884</td>
<td>0.251</td>
<td>0.110</td>
<td>1.000</td>
<td>0.192</td>
</tr>
<tr>
<td>VD (50)</td>
<td>0.099</td>
<td>0.934</td>
<td>0.150</td>
<td>0.046</td>
<td>1.000</td>
<td>0.087</td>
</tr>
<tr>
<td>VD+ (10)</td>
<td>0.555</td>
<td>0.635</td>
<td>0.490</td>
<td>0.503</td>
<td>0.723</td>
<td>0.510</td>
</tr>
<tr>
<td>VD+ (25)</td>
<td>0.297</td>
<td>0.763</td>
<td>0.367</td>
<td>0.251</td>
<td>0.910</td>
<td>0.352</td>
</tr>
<tr>
<td>VD+ (50)</td>
<td>0.246</td>
<td>0.784</td>
<td>0.325</td>
<td>0.214</td>
<td>0.931</td>
<td>0.311</td>
</tr>
<tr>
<td>VOSM</td>
<td>0.559</td>
<td>0.728</td>
<td>0.571</td>
<td>0.589</td>
<td>0.773</td>
<td>0.591</td>
</tr>
<tr>
<td>VOSM+</td>
<td>0.741</td>
<td>0.621</td>
<td>0.625</td>
<td>0.783</td>
<td>0.664</td>
<td>0.661</td>
</tr>
</tbody>
</table>

Table 4.1: Validation results of singular approaches

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VD+ (10) ∩ VOSM</td>
<td>0.798</td>
<td>0.499</td>
<td>0.560</td>
<td>0.831</td>
<td>0.507</td>
<td>0.589</td>
</tr>
<tr>
<td>VD+ (25) ∩ VOSM</td>
<td>0.764</td>
<td>0.583</td>
<td>0.612</td>
<td>0.794</td>
<td>0.614</td>
<td>0.650</td>
</tr>
<tr>
<td>VD+ (50) ∩ VOSM</td>
<td>0.759</td>
<td>0.597</td>
<td>0.620</td>
<td>0.793</td>
<td>0.635</td>
<td>0.659</td>
</tr>
</tbody>
</table>

Table 4.2: Validation results of the best approach combinations

The average recall for VRD turns out to be quite high for our baseline approach VRD. This is due to the large number of POLOIs that have been approximated as larger than the entire parcel-building intersection, as can be seen from the median value of exactly 1 for recall. The precision of 0.321 provides a good basis to interpret the results of the other approaches. The results for VRDB only differ slightly from those of VRD. This is due to the relatively small number of VRD estimations that intersect with roads.

Due to the density of POIs in our dataset, VD(50) can be seen as Voronoï diagrams without a limit to the extent of the POLOIs. As predicted, VD(50) has a very high recall, however, its precision is much lower than the precision of even our baseline approach VRD. Limiting the extent to 10 meters indeed increases the precision, and also increases the F measure. A proper value for this variable depends on the density of POIs and the POI categories in the set.

The addition of addresses in VD+ has a positive effect both for the Voronoï diagrams with a limited extent and for those without. The effect
is stronger for those without, as it limits the overapproximation of the parcel-building intersections naturally.

The results for VOSM are better than those for the approaches discussed so far. Especially for buildings that are described in detail in OSM, and only contain one address, this approach scores well. This approach does not score very well in buildings with many addresses, but only a few POIs.

VOSM+ performs the best of all our approaches. The high average and median F measure shows that for most POLOIs this approach provides a precise approximation, while recalling a large part of the actual parcel-building intersection.

We also validated combinations of approaches to determine whether a combination could be found that yields better results than any singular approach. As can be seen in Table 4.2, the combination of VD+(50) ∩ VOSM yields the best results. That this combination performs similarly to VOSM+ is in line with our expectations, since VOSM+ combines the ideas used in VD+ ∩ VOSM.

Not only the combination of precision and recall is better for the VOSM+ approach than for the other singular approaches. Due to the low number of points inside each OSM polygon, the complexity of the Voronoi diagram creation is low, making this approach more scalable than for example VD+ (and thus VD+ ∩ VOSM). This advantage is the reason why we recommend VOSM+ over VD+ ∩ VOSM.

Another advantage of VOSM+ is its potential use for remote locations, where no other POIs nearby are known, while some of the other approaches heavily depend on dense data, to determine the size of the disc (VRD) or the border of the polygon (VD). The maximum polygon size (in the case of no other known addresses inside the OSM polygon) is always limited to the size of the OSM polygon, which limits the overshoot, even if no additional addresses can be found in the vicinity.

To determine which approaches yield better results in areas with high POI density and which ones in areas with low POI density, we divided the city into several cross-sections, using polygons of the OSM dataset. As can be seen in Figure 4.8, VOSM+ yields the best results for the three types of land use in OSM that contain the most POIs in our dataset. From this validation, it cannot be concluded that VOSM+ always yields the best results in residential areas, since this is strongly biased by the categories of POI that we chose as input for our algorithms. However, based on these results, it is possible to conclude that both for areas with a high POI density, as for those with a low POI density, VOSM+ outperforms (most) other methods, with a consistent high F measure. Only VOSM outperforms VOSM+ in the areas classified as “grass sections”, due to a small number of POIs registered in residential building blocks in such areas that are not subdivided into different parcels in the cadastral data. Most of these points
4. POI to POLOI conversion

Figure 4.8: F Measure by land use type

are also contained by residential polygons in the OSM dataset, for which VOSM+ generally performs much better than VOSM.

4.7 Privacy preservation

Tracking people’s locations is a privacy sensitive topic: most people do not want to be traced, and most definitely not 24/7. While one can think of several applications where it is beneficial to know a person’s exact location real-time, such as a mobile emergency application for elderly, most applications do not need to have this information real-time, or at this fine-grain level of detail. The latter applies also to a GRS, and therefore we need to preserve the privacy of users as much as possible, while still allowing the creation of a user profile for personalized services. An important step to privacy preservation is to mask time and location details. We therefore propose the following client-server interaction:

1. At steady intervals\(^2\), the client determines which points indicate a visit

\(^2\)Since user profiles are generally not needed continuously, and do not need to be up-to-
4.8 Conclusion

Our main objective in this chapter is to define several approaches for POLOI approximation and to validate these. We proposed six different approaches and a validation technique using the concepts of precision and recall, and presented the results of this validation.

Based on the validation, we conclude that “Voronoi diagrams of the OSM polygons with additional geocoding” (VOSM+) is the approach that yields the best results from the ones we propose. It recalls a decent portion of the actual POLOI and has very good precision. Moreover, this approach can be carried out for areas with low POI density without loss of accuracy, and, due to the relatively low calculation complexity, the calculation finishes within a reasonable amount of time.

The main drawback of the VOSM+ approach is the necessity of a geocoding service for accurate coordinates, and the dependency on the quality of the OSM data. The OSM data that was used in our experiments has a relatively high quality level. We expect both of these prerequisites to be fulfilled throughout Europe in the next years, the first one especially, in the light of the INSPIRE directive [48].

Of the combined approaches, results similar to the ones of VOSM+ are achieved by the union of the “Voronoi diagrams with additional addresses” (VD+) with the “Voronoi diagrams of OSM polygons” (VOSM), VD+ ∩ VOSM. However, this approach is computationally more complex than VOSM+, which makes this combined approach less scalable.

date at all time, we use a steady interval to mask the exact time the area was visited.
Finally, we also propose a client-server interaction which preserves the privacy of the user. This interaction uses techniques to mask the time and exact locations, while still allowing for the creation of a user profile.
Automated semantic trajectory annotation

Abstract

Now that we have shown how POIs can be collected from the web, and converted into POLOIs, we show in this chapter how these POLOIs can be used to detect which POIs were visited. The ultimate goal is to use that information for recommendation selection in our GRS. This conversion of geometric trajectory data into a semantic collection of visited points-of-interest makes the data more suitable for data mining. However, trajectory data is often imprecise. Especially the detection of indoor activities in urban areas is a challenging task. In this chapter, we propose a new algorithm for the detection of visited points-of-interest that performs well, even for this challenging type of activity detection. We demonstrate the strength of the algorithm by comparing it to three existing algorithms, using annotated trajectory data, collected through an experiment with students in the city of Hengelo, The Netherlands. Our algorithm, which combines multiple trajectory preprocessing techniques from existing work with two new ones, shows significant improvements.

This chapter is based on [24]

5.1 Introduction

The places a person visits regularly are a strong indication for personal preferences and needs. A person visiting a kindergarten each morning most likely has a young child, and a person visiting a concert hall on a regular basis probably has a more than average interest in music. This makes trajectory data a promising source to determine personal preferences and needs, as discussed in Chapter 2. Furthermore, trajectory data can be used to assess the trustworthiness of UGC, as discussed in Chapter 6: a negative review of a restaurant that the user visited in the past week can probably be trusted more than a negative review of a person who works at
5. Automated semantic trajectory annotation

a nearby competitor. Before we can perform such an analysis, we need to know which places were visited. This information can be extracted from the geometric trajectory data, by converting it into a collection of visited points-of-interest (POIs). This type of trajectory analysis has been discussed in the literature as stop detection, and we follow the definition of Yan et al. for a stop: a temporary suspension of the travel for some reason [92]. In this work, we are particularly interested in those stops that take place at a pre-defined POI.

Smartphones these days base the location on multiple information sources: WiFi positioning, (assisted) GPS positioning and cell phone tower locations. In this chapter, we focus on the detection of POI visits based on smartphone data, which does not reveal from which source the location was derived. This POI visit detection is a challenging task, especially for indoor activities in urban areas, since trajectory data is often incomplete, due to temporary inability to receive GPS signals from enough satellites, or imprecise, due to signal multipath. In the words of Alvares et al.: “to transform a sample trajectory into a semantic one (sequence of stops and moves) is not an easy task” [5], or as Yan et al. put it: “dense urban areas can have several different POIs. (...) Such large number makes it probabilistically intractable to infer the exact POI from imprecise location records” [90, 91]. Nevertheless, several attempts have been undertaken to extract POI visits from trajectory data. There are generally two ways to do this. The most common way is to detect POIs from slow movement over a longer period, and defining the locations at which this happens regularly as the POI set, as for example by Ashbrook et al. [6] or Zheng et al. [97]. A less common way is to match trajectories with a given, or specifically collected, POI set, such as the one by Alvares et al. [4]. The advantage of the latter approach, is that more information on the matched POI may already be available, such as a name, address, website, and POI category. This makes the approach with a given POI set more suitable for a semantic analysis of the visited POIs, which is why we focus on this form of POI visit extraction.

However, existing approaches have two important drawbacks: (1) they assume the availability of the GPS signal while residing at the POI, and (2) they do not take the accuracy of the GPS samples into account. The first drawback leads to the non-detection of indoor POI visits, while the latter leads to false positives for imprecise signals, as discussed in more detail in Section 5.2.

In this chapter, we present the trajectory annotation algorithm Point-of-Interest Extraction (PIE), illustrated in Figure 5.2, which is designed to overcome the aforementioned problems. In this figure, the circles indicate the trajectory samples. The green, blue and red areas indicate extracted POI visits. The differences between the meaning of the colors is discussed in Section 5.4.5. The PIE algorithm was designed especially for indoor activities, while using the GPS sensor of mobile devices. In this chapter,
5.1. Introduction

Figure 5.1: Trajectory annotation using Point-of-Interest Extraction (PIE). Green, blue and red polygons indicate the footprint of POIs that were extracted as visited. White and grey polygons were not extracted as visited. For green polygons proof was available of a visit, blue and white polygons were possibly visited, and red and grey polygons were definitely not visited in the real world.

we also present the results of a comparison with the existing algorithms of Alvares [4], Palma [71], and Rocha [76]. These three methods (especially the first two), are regularly used for POI extraction (for example by Xie et al. [89]), due to their availability as an extension for the data mining tool Weka [5, 76]. For this comparison, we set up an experiment with students of the University of Twente, who used their own mobile devices as they were invited to visit pre-defined POIs as part of a treasure hunt game in the city of Hengelo, The Netherlands.

The remainder of this chapter is structured as follows: Section 5.2 provides an overview of related work. The details of our approach are laid out in Section 5.3. Our validation method and results are presented in Section 5.4. Privacy considerations are discussed in Section 5.5. Section 5.6 finally, concludes the chapter.
5. Automated semantic trajectory annotation

5.2 Related work

In this section, we focus on the explanation and illustration of the approaches with which we compare our work: the approaches of Alvares, Palma, and Rocha, who (in collaboration with each other) tried to solve the same problem as we are, using three different approaches. We conclude this section with a short discussion of other related work.

5.2.1 Alvares’s IB-SMoT

Alvares’ Intersection-Based Stops and Moves of Trajectories (IB-SMoT) is the most straightforward of the three approaches: it intersects the trajectories with the polygons that represent the POIs, as illustrated in Figure 5.3a. The trajectory is split up into intervals that are either a stop at a POI, or a move between POIs. Those intervals during which the trajectory intersects with a known polygon are marked as a stop (illustrated by filled trajectory points in Figure 5.3a), and those intervals during which the GPS samples do not intersect with such a polygon are annotated as a move (illustrated by empty trajectory points). Besides a trajectory and a set of disjoint relevant polygons, this approach takes no input parameters. Drawbacks of Alvares’s approach are: (1) distortion of the GPS signal, for example due to signal multipath, can easily lead to false positives, (2) absence of the GPS signal during indoor activities is not taken into account, and (3) the accuracy indic-
ator of GPS signals is ignored, causing imprecise signals to be interpreted with equal importance as precise ones.

Figure 5.3: Existing POI extraction approaches: circles represent trajectory points, filled circles represent trajectory points with significance in the illustrated approach

5.2.2 Palma’s CB-SMoT

Palma’s Clustering-Based Stops and Moves of Trajectories (CB-SMoT) [71], illustrated in Figure 5.3b, is based on a variation of the well-known clustering algorithm DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [29]. Rather than setting the clustering parameter \( \text{Eps} \) (used to determine the maximum allowed distance between points before starting a new cluster) for all trajectories to the same value, they introduce a quantile function that takes an input parameter \text{area}, and uses this to automatically determine a proper setting for \( \text{Eps} \) per trajectory. In this way, trajectories with faster movement have distance threshold values that are higher than the trajectories of slower moving ones. Palma et al.’s approach also requires the input parameter \text{minTime}, a minimum time threshold that is used as an alternative for DBSCAN’s minimum number of points to create a cluster. The clustered trajectory parts (illustrated by filled trajectory points) are then intersected with the known POI polygons. Because of the clustering approach, a reflecting signal while moving past a POI is generally discarded. However, the second and third drawback of Alvares’s approach also apply here: due to the absence of GPS signals inside buildings, the clusters are often located outside the POIs parcel polygon, and cluster building is disturbed by imprecise signals.

5.2.3 Rocha’s DB-SMoT

Rocha’s Direction-Based Stops and Moves of Trajectories (DB-SMoT) [76], illustrated in Figure 5.3c, is based on the notion of heading change: if the direction of a trajectory frequently changes, this indicates a stop (illustrated by filled trajectory points). Besides a trajectory, their algorithm takes three
parameters as input: the minimum direction change \(\text{minDC}\), the minimum time interval to build a new cluster \(\text{minTime}\), and the maximum tolerated consecutive number of trajectory samples that do not exceed \(\text{minDC}\) inside a cluster: \(\text{maxTol}\). Rocha et al. validated their method using the fishing locations of Brazilian vessels, for which the POI areas are relatively large compared to the inaccuracy of the GPS signal. In the application for indoor urban activity detection, the location accuracy is much lower, as the receiver often cannot pick the optimal satellite constellation. Furthermore, just like in the other two approaches, knowledge of the signal’s accuracy is not taken into account, causing inaccurate points to be regarded relevant for this geometric analysis.

### 5.2.4 Other related work

Besides POI visit extraction, other types of trajectory annotation have received their share of attention. Stefanakis showed for example, how trajectories can be annotated with geometric properties \[82\]. These properties can then be used for trajectory simplification, assisting us to select the relevant sampling points from the trajectories. An example of a useful trajectory simplification metric is the one from Chen et al. \[20\], which we discuss in the next section as a part of our approach. Guc et al. also annotate trajectories with POI visits \[39\], but this annotation is done manually. SeMiTri of Yan et al. annotates trajectories automatically, but since they find it probabilistically intractable to infer the exact POI, they annotate the trajectories with properties of the region where the trajectory was created \[91\]. Another algorithm that could be used for POI visit detection is Continuous Nearest Neighbor (CNN) Search by Tao et al. \[85\]. However, since by definition, this approach continuously picks a POI, this approach leads to many false positives, albeit for a very short time interval. This approach would therefore need to be extended with a filter of those POIs that should be marked as visited, and those that were simply nearby for a shorter amount of time. However, in this chapter we prefer to focus on the comparison with existing algorithms, rather than adaptations of existing algorithms, and therefore we consider it outside the scope of this chapter. Kang et al. \[52\], to conclude, came up with a time-based clustering approach based on WiFi positioning purely for POI visit detection. As stated in the introduction however, we focus on POI visit detection from smartphone trajectory data, which may come from multiple sources.

### 5.3 Approach

Our PIE algorithm is mainly based on four parameters from the smartphone location sensor readings and its spatio-temporal derivatives: the accuracy
5.3. Approach

of the location samples as provided by the smartphone OS, reductions in speed, changes in direction and projection of signals onto parcel polygons. We begin this section by defining several geometric concepts that we use as selection filters, and then present the algorithm that combines these.

5.3.1 Definitions

Trajectory and trajectory sample point

The location signal in a mobile device is only accurate up to several meters (see for example [94]), depending on several factors, such as the vicinity of known WiFi networks, the quality of the GPS sensor, the number of detected satellites by the GPS sensor, their spatial arrangement in the sky, and the presence of reflecting surfaces. This is modeled in mobile devices using an accuracy parameter, measured in meters, which indicates the radius around the point in which the device may also be located. Even though the APIs of the large manufacturers have named this an accuracy parameter, we prefer to refer to this as an inaccuracy indicator, as its value increases with increasing inaccuracy.

Definition 5.3.1. A trajectory sample point is a spatio-temporal point, associated with an inaccuracy indication, represented by a tuple \( p = (x, y, t, i) \). We express \( x \), \( y \), and \( i \) in meters, and \( t \) in seconds. Other units suitable for distance and time calculation can be used as well, as long as the thresholds discussed below are set accordingly.

Definition 5.3.2. A trajectory is a chronological sequence of sample points:
\[ T = \langle p_0, \ldots, p_n \rangle. \]

Since trajectory samples with a high value for the inaccuracy indication are unsuitable for geometric calculations, we also introduce the inaccuracy threshold value \( i_{\text{max}} \).

Staypoint

Li et al. introduced the concept of a staypoint [57]; it is illustrated in Figure 5.4. Zheng elaborated on this in [97], from which we follow the definition: a geographical region where a user stayed over a time threshold \( T_r \) within a distance threshold of \( D_r \). However, the definitions and algorithms of Li and Zheng do not take maximality into account. This causes multiple staypoints are created, based on sequences that contained each other, as for example is the case for \( p_4 \) and \( p_5 \) in Figure 5.4. To avoid this, and let a trajectory sample, we follow a slightly different definition.
Definition 5.3.3. A candidate stop sequence $C$ is a non-empty subsequence of a trajectory $T$ for which $p = \langle p_m, \ldots, p_n \rangle$, for which $\forall m < i \leq n$, $\text{Dist}(p_m, p_i) \leq D_r$, $\text{Dist}(p_m, p_{n+1}) > D_r$ or $n + 1 \geq |T|$, and $\text{Int}(p_m, p_n) \geq T_r$, where $\text{Dist}$ is the geospatial distance between two sample points and $\text{Int}$ is the time difference between two sample points, $|p_i.t - p_j.t|$.

Definition 5.3.4. A candidate stop sequence $C$ of a trajectory $T$ is a stop sequence $S$ if and only if no other candidate stop sequence $C'$ exists for $T$ that contains all samples contained in $C$.

Definition 5.3.5. A stay point $s$ is defined as the centroid of a stop sequence $S$. $s$ is a tuple $(x, y, t_a, t_d, i)$ where $x$ and $y$ are the coordinates, $t_a$ is the arrival time, $t_d$ is the time of departure, and $i$ is the maximum inaccuracy:

- $(x, y)$ is the centroid of $S$,
- $t_a = \min_{p \in S}(p_i.t)$,
- $t_d = \max_{p \in S}(p_i.t)$,
- $i = \max_{p \in S}(p_i.i)$.

Figure 5.4: A trajectory, where three trajectory points $\langle p_3, p_4, p_5 \rangle$ form a staypoint $s$ (after [97]).

Heading change

Chen et al. introduced several forms of heading change for their trajectory simplification method proposed in [20]. We adopt their concept of neighbor heading change, which corresponds with the definition of direction change used by Rocha in [76]. The heading change is the change in heading that takes place at a point $p_i$ with respect to its direct neighbor points in the trajectory:

Definition 5.3.6. The heading change $\theta$ for a point $n$ in the trajectory $T$ is given by:

$$\theta_i = \frac{180}{\pi} \cos^{-1} \left( \frac{a \cdot b}{||a|| ||b||} \right) \degree,$$
5.3. Approach

where

\[ a = \begin{bmatrix} p_i.x - p_{i-1}.x \\ p_i.y - p_{i-1}.y \end{bmatrix}, \]

\[ b = \begin{bmatrix} p_{i+1}.x - p_i.x \\ p_{i+1}.y - p_i.y \end{bmatrix}. \]

For the first or last point in a trajectory, the heading change is defined as \( \theta = 0^\circ \).

Where the algorithm of Rocha et al. looks for repeatedly changing directions, we are only interested in the direction change at the selected point. Small direction changes in a trajectory normally indicate a person travelling, while arriving at their destination, the movement deviates from this to enter their destination. Of course, people may have many other reasons to change direction, such as following a road network. Therefore, filtering based on direction changes alone does not suffice, but it is merely one of the features in our approach to filter out points. To extract those points where people change direction, we use a direction change threshold \( \theta_{\text{min}} \), where \( 0 \leq \theta_{\text{min}} \leq 180^\circ \). Note that a simplified trajectory is still a trajectory, and this definition therefore allows us to filter out sample points prior to applying this threshold to filter out points.

Parcel polygons and polygons-of-interest

As defined in Chapter 1, a POLOI is a location where goods and services are provided, geometrically described using a polygon, and semantically enriched with at least an interest category. A parcel polygon is similar to a POLOI, but does not necessarily describe a location where goods and services are provided, but may also be another type of parcel, such as a residence. In Chapter 4, we discussed how a POLOI can be estimated given a POI set using open data. In this chapter, we assume the availability of polygons for all nearby parcels as the polygon set \( P \). Those elements in \( P \) that also represent a POLOI are contained in a subset of \( P \), referenced as \( P_{\text{POLOI}} \). The result of the PIE algorithm, which is the set of visited POLOIs, is denoted as \( P_{\text{VP}} \).

Polygon projection and maximum projection distance

GPS signals are typically unavailable inside buildings. Therefore, we use selected points, derived from the GPS signal right before or right after a stop, to determine which place has been visited. These points are projected onto the nearest parcel polygon. We call this polygon projection. To limit the projection distance, we introduce the absolute maximum projection distance \( \pi_{\text{max}} \) within which the point is still considered indicative of a parcel visit.
5.3.2 Algorithm

In our PIE algorithm, we combine the concepts discussed above. First, we filter out those points for which the inaccuracy value exceeds the accuracy threshold $i_{max}$. These points are too imprecise to do further calculation with. Secondly, we extract the staypoints from a trajectory. Then, we determine the direction change between the staypoints to determine whether any staypoint is an indication for a visited location or should be attributed to natural behavior in traffic. Thereafter, we project the selected points onto the nearby polygons, taking the maximum projection distance $\pi_{max}$ into account. Those polygons that are POLOIs, are added to the result set of visited polygons that are not POLOIs. Note that, unlike in the work of Palma and Chen, the heading change is taken into account after staypoint detection: the advantage is that this filters out those staypoints that were on a relatively straight path with respect to the previous and next staypoints. This leads to the formalization of our algorithm shown in Algorithm 1.

5.4 Validation

We begin this section with a description of the validation approach, followed by the way we collected and cleaned the ground truth data. Then, we discuss the metrics used to evaluate how our algorithm performs in comparison to existing algorithms. Finally, we present the results of the validation with a short discussion.

5.4.1 Approach

To validate our approach, we rebuilt the approaches of Alvares, Palma and Rocha, based on their papers. Just like our approach, the approaches by Palma and Rocha require several parameters to be set. For these three approaches, we used a ten-fold cross validation to detect the proper parameter settings using a brute force approach. To limit the number of combinations, we set the parameters $i_{max} = 10\,\text{m}$, $\pi_{max} = 10\,\text{m}$, and $\theta_{min} = 30^\circ$ for the PIE algorithm, while detecting $T_r$ and $D_r$ automatically.

5.4.2 Data collection

To collect validation data, we set up an experiment similar to the one we described in [26]. During the welcome week of the University of Twente, called the Kick-In, the new students were invited to participate in a treasure hunt-like game that lasted four hours. The students were supposed to carry out exercises, and upload a picture of this activity with the specifically designed mobile application Kick-In Discover Hengelo, illustrated by the
Algorithm 1 Point-of-Interest Extraction (PIE)

**input:** 
- $T$ // trajectory as sequence of sample points
- $i_{max}$ // maximum inaccuracy in meters
- $D_r$ // staypoint distance threshold in meters
- $T_r$ // staypoint time threshold in seconds
- $\theta_{min}$ // minimum direction change in degrees
- $\pi_{max}$ // maximum projection distance in meters
- $P$ // set of all existing polygons
- $P_{POLOI}$ // subset of $P$, containing POLOIs

**output:** 
- $P_{VP}$ // subset of $P_{POLOI}$, containing visited POLOIs

1: procedure PIE

2: // Select points based on inaccuracy
3: accuratePoints $\leftarrow \{ p \in T \mid p.i \leq i_{max}\}$;

4: // Select points based on staypoints
5: stayPts = stayPointDetection(accuratePoints, $T_r$, $D_r$);

6: // Select points based on heading change
7: selectedPoints $\leftarrow \{ p \in stayPts \mid headingChange(stayPts, p) \geq \theta_{min}\}$;

8: // Select nearest polygon, apply maximum projection,
9: // and check if it is a POLOI
10: $P_{VP}$ $\leftarrow \{ p \in selectedPoints \mid$
11: polygon = getNearestPolygon($P$, $p$)
12: $\land dist(polygon, p) \leq \pi_{max}$
13: $\land polygon \in P_{POLOI}\}$;

14: return $P_{VP}$
15: end procedure

screenshot in Figure 5.5. The exercises could be carried out in any given order, and the best picture was rewarded with a prize.

The application, which was built on the PhoneGap platform [3], was available on both Android phones and iPhones. Ten of the 24 exercises had to be carried out at a specific POI. The employees of these POIs prepared a set-up for the respective exercise, such as a poker table, or a karaoke set. The organization of the event, which we had collaborated with while designing the app, had the possibility to send messages to all users at once. They used this to motivate the students to move on to the next exercise roughly every half hour. In the background of the application, the trajectory was sampled with explicit prior user consent. The pictures taken by the students, were linked to their trajectory and formed a proof of their visit to that specific
5. Automated semantic trajectory annotation

POI. We used this proof as a manual ground truth POI visit annotation of the trajectories.

This experiment, which was carried out in 2014, was similar to the one that we carried out in 2013, which is described in Chapter 6. However, as this was the second iteration, we applied several improvements to the set-up. First of all, the students had to upload pictures to prove their presence at the exercise locations. This way, we could objectively remove wrongfully added annotations. Secondly, in 2014, the students were allowed to perform the exercises in any order. This way, the students did not feel obligated to perform an exercise, simply to see the next one. Thirdly, the locations were prepared to receive the students, which made the event more inviting to actively participate. Lastly, the event took place in the evening, rather than in the late morning, which is typically a more popular time slot of the Kick-In events.

An unforeseen, yet important side-effect of this data collection, is that the resulting annotation is not complete: students did not upload a picture from their own device at every location they went to, but at several occasions, one picture was uploaded with several people on it. We address this when discussing the validation metrics in Section 5.4.4.

(a) For each exercise the students were invited to upload a picture

(b) A map helped the students to move between participating POIs

Figure 5.5: Screenshots from the Discover Hengelo app

For the polygon sets \( P \) (and thus \( P_{\text{POLOI}} \)), we used authoritative data
from a public web feature service, offered by the Dutch government through the open data initiative Nationaal Georegister [67] (the same as the one we used for the regeocoding of addresses in Chapter 4). To thoroughly test the algorithms for false positives, we assumed all parcel polygons in the vicinity of the event to be POLOIs, and thus in this case: \( P = P_{\text{POLOI}} \).

5.4.3 Data cleaning

Several pictures were uploaded outside the time window of the event, due to people playing around with the app before and after the event. Furthermore, several pictures did not meet the exercise requirements. For example, exercises were carried out while not being at the correct location, or the picture was entirely black. These pictures were discarded as annotations after manual inspection. Trajectory samples that were not within the time window of 2 minutes before the user’s first annotation and 2 minutes after his last annotation were discarded as well. Trajectories that did not contain at least 50 samples with an inaccuracy value below 30 were also excluded from the validation data, as these students most likely did not have GPS positioning turned on. This resulted in 23 valid trajectories from a wide range of device types, containing a total of 30,500 trajectory samples and 128 annotations.

5.4.4 Validation metrics

Our validation metrics are based on those that are commonly accepted in information retrieval: precision, recall, and their harmonic mean F measure [63]. These metrics are indications for the relationship between the numbers of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). Since the annotations were not entirely complete (as discussed above), we introduce a new category: probable positives (PP). These are the locations where students may have been, since these POIs participated in the event, but for which we do not have proof in the form of a (valid) photo. We extend the notion of precision, the fraction of retrieved POIs that are relevant, accordingly by interpreting PPs as TPs:

\[
\text{OptimisticPrecision} = \frac{TP + PP}{TP + PP + FP}
\]

We also use the more conservative pessimistic precision, in which we consider PPs as FPs:

\[
\text{PessimisticPrecision} = \frac{TP}{TP + PP + FP}
\]
For the fraction of relevant POIs that are retrieved, or \textit{recall}, we use the regular formula:

\[
\text{Recall} = \frac{TP}{TP + FN}
\]

For our validation, we consider precision and recall equally important, in which case the optimistic and pessimistic F measures are formally given as:

\[
OF = \frac{2 \times \text{OptimisticPrecision} \times \text{Recall}}{\text{OptimisticPrecision} + \text{Recall}}
\]

and

\[
PF = \frac{2 \times \text{PessimisticPrecision} \times \text{Recall}}{\text{PessimisticPrecision} + \text{Recall}}
\]

\subsection*{5.4.5 Results}

In Figure 5.6, we illustrate POI visits as extracted by the different approaches for the same trajectory. A green polygon indicates a true positive for POI visit extraction, a blue polygon a probable positive, a red polygon a false positive, a yellow polygon a false negative, and grey polygon a true negative. White polygons indicate locations that participated in the event, but were not visited by this student.

The straightforward intersection-based approach by Alvares in Figure 5.6b leads to many false positives, due to the inclusion of location samples where signal multipath was in play. Similarly, Palma’s clustering-based approach in Figure 5.6c suffers from this: detected clusters are often located in front of the POI, rather than intersecting with the polygon. Without a projection on the nearest polygon, this leads to many false negatives. The direction-based approach of Rocha in Figure 5.6d suffers from the constant reflection of signals in urban areas: many clusters are detected due to inaccuracies of the signal, rather than actual back-and-forth movement. As a result, clusters are created that were caused by signal multipath.

In Figure 5.7, we show the results of our validation aggregated over all valid trajectories. PIE outperforms the three existing approaches in pessimistic precision, optimistic precision, and recall, and as a result in F measure as well. We are able to extract visited POIs with a precision of 57.9\%, classical precision of 44.7\%, and recall 68.0\% of the visited POIs. This corresponds with an optimistic F measure of 0.625, substantially higher than the optimistic F measures for Alvares (0.441), Palma (0.371), and Rocha (0.443). The pessimistic F measure for PIE equals 0.540, also substantially higher than those for Alvares (0.335), Palma (0.331), and Rocha (0.344).

To give an indication of appropriate parameter settings for the PIE approach: nine validation batches from the 10-fold cross validation used...
5.4. Validation

(a) PIE: clusters outside POLOIs still lead to POI/POLOI visit detection

(b) Alvares: many false positives due to simple intersection

(c) Palma: missed polygons due to location of clusters outside POLOIs

(d) Rocha: missed polygons due to location of clusters outside POLOIs; going around a corner also causes cluster creation

Figure 5.6: POI extraction from trajectory #110 using different algorithms.
a distance threshold $3m \leq D_r \leq 5m$ and a time threshold $11s \leq T_r \leq 14s$. 

We also experimented with a relative value for the $\pi_{max}$ parameter, that relates the projection distance to the distance to the second nearest parcel, but this led to significantly inferior results. Another idea we experimented with was to take signal loss into account, but since this can be caused by several factors, this turned out to be a rather weak indicator for parcel or POI visits.

### 5.5 Privacy considerations

With great power, comes great responsibility. Algorithms like ours can be used to the benefit of a user, just as well as to his discomfort. Users are often careless about the permissions that a mobile application requires. We consider it the application developer’s moral obligation to reduce the collection of private data to a minimum. Therefore, we share this algorithm with the following three privacy considerations:
**Information ownership**

Trajectory data is personal data that belongs to a user. Therefore, which POI a person visited is information that shall be available to only that person, unless he agrees to share this information. Extracted POI visits shall remain on the device of the user. Only an aggregated profile, based on numerous POI visits, shall be sent to a service that uses these. For example: the fact that a user has children, can be beneficial to offer the right services at the right time. However, where his children go to school, is too personal to share.

**Informed user consent**

For certain services, it is required to know the exact POI where a person is currently located. One can think of a tag for social media, or, more importantly, an emergency call application for an elderly person. In this case, we consider it reasonable to share the exact POI, but only with the consent of the informed user.

**POI set scope**

POI sets shall contain only those POIs that increase the quality of the service. The scope of object types contained in the POI set shall be carefully selected for each application, and shall certainly not contain location types that a person may not be willing to share in public.

### 5.6 Conclusion

In our GRS, knowledge on POI visits derived from trajectories using the algorithm introduced in this chapter, can be used to populate user profiles, or to determine popular locations in a specific region to fill geoprofiles. The strength of the algorithm has a direct relationship with the accuracy of the profiles. In this chapter, we introduced and validated our POI visit extraction algorithm PIE. Using a combination of several geometric and spatiotemporal processing steps, we are able to infer the visited POIs with significantly better results than those of existing approaches, as this algorithm is specifically designed for urban indoor trajectory analysis. Even for those trajectories for which typical challenges for this type of trajectory analysis, such as signal loss and inaccuracy, play a role, PIE manages to retain a combination of high precision and high recall. It is important to note though, that the validation was carried out with high quality polygon data.
Spatiotemporal profiling for UGC quality assessment

Abstract

Knowing where people have been, as described in the previous chapter, provides us with information that can be used to assess the quality and relevance of user-generated content. Someone who went to a restaurant, and writes a review on it, is more likely to write a fair review, than someone who works at a competitor. Similarly, a mother of a young child is more likely to write a useful review on the child friendliness of a holiday home, than a young adult without children. In this chapter, we propose a method for spatiotemporal user profiling. In four phases, we process trajectory data to predict the quality of the associated UGC: (1) data collection, (2) data cleaning, (3) trajectory analysis, and (4) user classification and content quality prediction. Our method provides a step-by-step way for human interpretation to detect patterns in the trajectory data.

We use a treasure hunt for students in the city of Enschede, The Netherlands, as a running example of how these methods can be used, and to evaluate the proposed profiling method. The profiling method is used to predict the degree to which the students adhere to the prescribed rules by means of a simple team type classification, and a feature-based answer type classification.

6.1 Introduction

Trajectories are more than a simple collection of time-stamped geographical coordinates. They represent user actions and can, when interpreted properly, lead to an in-depth analysis of behavior and, consequently, user profiling. As a matter of fact, you are what you “where”!

Research involving trajectories has mostly focussed on data management and data mining aspects at the geometrical levels. Results have
led, for example, to improved telematics services using live traffic assessment by means of vehicle tracking and, more recently, map construction algorithms resulting into automatic road network generation and updates (e.g. [11]). In this chapter, we focus on how we can extract user behavior from trajectories, to use this information for UGC quality and relevance assessment. Trajectory pattern mining can lead to interesting insights, as was also acknowledged by Jeung et al. [49], and Zheng et al. [99]. Recently, two in-depth overviews of the state of the art in the field of modeling and semantic enhancement of trajectory data were presented by Parent et al. [73] and Jiang et al. [50]. Behavior, as we discuss it in this chapter, is defined as semantic behavior in [73]: “trajectory behavior whose predicate bears on some contextual data and possibly on some spatial and/or temporal data.” Instead of trajectory behavior, we prefer the wording trajectory characteristics, which we define as notable changes in a trajectory in the spatial or spatiotemporal domain, such as changes in speed or direction.

In this chapter, we introduce a method to classify users based on their movement, with the goal to predict the correctness of their UGC. Our overall method consists of four phases: (1) data collection, (2) data cleaning, (3) trajectory analysis, and (4) user classification and content quality prediction. The first two are addressed in Section 6.3, the third phase in Section 6.4, and the fourth phase in Section 6.5. In the third phase, the trajectory data is analyzed using a timeline and a staypoint analysis, as well as a semantic trajectory inspection relating actual and expected paths. Timeline analysis detects the differences between users and between their changing behavior over time. Staypoint analysis determines where teams spent a significant amount of time and helps in assessing the impact of the environment on user behavior. Trajectory inspection is then used to distinguish between engaged teams and indifferent teams.

The outcome of the analysis phase is used as input to the profiling and prediction phase. This chapter proposes two UGC quality prediction methods: a simple classification approach, based on a small number of features, and a machine-learning based approach, that uses more features, and uses a decision tree to predict UGC quality. In our running example, as well as the validation, we show how a team type classification is created based on two distinguishing, spatiotemporal features. Answer type classification is based on a total of 10 features overall describing the teams and their answers. The generated profiles and their features are used as input to the automated generation of a decision tree to predict answer types, both with respect to correctness of the answer, and adherence to the treasure hunt rules. Being able to predict the likelihood of correctness, is especially useful in scenarios where answer correctness is not as sharply defined, such as user-generated reviews of products or places.

To validate our method, we use a treasure hunt contest as a case study. During the welcome week at the University of Twente, The Netherlands,
6.2 Related work

Work on behavior profiling based on GPS data often describes the analysis and prediction of travel patterns. Liao et al. use a hierarchical Markov model that is trained with historical data from that user to predict new locations and recognize deviating travelling patterns [59]. Monreale et al. use a decision tree, named T-pattern Tree, that requires manually annotated training data [66]. Giannotti et al. [34] who use visited regions of interest (such as a railway station, bridge, or a museum) and sequential patterns thereof to describe human movements. Zheng et al. [96, 98] used a similar, but more formalized approach, using staypoints, and a tree-based hierarchy of clustered stay points. Spaccapietra et al. [81] introduced a conceptual model using movement types and visits to specific points of interest to describe the behavior of people in sequential patterns including the movement. Yan et al. [90] introduce a framework for semantic annotation of trajectories using several abstraction layers, and discuss typical challenges when dealing with trajectory data from mobile devices.

In this chapter however, we focus on the patterns in trajectory data that tell us something about the quality of UGC by that user (group). The method we introduce contains a timeline analysis that resembles the approach taken by Guc et al. [39], but contrary to their manual annotation approach, the annotation in our work is done automatically. For another in the trajectory analysis phase, we use the staypoint detection by discussed prior in Section 5.3.1. As in this chapter we provide a method for pattern detection that relies on human cognitive skills, we use it here to create a visualization, rather than to support (other) algorithms.
6.3 Case study, data collection & data pre-processing

The trajectory data used in this work was collected as part of a case study that involved 54 teams comprised of first-year university students, who used a total of 132 mobile devices and participated in a treasure hunt as part of the student orientation. This event and the respective application, have similarities with the ones from Chapter 5, but the event discussed here took place on August 24th, 2013, roughly a year before the one from Chapter 5. The rules of the game were also changed significantly. In the following, we describe the 2013 event during which the data was collected, the technology that was used, the collected data, and the pre-processing of the data.

6.3.1 Event

Every year, new students are welcomed to the University of Enschede with a voluntary, but popular welcome week called the Kick-In. As part of the welcome week for new Bachelor students in 2013, the Kick-In Quest, as seen in Figure 6.1, took place on a Saturday morning, from 10:00 am until noon. The Kick-In Quest manifested itself as a treasure hunt, in which students had to answer questions at certain locations using a mobile app. Points were awarded for (1) answering questions correctly, and for (2) collected GPS location data. This was explained to the students using information screens in the app. The students worked together in teams, and were motivated to use multiple devices per team to obtain more location points (resulting in more collected tracking data). The awarded amount of points for correct questions was also based on the proximity to the question location, which forced students to move around the city center of Enschede, even if the answer was already known, or could be found online. Each of the 54 teams that participated had its own designated question sequence, guiding them to 20 locations. These sequences were put together from a total of 24 questions and respective locations. It was not possible to look ahead or change a previously entered answer in the app. In case the app was closed, the app continued where it had previously stopped.

6.3.2 Technology

The mobile application was developed using the PhoneGap platform [3]. This platform allows app developers to build an application as if it were a web page using HTML5, CSS, and JavaScript. The features of the mobile device, such as the GPS sensor, can be accessed through asynchronous JavaScript calls. PhoneGap supports many different platforms, but for this app only Android and iOS were used.
6.3 Case study, data collection & data pre-processing

6.3.3 Collected data

During the game, trajectory data was uploaded from the mobile devices to a server as tuples containing the trajectory id and all fields of PhoneGap’s Position object: latitude, longitude, altitude, accuracy, altitude accuracy, heading, speed, and timestamp. Furthermore, answer data was uploaded containing the following information: question ID, trajectory ID, answer, latitude, longitude, accuracy, and operating system (i.e., Android, or iOS).

6.3.4 Data pre-processing

Besides several minor data cleaning tasks, such as resolving differences between the date formats of the different platforms, data pre-processing involved the removal of location references based on cell phone tower locations and trajectory point outliers. Android offers two types of locations: fine-grain locations and coarse-grain locations. PhoneGap uses the coarse location data when the fine-grain location is unavailable. This results in trajectories as illustrated in Figure 6.2. Coarse-grain locations can be detected by setting a threshold on the accuracy indicator, with which each location sample is annotated by the APIs of the GPS sensors. Removing all data points for which the accuracy value exceeds 50m has filters out all such coarse points, as can be detected through manual inspection by observing the existence of repeating outliers at the exact same location as in Figure 6.2. In addition, by using a realistic threshold for the speed of the participants...
(30 m/s in the urban area where the event took place), extreme outliers were removed from the GPS trajectories as well.

Figure 6.2: Trajectory before removal of location references based on cell phone tower locations. The signal keeps jumping back and forth between relatively accurate GPS positioning and less accurate cell phone tower locations.

6.4 Spatiotemporal behavior analysis

Now that we have collected and cleaned the data, we have reached the third phase of our prediction method: trajectory data analysis. In this phase, we use three steps to understand the data and the respective patterns in the participants’ behavior. First, we perform a timeline analysis to observe the differences between (groups of) users and changing behavior over time. Secondly, we use a staypoint analysis to determine where teams spent a significant amount of time and to get insight on the impact of the respective environment on the participants’ movements. Lastly, we inspect the actual
6.4. Spatiotemporal behavior analysis

Table 6.1: Spatial Question/Answer categories.

<table>
<thead>
<tr>
<th></th>
<th>Close</th>
<th>Distant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>Correct Close (CC)</td>
<td>Correct Distant (CD)</td>
</tr>
<tr>
<td>Incorrect</td>
<td>Incorrect Close (IC)</td>
<td>Incorrect Distant (ID)</td>
</tr>
</tbody>
</table>

trajectories with respect to the expected trajectories using the question locations.

6.4.1 Timeline Analysis

The spatiotemporal question-answer behavior results in a specific profile for each group. Our hypothesis is that by quantifying this behavior, we can easily identify more or less successful participants. A timeline analysis is used to relate the behavior of (groups of) users by means of analyzing the temporal answering behavior, i.e. how much time did they spend on answer types, and how did their answering behavior evolve over time.

The questions in the treasure hunt were location-bound. This means that the students were instructed to answer the question at a stated location, even if the answer was known or found online. The students were motivated to visit the location by awarding an increasing amount of points based on the inverted distance to that location at the moment of answering the question. Therefore, to score the maximum number of points, the answers needed to be (1) correct, and (2) answered at the respective location. The students were informed of these rules at the start of the game, and were motivated to score points by the intermediate score indication at the top of the screen. Combining right and wrong answers with close or distant answer locations resulted in the four different answer types shown in Table 6.1.

In Figure 6.3, we provide the timeline analysis graph for our case study. The $x$-axis represents the time, the $y$-axis represents the team number. The teams are numbered by their first answering time. The bar of each team consists of a sequence of answers and the time it took to answer each. The longer each bar, the longer it took to answer. The four colors represent the four answer types. CC answers are illustrated in green, CD in blue, IC in yellow, and ID in red. For each answer, the start of the bar represents the moment at which the question is presented on the screen, and the end of the bar the moment the answer is entered in the application (with a small degree of freedom for visualization purposes). For teams that used multiple devices, only the first given answer is considered, because the mobile application often hints at the correct answer after an answer is provided. This influenced the answering behavior on the other devices.

The green (CC) answers, which are correct in both actual answer and location, are more prominent at the beginning of the game. CC answers
6. Spatiotemporal profiling for UGC quality assessment

Figure 6.3: Timelines of teams. Colors indicate answer types. Highlighted teams are discussed in detail below. The length of the bars indicates the time taken for each answer.

take a substantial amount of time to obtain, since the students have to move to the right location. Therefore, a large part of the graph is green. This makes this visualization type suitable to get insight into the amount of time spent on each answer type. If additionally it is of interest to see the ratio of behavior occurrences, other techniques can be used, such as the pie chart of answer types in Figure 6.4. A closer look at Team 33, for example, shows a very green timeline, while only 9 out of the 20 answers are actually CC answers. The team with the highest percentage of CC answers was Team 16, with 12 out of the 13 answers correct and at the right location. Not only did this team have the highest percentage, the score of 12 was a tie for the highest number of CC answers. A deeper analysis of the data for this specific team gave us insight in their behavior. The reason that this team could achieve this high number of CC answers without skipping questions, by providing a CD or ID answer in between, was that it participated with four devices, and split up into a walking group with two devices, and a cycling group with two more devices. Judging by the perfect timing of
which devices were used to answer the question first, we can conclude that they must have communicated about this throughout the game.

![Answer type frequencies](image)

Figure 6.4: Answer type frequencies. Only the first answer for each team has been taken into account, in case a team participated with multiple devices, since the explanation in the next screen often revealed the correct answer.

From this timeline, we can detect several team behavior types. Some teams were indifferent about the outcome of the entire game, which is reflected by a late starting time, short answering times and many incorrect answers (e.g., Teams 51 and 52 in Figure 6.3). Other teams were indifferent about answering the questions at the right location, while still trying to answer questions correctly (e.g., Team 2). Teams like Team 1 just wanted to finish the game. Finally, teams like Team 16 were dedicated throughout the entire game. From those teams that played the game all the way until the end, there is another behavior pattern that can be detected. Towards the end of the game, these teams became rushed. This can be seen by the many CD and ID answers towards the end (e.g., Team 33). Another important observation is that there were barely any teams that were consistent in their answer types, except for team 16 that has a nearly entirely green timeline. A deeper analysis of this observation was done using the bar chart in Figure 6.5. This chart, combined with the knowledge of question locations from Figure 6.6, shows that remotely located questions were less popular to be answered near the indicated location.

In addition to the timeline analysis, different visualizations are useful to compensate for the coloring bias that is induced by the fact that some behavior requires more time than other behavior. In the treasure hunt case for example, CC answers require more traveling time than CD answers. This results in an overall “greener” graph, while the pie chart in Figure 6.4 shows that CD answers (blue) are actually more common.

### 6.4.2 Staypoint Analysis

Staypoint analysis is another form of spatiotemporal analysis. It provides insight into the most significant locations of the people’s stop-and-go behavior. Following the definition from Section 5.3.1 staypoints are points
where the speed was below a certain threshold for a respective time period. This analysis is uncoupled from that what the students were asked to do (answering questions), and gives us more insight into what they actually did besides playing the game.

In our case study, we use a speed threshold of $1\text{m/s}$ and a stay threshold of 30s. To overcome the problem of GPS signals bouncing around when no, or little movement is observed, we did not use the typical sampling rate of 1Hz, but compared the position with a GPS signal sample of 5s ago (under-sampling).

The results of this analysis are shown in Figure 6.6. The red circles are the actual question locations. Blue circles are used to visualize the staypoints after clustering them on a trajectory basis. Multiple blue circles located close to each other therefore represent multiple trajectories slowing down significantly in that area. For visualization purposes, the radius size of the question location circles was reduced to 25m, as opposed to the 100m used to distinguish between “close” vs. “distant” answers in the timeline.
Figure 6.6: Staypoints derived purely from GPS data. Blue circles indicate detected staypoints, the red circles and numbers indicate the question locations. The park in the northwest corner is the location of the event that took place afterwards. Question 4 (in the southeast) was in a supermarket.
Several groups of staypoints can be detected. The cluster of points in the upper left corner of the map reveals the location of the next program element for the students. Several teams went to this location early, or left the app running while the game was already over. The game was initiated near the center of the map. This is where the students were supposed to gather, and obtained a team code to start the app. Question no. 4 was to be answered inside a supermarket. The cluster of staypoints in the lower right corner of the map surrounds that location. Judging by the time spent near this supermarket, several students went into the supermarket for more than just the answer of the question. The slow movement afterwards is probably caused by the consumption of snacks and drinks afterwards.

### 6.4.3 Trajectory inspection

To gain further insight into the movement patterns of teams, we carried out a visual inspection of trajectories. This revealed some challenges as several teams had turned off their GPS tracking between answering questions, most likely to save battery power. Also, we could cluster the trajectories into four types, (1) barely moving, (2) not leaving the center of the town, (3) leaving center of town once, and (iv) moving around.

Figure 6.7 shows examples of different trajectory types. However, no clear correlation could be found between the correctness of answers and the movement patterns of the teams. For example, Team 25 answered 10 questions, all correctly, without moving substantially.

### 6.5 Behavior prediction

In many scenarios of UGC quality analysis, the correctness of content is unknown. An example scenario is the analysis of reviews and ratings of a restaurant. A positive review may be a realistic description of the user’s experience, or it may be a bogus review written by the restaurant owner. The ability to distinguish serious users from the less serious ones, allows us to predict the quality of their content. In the case study, we have this information on answer correctness and used this as a ground truth to validate two prediction mechanisms built using our approach. The first one, a simple team type classification, is mainly based on the outcome of the timeline analysis. For the second one, a machine-learning approach to predict the answer types, we have used the outcome of all three analyses.
6.5. Behavior prediction

(a) Type 1 - Team 51 barely participated and answered the questions from a cafe near the start of the event.

(b) Type 2 - Team 1 participated, but did not leave the center.

(c) Type 3 - Team 14 actively participated and moved somewhat around to answer questions.

(d) Type 4 - Team 16 actively participated, and visited even remote question locations.

Figure 6.7: Raw trajectories of several teams. Each device (per team) has an own color for the trajectory. The red numbers indicate the question locations for that specific team. Teams were unable to look ahead at the next questions.
6.5.1 Team type classification

A simple team type classification categorizes teams into four groups using (i) start time and (ii) average speed of each team. Table 6.2 shows the resulting four types. Teams that started within 2000s of the first starting team were classified as early starters. The cut-off speed between the *slow* and *fast teams* was set at the median speed of all teams, being 1.2859 m/s. For teams that participated with multiple devices, the entire travelled distance was divided by the total amount of time the devices uploaded a GPS signal to the server. Teams that answered less than 10 out of the 20 questions were not taken into account for this classification. The resulting classification for all the teams can be found in Figure 6.8.

Table 6.2: Team characteristics

<table>
<thead>
<tr>
<th>High average speed</th>
<th>Early start time</th>
<th>Late start time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serious</td>
<td>18</td>
<td>Rushed</td>
</tr>
<tr>
<td>Get-It-Over-With</td>
<td>32</td>
<td>Indifferent</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Low average speed</th>
<th>Starting time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get−It−Over−With</td>
<td>0-500</td>
</tr>
<tr>
<td>Serious</td>
<td>500-1000</td>
</tr>
<tr>
<td>Serious</td>
<td>1000-1500</td>
</tr>
<tr>
<td>Serious</td>
<td>1500-2000</td>
</tr>
<tr>
<td>Serious</td>
<td>2000-2500</td>
</tr>
<tr>
<td>Serious</td>
<td>2500-3000</td>
</tr>
<tr>
<td>Serious</td>
<td>3000-3500</td>
</tr>
<tr>
<td>Serious</td>
<td>3500-4000</td>
</tr>
<tr>
<td>Serious</td>
<td>4000-4500</td>
</tr>
<tr>
<td>Serious</td>
<td>4500-5000</td>
</tr>
<tr>
<td>Serious</td>
<td>5000-5500</td>
</tr>
</tbody>
</table>

Figure 6.8: Team types using the 2-dimensional team type classification based on average speed and starting time. Numbers indicate the team IDs.
6.5. Behavior prediction

The radar charts in Figure 6.9 show the distribution of answer types for each team type. Each of the four axes represents the fraction of answers for that answer type per team.

Again, it can be seen that IC (incorrect + close) answers are uncommon (none of the teams had more than 20% IC answers), and this makes them hard to predict. CC answers (correct + close) are much easier to predict and are especially common among the teams classified as Serious. CD (correct + distant) answers are especially common among Get-It-Over-With teams. ID (incorrect + distant) answers can be found primarily among Indifferent teams, and to a lesser degree among Rushing teams.

(a) Serious teams have a strong tendency towards CC answers.

(b) Rushed teams typically have a high ratio of CD and ID answers, and very little CC answers.

(c) Get-it-over-with teams have a strong tendency towards CD answers.

(d) Indifferent teams have a strong tendency towards ID answers.

Figure 6.9: Radar charts of answer types for each team type. An axis displays the fraction of answers of that answer type.
6.5.2 Feature-based answer type classification

Rather than classifying team behavior, in the following, we examine answer type behavior, to validate the potential to extract information about content quality from trajectory data. We created nine features to describe the team behavior in general, and one feature to compare the answer to the answer of other teams:

Per team, we record the following features:
1. start time,
2. average speed of all the team’s trajectories (total distance divided by the total time),
3. maximum covered distance,
4. time spent using the application,
5. number of devices,
6. median distances travelled between answers,
7. median time elapsed between answers, and
8. number of staypoints
To relate the given answers to other teams, we also recorded:
9. whether the answer is the prevailing answer over all teams, at that point in time.

With respect to the last feature: in a UGC quality assessment system in another context, such as product reviews & ratings, incorporating the answers of others into the prediction model suffers from the cold-start problem. However, once several reviews are given, especially when including a scale-based rating (e.g. stars or a grade), it may be a strong indication for the content’s relevance and accuracy.

We use a decision tree-based brute force approach to find that combination of features that leads to the best $F$-measure (the harmonic mean of precision and recall as commonly used in Information Retrieval) for answer correctness prediction. The best combination of features to predict CC answers turned out to be that of (i) start time, (ii) duration, (iii) median distance between answers, and (iv) whether or not the answer is the prevalent answer over all teams.

To validate the results of this method, we carried out a 10-fold cross validation. Compared to a baseline method, namely assuming all students adhered to the rules, and thus all answers are CC answers, this prediction mechanism performs well in precision (0.5369 vs 0.3486), decent in recall (0.5321 vs the automatically perfect recall of 1.0000) and slightly better in $F$-measure (0.5274 vs 0.5169). Since the majority vote is a very strong feature for answer correctness prediction, the model is very strong for distinguishing correct and incorrect answers. However, when it comes to
correct answers, the model has some trouble distinguishing the CC answers from the more common CD answers. For incorrect answers, that are most often ID answers, the real-time model approaches perfect results, with an $F$-measure of 0.8686.

6.6 Conclusions

Trajectory data is much more than a temporal sequence of position samples. Combined with the right metadata, as in this case question/answer behavior in a treasure hunt, it represents rich spatiotemporal data that can be used to analyze and predict the user’s actions. In this chapter, we showed how user content can be predicted based on trajectory data using several types of analyses. Although these methods have been derived in the context of a treasure hunt scenario, they are generally applicable for the analysis of “rich” trajectory data. The staypoint analysis can be used to detect those locations where people spend more time. Examples are train stations locations in commuter data, taxi stops, traffic lights, and traffic bottlenecks. The timeline analysis is useful for comparing the behavior of people, and can lead to several types of categorizations, depending on the application. The timeline analysis can be used to detect patterns such as school visits, to distinguish parents from other users, as they are more likely to provide useful feedback on the child-friendliness of a campsite. Similarly, people who visit restaurants regularly, are more likely to give an accurate review on a restaurant. Such patterns help us to assess the relevance and accuracy of user input, for example given in a review & rating system.

What was not discussed in this chapter, but may also be a promising application of our method in the GRS context, is to use this type of pattern analysis not only for UGC quality assessment, but also for classifying users and detecting user interests for recommendation selection. For example, it may be useful to know if a user travels by train regularly, or owns a car, or to use the information that someone has a young child for the selection of their holiday home.
Knowledge-based recommendation selection

Abstract

The final step towards building a GRS is the selection of recommendations, based on the profiling techniques of the previous chapters. A requirement to provide personalized recommendations is that the recommender system has knowledge of the person using it. To limit the scope of this chapter, we focus here on the semi-structured user profiles that can be derived from social media accounts, by analyzing pages liked on Facebook, accounts followed on Twitter, URLs pinned on Pinterest, etc. We introduce our Interest-Based Recommender System (IBRS). Knowledge-based recommender systems are used to match users based on information that links the two, but they often focus on a single, specific application domain, such as movies to watch or music to listen to. IBRS however is a knowledge-based recommender system that provides recommendations that are generic in three dimensions: IBRS is (1) domain-independent, (2) language-independent, and (3) independent of the used social media. To match user interests with items, the first are derived from the user’s social media profile, enriched with a deeper semantic embedding obtained from the generic knowledge-base DBpedia. These interests are used to select personalized recommendations from a tagged item set from any domain, in any language. We also present the results of a validation of IBRS by a test user group of 44 people using two item sets from different domains: greeting cards and, most importantly for the GRS application, holiday homes.

This chapter is based on [27]

7.1 Introduction

The aim of a recommender system (RS) is to help people find the items they are most interested in. A requirement to provide personalized recommendations is that the RS has knowledge of the person using it. In 2013, Facebook
claimed to have 1.11 billion active users [30], and the top-100 pages alone currently have a total of 5.87 billion Facebook-likes [9]. Those items that people express a preference for on social media, whether through a like of a Facebook page, a follow on Twitter, or a pin on Pinterest, can be taken to disclose personal traits of interest and the things they want to be associated with. This vast amount of information is the starting point for our Interest-Based Recommender System (IBRS), where we use a knowledge-base to link social media profiles to tagged item sets.

The creation of an RS that makes use of social media or knowledge-bases is not a new ambition. In this chapter, it is our aim to do this in a way that is independent of: (1) the item set domain, (2) the languages used in the item set and the social media profiles, and (3) the type of used social media. This means that the solution shall not rely on domain-specific features, such as the actor-movie relationship or artist-album relationship. It shall also be applicable to any language, and be able to recommend items with semantic information in a different language than the used social media. And lastly, it shall not rely on features that do not generally occur in social media, such as the availability of the job history in LinkedIn or location check-in history as it was available in FourSquare in the past. This leads to the following hypothesis that is validated in this chapter: independent of the item set domain, language and used social medium, automated user interest detection can select preferred items.

The cold-start problem, as for example discussed by Bobadilla et al. [13], is a common challenge in RSs. This is the problem that data scarcity makes it impossible to (1) recommend items to new users, (2) recommend newly added items to existing users, and (3) use new systems with only a few matches between users and items. As a boundary requirement to our solution, this cold-start problem needs to be circumvented.

People’s expressed preferences on social media cannot always directly be related to commonly used tags or words in descriptions in an existing item set. These items are often example instances of broader concepts. For example: Cristiano Ronaldo has 103 million Facebook-likes at the time of writing, whereas Soccer (66 million) and Football (46 million) have considerably fewer. Tag sets or descriptions, on the other hand, are more likely to contain these broader concepts, as for example is the case in greeting cards, sports equipment, or campsites with soccer fields. In fact, one of our validation item sets contains tagged greeting cards with practically only generic terms such as soccer/football. To bridge this generalization gap in a domain- and language-independent way, we use the multilingual, generic knowledge-base DBpedia as a query expansion mechanism to automatically detect broader concepts. We call these concepts the user’s interests.

1Synonyms like this one cause problems as well, and are discussed in more detail in Section 7.3
7.2 Related work

A user group of 44 people tested IBRS, using item sets from two completely different domains: greeting cards and holiday homes. Both the recommendation selection, as well as the explanation interface were validated by these users, applying their own social media profile.

This chapter is further structured as follows: related work is discussed in Section 7.2, the motivation behind this research is discussed in Section 7.3, the IBRS technology is presented in Section 7.4, while the validation approach and results are laid out in Section 7.5, and Section 7.6 finally contains concluding remarks.

7.2 Related work

The cold-start problem is frequently described in the RS domain, and refers to the problem of data scarcity when trying to find similarities between users, similarities between items, or matches between users and items. Bobadilla et al. split up the cold-start problem into three categories \[\text{[13]}\]:

- new items;
- new users, and;
- new communities (i.e. new systems).

Different RS types have been discussed in literature. We follow the categorization by Burke and Ramezani \[\text{[17]}\], and discuss to what extent these system types suffer from the different cold-start problem types:

- In a collaborative filtering RS items are recommended based on user similarity, by locating peer users with a rating history similar to the current user and generate recommendations using this neighborhood.
  The drawbacks of a CF-based RS are that new items have not been linked to any users yet, new users cannot be compared with existing users yet, and new communities suffer from both.

- A content-based RS, recommends items based on features associated with products and the ratings that a user has given them.
  The drawbacks of a content-based RS are that new items with new features cannot be compared with other items yet and new users have not rated enough features to base recommendations on yet.

- A demographic RS recommends item based on a demographic profile of the user: recommended products can be produced for different demographic niches, by combining the ratings of users in those niches.
  A demographic RS also suffers from the difficulty to recommend new items, as it has not been matched with any users yet. New users can be related to existing users however, as long as enough comparable users are available in the system, which relates to the new community problem.
• A knowledge-based RS recommends items based on inferences about a user’s needs and preferences.

These systems do not suffer from the cold-start problem, but often require domain knowledge to provide useful recommendations.

Social media have received especially much attention in the field of content-based RSs. Fijałkowski and Zatoka presented an architecture of a recommender system for e-commerce based on Facebook profiles [32]. Guy et al. proposed five recommender types, based on social media and/or tags [41]. In their approach, they also presented the users with recommendation explanation. The social media they focused on, however, are not of the mainstream type, but specific for the Lotus Connections suite. The system of He et al., on the other hand, used common social media [44]. Whereas they claim to overcome the cold-start problem, their system still appears to suffer from the cold-start problem with respect to the addition of items.

The creation of an RS based on DBpedia has also received quite some attention, especially in the field of music [74, 75] and movie [65, 35, 64, 70, 84] recommendation. Di Noia et al. took it a step further and also benefited from the integration of DBpedia in the linked open data (LOD) initiative. Their movie recommendations are not only based on DBpedia knowledge, but also on Freebase and LinkedMDB. A more generic approach to create an RS using LOD was done by Heitmann and Hayes [45], who also use LOD to overcome the cold-start problem. Even though their validation is based on a music dataset, their approach has the genericity to be used for other applications as well. Our approach for broader concept detection through DBpedia is a form of knowledge-based query expansion. Liang et al. already showed in [58] that document recommendation based on the user’s interests improves as a result of query expansion, or semantic expansion as they call it.

What distinguishes our approach from other RS research, is that we use both social media profiles and DBpedia data to create a generic RS. Passant and Raimond, for example, created an RS based on exported social media profiles and DBpedia data in [75], but their approach is limited to the music-specific relations in DBpedia. To the best of our knowledge, the only other generic approach is TasteWeights by Bostandjiev et al. [15]. They built a user profile based on social media data, and then applied a collaborative filtering-based approach to select recommendations.

As it is exactly our goal to overcome the cold-start problem, our approach is knowledge-based, according to the RS classification by Burke and Ramezani [17]. Basile, Lops et al. would classify IBRS as a top-down semantics-aware, content-based RS [62, 11].

Our recommendation extraction work is inspired by Shi et al.’s HeteRecom [80], which is based on the similarity calculation HeteSim [79]. In their approach, they find a match between users and items using the knowledge
of only one domain (for example conference recommendation for authors based on their previous papers), but the HeteSim calculation could also be applied to cross-domain recommendation. In our work however, we focus on combining multiple graphs (i.e. the social media graphs, knowledge-base graphs, and a tagged item set) into one (unweighted) graph, and the detection of existing paths in this combined graph.

7.3 Motivation

We aim to select recommendations in a way that is generic in three dimensions: the recommendation approach shall be independent of (1) the item set domain, (2) the natural item set language, and (3) the used social medium. As a fourth criterium, it shall not suffer from any of Bobadilla’s three cold-start problem categories. Below, we discuss the motivation for all of these challenges:

Domain-independence

As discussed in the previous section, currently most recommender systems based on knowledge-bases and social media are focused on one specific domain. Independence of the item set domain allows the reuse of the solution and its future improvements for multiple applications.

Language-independence

Similar to domain-independence as a requirement for reusability, a language-independent solution improves the RS’s potential to be used in multiple applications. A sub-requirement of language-independence is synonym-independence. As Zanardi and Capra pointed out in [93], synonyms are a typical RS problem, especially for tag-based RSs. The example of people facebook-liking either the Soccer page or the Football page from Section 7.1 already showed that people may facebook-like different pages, while referring to the same concept. Despite recent efforts by Facebook to merge pages about the same topic from different languages into one page, and improving the search functionality to help people finding such pages while searching for their name in a different language, concept duplication in pages still exists.

Social medium-independence

From the first form of genericity, domain-independence, follows another requirement. Several social media, such as Facebook, LinkedIn, Twitter, Instagram, and Pinterest, are widely used, and each of these has its own focus. When one decides to create an RS for job vacancies, LinkedIn may be
a more logical social medium to base the recommendations on than any of the other, while an RS for touristic hotspots will most likely lead to another choice. Therefore, to create an RS based on social media content that is domain-independent, it shall also be independent of the underlying social medium.

**Cold-start problem**

As discussed in Section 7.2, knowledge-based RS have been designed to overcome all of these problems, but often require domain-specific knowledge.

Overcoming all of these four challenges at the same time has motivated us to create IBRS: a domain-independent, language-independent, social medium-independent, knowledge-based RS.

### 7.4 Concept & technology

The foundation of IBRS is the idea that people are more likely to be interested in items that have a not too distant relation with named entities we know they like. Although things people express a preference for on social media are typically in a different domain than our item set, they may still give hints towards a person’s interests. In IBRS, we link the preferred items (i.e. pages only, on social media to resources in the DBpedia Resource Description Framework (RDF) graph. We use this graph to explore related concepts, which are then matched with a known tag set, that is used to label
the item set. As a final step, we rank the item set based on the number of matched tags. This concept is illustrated, using the holiday home domain, in Figure 7.1. In this example, the user facebook-liked the Colosseum, pizza, and Francesco Totti. These facebook-likes are mapped onto DBpedia, and the DBpedia RDF graph is explored to detect the broader concepts Rome, Italy, and Stadio Olimpico. These items are mapped onto holiday home tags, to ultimately match the user with a specific holiday home.

The remainder of this section is structured as follows: RDF graph exploration is discussed in Section 7.4.1. The data model of the IBRS abstraction layer is presented in Section 7.4.2. Section 7.4.3 presents a method for automated tag generation from descriptions. In Section 7.4.4 the ranking mechanism and Facebook-DBpedia mapping approach are presented. Section 7.4.5 finally, presents a short introduction of the IBRS prototype.

### 7.4.1 DBpedia graph exploration

After matching a facebook-like with a DBpedia resource, we traverse the RDF graph in exactly two steps. Since RDF tuples have a subject, predicate and object, RDF graphs are directed. Therefore, there are four possible different direction combinations to travel from some node \( A \) through node \( B \) to its second neighbor \( C \)

\[^{2}\] In Table 7.1 we show the top-10 of second neighbors when traversing the DBpedia graph starting from the Eiffel Tower as node \( A \), using all four possible direction combinations. DBpedia pages in italics also occur as tags in at least one of our two validation sets, which are discussed in detail in Section 7.5. The first approach, \( A \rightarrow B \rightarrow C \), leads to results describing France, influential French people, and several other buildings in France. The second approach, \( A \leftarrow B \rightarrow C \), has some overlap with the first approach, but also contains several results unrelated to France, such as Los Angeles and the United States. The third approach, \( A \leftarrow B \leftarrow C \), shows some remarkable buildings throughout Europe, but also very unrelated lists towards the bottom of the top-10. The fourth and final approach, \( A \rightarrow B \leftarrow C \), results in several famous French people, especially scientists. Other starting points show similar results: the third approach, \( A \leftarrow B \leftarrow C \), shows promising results for single domain recommendations, whereas the first approach shows the best results for broader concept detection. Since our aim is to match these second neighbors with a tag set, we use the first approach, \( A \rightarrow B \rightarrow C \). This is caused by the type of commonly used relationships in DBpedia. Table 7.2 shows a list of the most popular predicates in the DBpedia dataset, after removal of administrative relationships such as dbpont:wikiPageID. DBpedia resources about people often refer to the city where they were born using the person as the object,

\[^{2}\] Depending on the directions of the relationships, and the existence of bi-directional relationships, node \( A \) may be equal to node \( C \), as can also be seen in Table 7.1.
and the city as the subject (and either \textit{dbpont:birthPlace}, \textit{dbp2:birthPlace}, or \textit{dbpont2:birthPlace} as the predicate). The relationship the other way around is not available. The same goes for the way generalizing predicates, such as \textit{dbpont2:genre} or \textit{rdfs:subClassOf}, are used.

<table>
<thead>
<tr>
<th>Rank</th>
<th>$A \rightarrow B \rightarrow C$ (#)</th>
<th>$A \leftarrow B \rightarrow C$ (#)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Paris (20)</td>
<td>Eiffel Tower (41)</td>
</tr>
<tr>
<td>2</td>
<td>France (20)</td>
<td>France (17)</td>
</tr>
<tr>
<td>3</td>
<td>Eiffel Tower (7)</td>
<td>Paris (15)</td>
</tr>
<tr>
<td>4</td>
<td>Manuel Valls (6)</td>
<td>Los Angeles (4)</td>
</tr>
<tr>
<td>5</td>
<td>François Hollande (6)</td>
<td>British Library (4)</td>
</tr>
<tr>
<td>6</td>
<td>Unitary state (6)</td>
<td>Bonnétable (4)</td>
</tr>
<tr>
<td>7</td>
<td>\textit{French language} (6)</td>
<td>Aarhus University (4)</td>
</tr>
<tr>
<td>8</td>
<td>Anne Hidalgo (6)</td>
<td>Garabit viaduct (4)</td>
</tr>
<tr>
<td>9</td>
<td>Bonnétable (4)</td>
<td>St Paul’s Cathedral (4)</td>
</tr>
<tr>
<td>10</td>
<td>Garabit viaduct (4)</td>
<td>United States (4)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>$A \leftarrow B \leftarrow C$ (#)</th>
<th>$A \rightarrow B \leftarrow C$ (#)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Eiffel Tower (7)</td>
<td>Paul Langevin (51)</td>
</tr>
<tr>
<td>2</td>
<td>Palácio de Ferro (3)</td>
<td>Léon Foucault (48)</td>
</tr>
<tr>
<td>3</td>
<td>Cologne Cathedral (2)</td>
<td>Jean Témerson (48)</td>
</tr>
<tr>
<td>4</td>
<td>Eiffel Bridge, Ungheni (2)</td>
<td>Frédéric Passy (45)</td>
</tr>
<tr>
<td>5</td>
<td>Souleuvre Viaduct (2)</td>
<td>L.A. de Bougainville* (45)</td>
</tr>
<tr>
<td>6</td>
<td>Samuel Hibben (2)</td>
<td>Cecile de Brunhoff (45)</td>
</tr>
<tr>
<td>7</td>
<td>Casa de Fierro (2)</td>
<td>Adrien-Marie Legendre (45)</td>
</tr>
<tr>
<td>8</td>
<td>Modern Marvels episodes* (2)</td>
<td>Robert Perrier (45)</td>
</tr>
<tr>
<td>9</td>
<td>Monopoly editions USA* (2)</td>
<td>Paul Lévy (math.)* (45)</td>
</tr>
<tr>
<td>10</td>
<td>Garabit viaduct (2)</td>
<td>Émile Drain (45)</td>
</tr>
</tbody>
</table>

Table 7.1: Top-10 of second neighbor nodes $C$ through DBpedia graph exploration in multiple directions for the \textit{Eiffel Tower} resource as node $A$. Numbers between brackets indicate number of paths between that node and the Eiffel Tower node. Items in italics also occur as tags in at least one of our two validation tag sets. Items marked with an asterisk are abbreviated.

### 7.4.2 Abstraction layer data model

To ensure IBRS genericity, an abstraction layer is used on top of the underlying data source, such as a product database. This abstraction layer can consist of physical tables, views, or a mix of these, but we will refer to its items as tables from here on. The abstraction layer, depicted in Figure 7.2 contains the following tables:
Table 7.2: Commonly used predicates in DBpedia, after filtering out administrative predicates, such as `dbpont:wikiPageID`.

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Frequency</th>
<th>Predicate</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>dc:language</td>
<td>11,519,143</td>
<td>dbp2:length</td>
<td>731,472</td>
</tr>
<tr>
<td>foaf:primaryTopic</td>
<td>11,519,143</td>
<td>foaf:homepage</td>
<td>686,240</td>
</tr>
<tr>
<td>foaf:isPrimaryTopicOf</td>
<td>11,519,143</td>
<td>dbpont:country</td>
<td>671,062</td>
</tr>
<tr>
<td>foaf:name</td>
<td>4,220,793</td>
<td>dbpont:numGoals</td>
<td>611,328</td>
</tr>
<tr>
<td>dbp2:hasPhotoCollection</td>
<td>4,041,585</td>
<td>dbp2:clubs</td>
<td>585,387</td>
</tr>
<tr>
<td>dbp2:name</td>
<td>4,021,368</td>
<td>dbp2:location</td>
<td>582,932</td>
</tr>
<tr>
<td>skos:broader</td>
<td>2,485,421</td>
<td>dbp2:rd1Team</td>
<td>582,111</td>
</tr>
<tr>
<td>dbpont:team</td>
<td>1,571,760</td>
<td>dbp2:genre</td>
<td>577,304</td>
</tr>
<tr>
<td>foaf:depiction</td>
<td>1,470,653</td>
<td>dbpont:numMatches</td>
<td>539,838</td>
</tr>
<tr>
<td>dbpont:birthPlace</td>
<td>1,211,868</td>
<td>dbp2:caption</td>
<td>516,094</td>
</tr>
<tr>
<td>dbp2:shortDescription</td>
<td>1,194,274</td>
<td>dbpont:deathDate</td>
<td>506,983</td>
</tr>
<tr>
<td>dbpont:birthDate</td>
<td>1,131,616</td>
<td>dbp2:votes</td>
<td>501,292</td>
</tr>
<tr>
<td>foaf:givenName</td>
<td>1,129,044</td>
<td>dbp2:birthDate</td>
<td>475,996</td>
</tr>
<tr>
<td>foaf:surname</td>
<td>1,103,069</td>
<td>dbpont:deathYear</td>
<td>473,486</td>
</tr>
<tr>
<td>dbpont:birthYear</td>
<td>1,028,894</td>
<td>dbp2:type</td>
<td>471,647</td>
</tr>
<tr>
<td>dbp2:dateOfBirth</td>
<td>1,023,951</td>
<td>dbp2:dateOfDeath</td>
<td>471,542</td>
</tr>
<tr>
<td>dbp2:subdivisionName</td>
<td>1,004,294</td>
<td>dbp2:caps</td>
<td>468,331</td>
</tr>
<tr>
<td>dbp2:goals</td>
<td>969,216</td>
<td>w3:geometry</td>
<td>452,531</td>
</tr>
<tr>
<td>dbp2:placeOfBirth</td>
<td>903,529</td>
<td>w3:long</td>
<td>452,212</td>
</tr>
<tr>
<td>dbp2:birthPlace</td>
<td>890,716</td>
<td>w3:lat</td>
<td>452,212</td>
</tr>
<tr>
<td>dbp2:years</td>
<td>826,515</td>
<td>georss:point</td>
<td>450,664</td>
</tr>
<tr>
<td>dbpont:isPartOf</td>
<td>826,399</td>
<td>owl:equivalentClass</td>
<td>448,766</td>
</tr>
<tr>
<td>dbpont:careerStation</td>
<td>781,814</td>
<td>rdfs:subClassOf</td>
<td>446,982</td>
</tr>
<tr>
<td>dbpont:years</td>
<td>731,778</td>
<td>dbpont:genre</td>
<td>446,237</td>
</tr>
<tr>
<td>dbp2:date</td>
<td>731,472</td>
<td>dbp2:wordnet_type</td>
<td>437,797</td>
</tr>
</tbody>
</table>

- **ABSTRACT_ITEMS**, containing the IDs and type of objects to recommend;
- **TAGS**, which holds the reference to the DBpedia resource for one or more **TAG_TRANSLATIONS**;
- **ABSTRACT_ITEMS_TAGS**, a relationship table between the **ABSTRACT_ITEMS** and the **TAGS**;
- **TAG_TRANSLATIONS**, containing the translated version of a tag in a specific language, and;
- **LANGUAGES**, containing the different languages used in the IBRS instance.
7. Knowledge-based recommendation selection

Figure 7.2: Abstraction layer data model: ABSTRACT_ITEMS are the items to recommend, linked to TAGS through the ABSTRACT_ITEMS_TAGS relationship table. TAGS are containers with one or more TAG_TRANSLATIONS in several LANGUAGES.

7.4.3 Tag generation

In case an item set is not tagged, but does contain descriptive texts, tags can be extracted automatically. Natural language processing algorithms can be used for this purpose, such as the named entity extraction and disambiguation approach by Habib et al. [43]. We used Habib’s approach with a manually trained model to extract named entities from the EuroCottage holiday home descriptions from our inspirational scenario. A drawback of this approach is that descriptions are often the result of free-text input. Phrases such as “only a 3 hour flight from Amsterdam” or “25 kilometers from the border with France” lead to correctly extracted named entities, but semantically not the best tags to distinguish this object from others. Therefore, we additionally removed those tags that tagged a holiday home with another country than the one it is located in. In total, this approach allowed us to assign 455,777 (non-unique) tags to 42,148 holiday homes, from which 106,430 tags (of which 12,151 unique) could be mapped onto a DBpedia resource.

7.4.4 Ranking

The IBRS ranking method consists of four steps: (1) retrieving preferred items from social media, (2) matching these items with DBpedia resources, (3) extracting abstracts from DBpedia, (4) ranking items based on matched tags.
Obtaining preferred items from social media

To map social media items while remaining independent of the social medium, we must take into account that APIs have differences. Some social medium APIs allow developers to find out what a user’s friends prefer, while others limit the developer to information about the logged in user. Therefore, when using the Facebook Graph API, we limited ourselves to the NAME and CATEGORY elements of each facebook-liked page. These two elements can be expected in the API of any social medium, although the name of the latter may differ (e.g. LinkedIn’s INDUSTRIES field for companies [60]).

Matching social media items with DBpedia resources

Facebook-likes are mapped onto DBpedia resources through their name. Those facebook-pages that mapped onto ambiguous terms in DBpedia were filtered out. To create a more complete mapping, we added one Facebook-specific mechanism: we postfixed some of the name elements based on the contents of the category elements. If the category element contained the value “movie,” “tv show,” or “musician/band,” we also checked if a DBpedia resource existed with the additional postfix “_(movie),” “_(TV_series),” or “_(band)” respectively. This leads to the following SPARQL query for matching the movie The Net:

```sparql
PREFIX dbpont2: <http://dbpedia.org/ontology/>
PREFIX dbpres: <http://dbpedia.org/resource/>
# We use the prefixed versions here for readability
SELECT ?uri ?label
WHERE {
  # Find exact match with category suffix
    FILTER(?uri = dbpres:The_Net_(movie)) }
  # Or exact match without category suffix
    FILTER(?uri = dbpres:The_Net) }
  # Or the label version
  UNION { ?uri rdfs:label "The_Net"@en. }
  # Check if page has redirect
  UNION { ?uri dbpres:The_Net_(movie)
    dbpont2:wikiPageRedirects ?uri }
  UNION { ?uri dbpres:The_Net
    dbpont2:wikiPageRedirects ?uri }
  FILTER (langMatches(lang(?label),"en")).
  # Filter out ambiguous terms
  FILTER NOT EXISTS { ?uri
dbpont2:wikiPageDisambiguates ?disambiguates }. }
# Filter out Wikipedia categories
```
Using this approach on a test set of 11,674 unique Facebook pages, obtained from the likes of 309 users, we were able to match 2,240 (19.2%) Facebook-pages with a DBpedia resource.

**Extracting abstracts from DBpedia**

For all matched DBpedia resources, the abstracts are retrieved from the SPARQL endpoint provided by DBpedia [23] using the following query for a fictional person with the facebook-likes Vienna, Recommender system, and Computer science:

```
PREFIX dbpont2: <http://dbpedia.org/ontology/>
PREFIX dbpres: <http://dbpedia.org/resource/>

SELECT DISTINCT
?o3 (count(?o3) as ?count) ?abstract ?label
WHERE {
    # UNION concatenation of mapped FB pages
    {dbpres:Vienna ?p1 ?o2} UNION
    {dbpres:Recommender_system ?p1 ?o2} UNION
    {dbpres:Computer_science ?p1 ?o2}
    # Neighboring object has Wikipage
    ?o2 dbpont2:wikiPageID ?o2id ;
    # Neighboring object has neighbor
    # Second neighbor object has Wikipage
    ?o3 dbpont2:wikiPageID ?o3id ;
    dbpont2:abstract ?abstract ;
    rdfs:label ?label .
    # English is used as an example
    FILTER(langMatches(lang(?abstract), 'en')) .
    FILTER(langMatches(lang(?label), 'en')) .
    # Second neighbor object must not be a category
    MINUS {?o3 rdf:type skos:Concept}
}
# 'Only' the 1000 most important abstracts
ORDER BY DESC(?count)
LIMIT 1000
```

**Ranking items based on matched tags**

Each tag that (1) has a DBPEDIA RESOURCE_ID and (2) is contained in at least one of the downloaded abstracts, is marked as a matched tag. The item set is then ranked on the basis of the number of matched tags. As a final step, those items that are too close to a higher ranked item, based on a pre-defined distance function, such as Euclidean distance for geographic objects, are removed from the ranking. This last step is added to ensure
7.5. Validation

For objects for which such a distance function is not so obvious, the cosine similarity (as for example discussed in [93]) of the item’s tags may be a good starting point. The tag input makes our RS domain-aware. However, since the approach can be applied to any tag domain, we still consider the concept itself domain-independent. This in contrast to for example music recommenders that rely on the artist-song relationship.

7.4.5 Prototype

For demonstration and validation purposes, we have created a prototype of IBRS, using the Cake PHP platform. The prototype can be used with either one’s own Facebook profile, or by manually combining several DBpedia resources. It can be accessed through http://ibrs.ewi.utwente.nl, where IBRS can be used for both domains.

7.5 Validation

To validate our ranking mechanism, as well as to determine the user perception of recommendations with explanations, we validated IBRS in a carefully designed user study with a test user group of 44 people. We used two product sets from different domains to demonstrate its domain-independence: greeting cards and holiday homes. The greeting card set contains Dutch tags, while the holiday homes did not contain any tags, but only descriptions. From the holiday homes, we used the English descriptions to extract (English) tags, to emphasize the potential to use IBRS in a language-independent way.

This section is further structured as follows: Section 7.5.1 describes the item set details. In Section 7.5.2 we present the approach taken to validate both our ranking mechanism and the recommendation explanation interface. Section 7.5.3 finally, discusses the validation results.

7.5.1 Item set details

The first item set contains greeting cards from the Dutch company Kaartje2Go (“Card2Go”). Customers search through an electronic collection of cards, which are distributed through regular paper mail by Kaartje2Go in name of the customer. The customers can choose between sending greeting cards to one or multiple people at once. 75% of the purchases are of the latter type, for which the preferences of the sender are more relevant than those of the (potentially many) recipients. To facilitate the search, users can search [http://www.kaartje2go.nl](http://www.kaartje2go.nl)
7. Knowledge-based recommendation selection

for tags that have been entered manually by the Kaartje2Go staff. These tags, which are mostly in Dutch, are inconsistent in their completeness: for example some of the soccer cards are also tagged using the names of popular Dutch soccer teams, but not all of them. Less popular teams are never mentioned as tags. The translation of the top-10 greeting card tags can be found in Table 7.3.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Birthday</td>
<td>7,535</td>
</tr>
<tr>
<td>Party</td>
<td>4,200</td>
</tr>
<tr>
<td>Love</td>
<td>2,521</td>
</tr>
<tr>
<td>Girl</td>
<td>2,268</td>
</tr>
<tr>
<td>Boy</td>
<td>2,084</td>
</tr>
<tr>
<td>Infant</td>
<td>2,056</td>
</tr>
<tr>
<td>Photograph</td>
<td>1,793</td>
</tr>
<tr>
<td>Marriage</td>
<td>1,543</td>
</tr>
<tr>
<td>Cool</td>
<td>1,381</td>
</tr>
<tr>
<td>Animals</td>
<td>1,373</td>
</tr>
</tbody>
</table>

Table 7.3: Translation of top-10 manual greeting card tags with a DBpedia reference, ordered by the number of cards with this tag

The second item set contains holiday homes from the holiday home portal EuroCottage. This item set did not contain tags, but a description in one, two or three languages (Dutch, English and/or German). We followed the approach discussed in Section 7.4.3 to extract mentions of geographic places from the English holiday home descriptions. The top-10 of resulting tags can be found in Table 7.4. The advantage of extracting geographic places is that these also often have Wikipedia pages, which makes them suitable for the requirement that the tags need to have a DBPEDIA_RESOURCE_ID. Many pages of the holiday home descriptions were in German, even though they were marked in the system by the holiday home owners as being English descriptions. As a result, many German words or phrases were extracted as geographical references, since the model was trained for English descriptions. However, the impact of these terms was practically zero, as these extracted tags were not matched with an English DBpedia resource (as the language of the TAG_TRANSLATIONS entry was set to English). For the validation, the holiday homes were plotted on a map that was zoomed in on Europe, since most holiday homes in the set are located there. A relatively small subset of homes outside Europe could therefore not be displayed on the map, and were removed from the validation set, just as those without a coordinate pair. This coordinate pair
was also used for the described diversity function: all top-10 holiday homes had to be located at least 250 kilometers away from higher ranked homes.

For both sets, we removed tags with a negative connotation, such as “die,” or “death.”

<table>
<thead>
<tr>
<th>Tag</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Florence</td>
<td>760</td>
</tr>
<tr>
<td>Siena</td>
<td>656</td>
</tr>
<tr>
<td>Mediterranean Sea</td>
<td>634</td>
</tr>
<tr>
<td>Tuscany</td>
<td>537</td>
</tr>
<tr>
<td>Legoland</td>
<td>513</td>
</tr>
<tr>
<td>Venice</td>
<td>508</td>
</tr>
<tr>
<td>Sotkamo</td>
<td>448</td>
</tr>
<tr>
<td>Europe</td>
<td>440</td>
</tr>
<tr>
<td>Ardennes</td>
<td>421</td>
</tr>
<tr>
<td>Pisa</td>
<td>363</td>
</tr>
</tbody>
</table>

Table 7.4: Top-10 of extracted tags for holiday homes with a DBpedia reference, ordered by the number of holiday homes with this tag

7.5.2 Validation approach

Our test users were requested through a Facebook post by two of the researchers involved in this project to participate in the validation experiment. The test users used their own existing Facebook account for the recommendations. The test users were not aware of what they were testing, except for the information that they were testing an RS. Most test users do not have a background in computer science, and none of them were aware of how IBRS works. We asked our test users to validate our algorithm through a total of 30 questions, split up into three batches of 10. Once a question had been answered, users could not return to that question. The first two batches were intended to validate our ranking mechanism, the third batch was intended to determine the user perception of recommendations with explanations, as compared to recommendations without explanations.

For the first ten questions, users were asked to select their favorite greeting card from a greeting card pair using the interface of Figure 7.3. On one side of the screen, an item from the top-10 greeting cards according to IBRS was shown. On the other side, a card was shown that was not tagged with any of the matched tags. We called this inverted approach Inverted IBRS. IBRS and Inverted IBRS were shown on the left or right side at random.
For the second batch of ten questions, our test users were presented with the choice between two holiday homes, in a similar way. Again, IBRS and Inverted IBRS were shown on the left or right side at random. For each holiday home, its location was shown on a map, with the name of the holiday home and the first 1000 characters of its description, as shown in Figure 7.4.

Figure 7.4: Test user interface for holiday home comparison

The final batch of ten questions required the test users to rate a recommendation. Each of the holiday homes was one of the top-10 holiday homes according to IBRS. At random, a user was assigned to the group of users...
who received recommendations with an explanation, as shown in Figure 7.5, or without an explanation.

Figure 7.5: Cut-out of test user interface for holiday home recommendation rating. The lines in orange/blue contain the matched tags.

In test runs of the validation process, we determined that in a set-wise comparison of the two systems, users tended to prefer the set that was spread out over the map, rather than one that contained clusters of recommendations. Since Inverted IBRS is extremely spread out, due to the fact that items had no relation with the users or each other, this caused a bias in the validation results. Therefore, we decided to only compare the results item-wise.

7.5.3 Validation results

The first two batches of the validation were used to determine the potential of the IBRS ranking mechanism. The results are shown in the pie charts of Figure 7.6. Figure 7.6a shows which system was the test user’s preferred system, based on a majority vote between the two systems. Most users participated in the validation of both the recommendation of greeting cards and holiday homes. Each batch was counted separately. Some 47% of the users preferred IBRS, 22% voted equally often for both of the systems, and 31% of the users preferred Inverted IBRS. In the pie chart of Figure 7.6b, the results are shown when the results of holiday homes with the greeting cards are combined per user. Since this increases the number of votes per
user, ties are less common. In this scenario, 55% of the users preferred the IBRS results, while 34% preferred Inverted IBRS.

(a) Split out between greeting cards and holiday homes (batches counted separately)

(b) Overall (batches combined)

Figure 7.6: Most frequent choices per user for the first two batches of questions

The final batch of the validation was used to determine the usefulness of the proposed recommendation explanation interface for holiday homes. The results of this batch are shown in the histograms of Figure 7.7. Using the 5-point Likert scale, the users who were presented with an interface with explanations rated 167 recommendations with an average score of 3.3772, while users without recommendation explanation rated 172 recommendations with a 3.4709 on average. As this difference falls within the normal margins of variation for this experiment size, we cannot conclude that one interface is clearly preferred over the other one.

Despite satisfying results with respect to the system’s potential to rank recommendations for users, we should not forget that many aspects play a role in the decision-making that cannot (yet) be detected from Facebook profiles. When choosing an item, one will always look at domain-specific item characteristics. For a greeting card, the user looks at colors, style, and the occasion the card is sent for. Similarly, for a holiday home, he looks at price, number of beds, the picture of the home, and the distance to the beach. For this reason, this approach is only intended to be used as a feature of a larger system, taking also these characteristics into account, just as well as a user’s previous purchase and/or browsing behavior.

7.6 Conclusion

In this chapter, we presented the IBRS recommendation technique. We discussed the concept of mapping items marked as preferred or liked in social media onto a generic knowledge-base, and query expansion using DBpedia. We presented the technology, including the abstraction layer, tag generation approach, and ranking mechanism. We also presented the
validation results of a user study. Users preferred items that were marked with tags that, through IBRS’s query expansion technique through DBpedia could be related to their facebook-likes over those that were not. Our user interface with recommendation explanation, however, did not turn out to be clearly preferable over an interface without it. In fact, this interface was slightly less favored, but still within the normal margins of variation for our experiment size.

We recommend to use the proposed recommendation technique from this chapter as a feature of a larger recommendation component for a GRS. In such a component, one also needs to take domain-specific features, as well as item popularity and other collaborative filtering features, into account. As discussed in Chapter 2, the POI visits extracted from the GPS signal, using many of the techniques discussed in this thesis, can also be used as a starting point for the recommendation process.
Conclusion

In this thesis, we explored the potential for recommendation of location-bound objects (LBOs) to users, based on trajectory data and social media content. While coping with the challenges of trajectory analysis, UGC quality analysis and recommendation selection, we used only public data and software libraries, to keep the threshold for using or geosocial recommender system GeoSoRS to a minimum. Using the architecture drawn up in Chapter 2, we identified several research questions for which we carried out an in-depth analysis in the respective other chapters of the thesis, to ultimately match user profiles with geoprofiles for location-bound objects. We revisit the research questions discussed in Chapter 1 in Section 8.1 suggest directions for future research on geosocial recommender systems in Section 8.2 and conclude the thesis in Section 8.3.

8.1 Research questions revisited

Throughout this chapter, we answer the research questions of Chapter 1. Below, we discuss our findings for each of them.

8.1.1 Geosocial recommender system architecture

To provide ourselves with a starting point for this research project, we needed to investigate existing work and discover potential research areas for exploration with respect to LBO recommendation by answering the question:

RQ1. Which software components can contribute to LBO recommendation based on the LBO’s geographic embedding and a user’s interests?

In Chapter 2, we presented the architecture of GeoSoRS. This architecture consists of three phases: information collection, information enrichment, and recommendation selection. For each of these components, we discussed
existing approaches and proposed possible (alternative) solutions. We also introduced the concept of geoprofiles, to describe the characteristics of a region relevant for recommendation selection in this application domain. The GeoSoRS architecture distinguishes itself from existing GRS architectures in that it is the only one, to the best of our knowledge, that is based on a combination of trajectory data and existing social media networks, such as Facebook.

8.1.2 Trajectory analysis

The locations that people visit may say something about their preferences and needs. Furthermore, the locations that are visited frequently in a certain region, say something about that region. The resulting potential to populate user profiles and geoprofiles from trajectory data led to the following research question:

**RQ2. How can visited POIs be detected from GPS traces?**

As the process of visited POI extraction is not straightforward, and several steps for data collection and enrichment are required beforehand, we split this research question in the following three subquestions:

**RQ2a. How can existing POIs be collected from the web using minimal resources, with respect to both human effort and computational complexity?**

This research question was addressed in Chapter 3, where we introduced the NeoGeo scraper. The NeoGeo scraper was designed specifically to reduce (1) the effort for scraper development and maintenance and (2) the required computational complexity, since the number of visited web pages increases rapidly with an increasing scope of information to scrape. The NeoGeo scraper is based on workflows of wrapper inducers, as introduced by Kushmeric [55]. The configuration effort for the NeoGeo scraper is limited to (1) a (potentially reusable) scraper workflow, (2) a starting point URL, and (3) a model object class. Furthermore, a pipeline mechanism is used to reduce the required computing resources.

**RQ2b. How can the size and shape of a parcel related to a POI be estimated?**

Since POIs collected from the web typically only contain an address, or a coordinate pair at best, we defined six different approaches for polygon-of-interest (POLOI) approximation in Chapter 4. We compared these using a POI test set for the city of Enschede, The Netherlands. Based on the
8.1. Research questions revisited

Validation, we concluded that “Voronoi diagrams of the OSM polygons with additional geocoding” (VOSM+), described in Section 4.5.6, is the approach that yields the best results from the ones we propose. It recalls a decent portion of the actual used POLOI and has good precision. Moreover, this approach can be carried out for areas with low POI density without loss of accuracy, and, due to the relatively low computational complexity, the calculation finishes within a reasonable amount of time. We also proposed a client-server interaction which preserves the privacy of the user. This interaction uses techniques to mask the time and exact locations, while still allowing for the creation of a user profile.

**RQ2c. How can POLOIs be used to detect POI visits from mobile trajectory data?**

Improving upon existing POI visit detection techniques was the main objective in Chapter 5, where we introduced our POI visit extraction algorithm PIE. PIE is based on (1) removal of imprecise samples, (2) sample clustering, (3) heading change filtering, and (4) projection onto POLOIs. We compared the results of our algorithm with existing ones, using smartphone trajectory data in urban areas for indoor activities. For this type of common, low quality data, we showed that we are able to infer the visited POIs with significantly better precision and recall than the approaches of Alvares [4], Palma [71], and Rocha [76].

8.1.3 UGC quality assessment

In the GeoSoRS architecture, we also allowed users to enter user-generated content (UGC), to review POIs. This information can be used both for creating stronger user profiles of the UGC creators, as well as stronger geoprofiles, for the LBOs located closely to the reviewed POI. To ensure the usage of high quality data, we attempted to answer the research question:

**RQ3. Which methods can be used to assess the quality of UGC, based on trajectory data?**

In Chapter 6, we showed that trajectory data is much more than a temporal sequence of position samples. We showed how the trustworthiness of UGC can be assessed. In four phases, information is collected, cleaned, analyzed and used for the creation of predictive models. Although these methods were derived in the context of a treasure hunt scenario, they are generally applicable for the analysis of “rich” trajectory data. The staypoint analysis can be used to detect those locations where people spend more time, from which locations can be detected that are relevant for the GRS context. The timeline analysis is useful for comparing the behavior of people, and can
lead to several types of (possibly complementary) categorizations, such as parents or commuters.

8.1.4 Recommendation selection

**RQ4. How can interests be used to select recommendations from a set of location-bound entities?**

We presented the recommendation engine IBRS in Chapter 7. We discussed how interests, automatically detected from Facebook-likes, can be mapped onto a generic knowledge-base. We also discussed how that knowledge-base, DBpedia in our case, can be used to expand the query to a broader set of interests, to finally be mapped onto a tagged item set. IBRS was created to be independent of the domain, which we demonstrated by validating it using both a greeting card set, and a holiday home set, from the case that inspired our research. We also required it to be language-independent, which we demonstrated by using Dutch tags for the greeting card set, and English tags, derived from the descriptions, for the holiday home set. We presented the technology, including the abstraction layer, tag generation approach, and ranking mechanism. We also presented the validation results of a test user group, using two potential interfaces for holiday home recommendations, and compared not only the results of the recommendation algorithm, but also the perceived recommendation relevance by test users.

8.2 Future research directions

In this thesis, we covered several aspects of geosocial recommendation, ranging from detailed trajectory data analysis to using social media profile data for recommendations. In the entire thesis, we have assumed to have no prior knowledge of customer behavior. An area that is yet to be explored is the combination of the techniques described in this thesis with such booking behavior data. For example, we referred to the example hypothesis that the booking behavior of parents differs from the booking behavior of non-parent adults. In the future, validating such hypotheses can strengthen the functionality for geosocial recommendations.

Furthermore, the potential of the trajectory analysis step from Chapter 6 could be further explored. First of all, trajectory analysis could be used for user profile population, rather than only for UGC quality assessment. Secondly, the described method requires human intelligence for pattern detection. Using prior customer online booking behavior as a ground truth for recommendations, the process of trajectory analysis may be automated as well. Features in the trajectory data that have a strong distinguishing factor for booking behavior could be detected automatically.
For the recommendation component of Chapter 7, the booking behavior data could be used for a collaborative filtering-based RS, possibly supported by the described knowledge-based RS. The website browsing and/or booking behavior of users could also be used to generate a feedback loop to create more appealing recommendations. One of our findings in Chapter 7 is that the occurrence of certain tags led to people dismissing the recommendation as irrelevant. This leads to the intuition that this can also be used to the advantage of the system, for example by finding the optimal combination of tags to present the holiday home in a positive way, through implicit feedback of users, who either click or don’t click on a recommendation with a certain tag.

8.3 Concluding remarks

Recommendations are predictions of what people may or may not like. No system can ever achieve 100% accuracy in such predictions, and this unpredictability is exactly what makes this world an interesting place to live in. The possibility to make the choices that we like, some predictable and rational, some unpredictable and based on emotions, gives us a feeling of freedom. However, in the overwhelming amount of options to choose from these days, people can use some help. As long as the privacy of people is respected, both with respect to trajectory data, as well as social media data, geosocial recommender systems can be a step forward in providing this help. With this thesis, we hope to have contributed to the goal of recommender systems: supporting people in their decision-making processes.
Bibliography


[46] IENS. http://www.iens.nl/


## SIKS Dissertation list

### 2009

<table>
<thead>
<tr>
<th>Number</th>
<th>Author(s)</th>
<th>Affiliation</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009-01</td>
<td>Rasa Jurgelenaitė (RUN)</td>
<td></td>
<td>Symmetric Causal Independence Models</td>
</tr>
<tr>
<td>2009-02</td>
<td>Willem Robert van Hage (VU)</td>
<td></td>
<td>Evaluating Ontology-Alignment Techniques</td>
</tr>
<tr>
<td>2009-03</td>
<td>Hans Stol (UvT)</td>
<td></td>
<td>A Framework for Evidence-based Policy Making Using IT</td>
</tr>
<tr>
<td>2009-04</td>
<td>Josephine Nabukwasi (RUN)</td>
<td></td>
<td>Improving the Quality of Organisational Policy Making using Collaboration Engineering</td>
</tr>
<tr>
<td>2009-05</td>
<td>Sietse Overbeek (RUN)</td>
<td></td>
<td>Bridging Supply and Demand for Knowledge Intensive Tasks - Based on Knowledge, Cognition, and Quality</td>
</tr>
<tr>
<td>2009-06</td>
<td>Muhammad Subianto (UU)</td>
<td></td>
<td>Understanding Classification</td>
</tr>
<tr>
<td>2009-07</td>
<td>Ronald Poppe (UT)</td>
<td></td>
<td>Discriminative Vision-Based Recovery and Recognition of Human Motion</td>
</tr>
<tr>
<td>2009-08</td>
<td>Volker Nannen (VU)</td>
<td></td>
<td>Evolutionary Agent-Based Policy Analysis in Dynamic Environments</td>
</tr>
<tr>
<td>2009-09</td>
<td>Benjamin Kanagwa (RUN)</td>
<td></td>
<td>Design, Discovery and Construction of Service-oriented Systems</td>
</tr>
<tr>
<td>2009-10</td>
<td>Jan Wielemaker (UVA)</td>
<td></td>
<td>Logic programming for knowledge-intensive interactive applications</td>
</tr>
<tr>
<td>2009-11</td>
<td>Alexander Boer (UVA)</td>
<td></td>
<td>Legal Theory, Sources of Law &amp; the Semantic Web</td>
</tr>
<tr>
<td>2009-12</td>
<td>Peter Massuth (TU, Humboldt-Universität zu Berlin)</td>
<td></td>
<td>Operating Guidelines for Services</td>
</tr>
<tr>
<td>2009-13</td>
<td>Steven de Jong (UM)</td>
<td></td>
<td>Fairness in Multi-Agent Systems</td>
</tr>
<tr>
<td>2009-14</td>
<td>Makym Korotkiy (VU)</td>
<td></td>
<td>From ontology-enabled services to service-enabled ontologies</td>
</tr>
<tr>
<td>2009-15</td>
<td>Rinke Hoekstra (UVA)</td>
<td></td>
<td>Ontology Representation - Design Patterns and Ontologies that Make Sense</td>
</tr>
<tr>
<td>2009-16</td>
<td>Fritz Reul (UvT)</td>
<td></td>
<td>New Architectures in Computer Chess</td>
</tr>
<tr>
<td>2009-17</td>
<td>Laurens van der Maaten (UvT)</td>
<td></td>
<td>Feature Extraction from Visual Data</td>
</tr>
<tr>
<td>2009-18</td>
<td>Fabian Groffen (CWI)</td>
<td></td>
<td>Armada, An Evolving Database System</td>
</tr>
<tr>
<td>2009-19</td>
<td>Valentin Robu (CWI)</td>
<td></td>
<td>Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets</td>
</tr>
<tr>
<td>2009-20</td>
<td>Bob van der Vecht (UU)</td>
<td></td>
<td>Adjustable Autonomy: Controlling Influences on Decision Making</td>
</tr>
<tr>
<td>2009-21</td>
<td>Stijn Vanderlooy (UM)</td>
<td></td>
<td>Ranking and Reliable Classification</td>
</tr>
<tr>
<td>2009-22</td>
<td>Pavel Serdyukov (UT)</td>
<td></td>
<td>Search For Expertise: Going beyond direct evidence</td>
</tr>
</tbody>
</table>
2009-23 Peter Hofgesang (VU)  
Modelling Web Usage in a Changing Environment

2009-24 Annerieke Heuvelink (VUA)  
Cognitive Models for Training Simulations

2009-25 Alex van Ballegooij (CWI)  
RAM: Array Database Management through Relational Mapping

2009-26 Fernando Koch (UU)  
An Agent-Based Model for the Development of Intelligent Mobile Services

2009-27 Christian Glahn (OU)  
Contextual Support of social Engagement and Reflection on the Web

2009-28 Sander Evers (UT)  
Sensor Data Management with Probabilistic Models

2009-29 Stanislav Pokraev (UT)  
Model-Driven Semantic Integration of Service-Oriented Applications

2009-30 Marcin Zukowski (CWI)  
Balancing vectorized query execution with bandwidth-optimized storage

2009-31 Sofiya Katrenko (UVA)  
A Closer Look at Learning Relations from Text

2009-32 Rik Farenhorst (VU) and Remco de Boer (VU)  
Architectural Knowledge Management: Supporting Architects and Auditors

2009-33 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin)  
Service Substitution – A Behavioral Approach Based on Petri Nets

2009-34 Ian van de Weerd (UvT)  
Multinomial Language Learning: Investigations into the Geometry of Language

2009-35 Igor Berezhnyy (UvT)  
Digital Analysis of Paintings

2009-36 Toine Bogers  
Recommender Systems for Social Bookmarking

2009-37 Virginia Nunes Leal Franqueira (UT)  
Finding Multi-step Attacks in Computer Networks using Heuristic Search and Mobile Ambients

2010-01 Matthijs van Leeuwen (UU)  
Patterns that Matter

2010-02 Ingo Wassink (UT)  
Work flows in Life Science

2010-03 Joost Geurts (CWI)  
A Document Engineering Model and Processing Framework for Multimedia documents

2010-04 Olga Kulyk (UT)  
Do You Know What I Know? Situational Awareness of Co-located Teams in Multidisplay Environments

2010-05 Claudia Hauff (UT)  
Predicting the Effectiveness of Queries and Retrieval Systems
2010-06 Sander Bakkes (UvT)  
Rapid Adaptation of Video Game AI

2010-07 Wim Fikkert (UT)  
Gesture interaction at a Distance

2010-08 Krzysztof Siewicz (UL)  
Towards an Improved Regulatory Framework of Free Software

2010-09 Hugo Kielman (UL)  
A Politieke gegevensverwerking en Privacy, Naar een effectieve waarborging

2010-10 Rebecca Ong (UL)  
Mobile Communication and Protection of Children

2010-11 Adriaan Ter Mors (TUD)  
The world according to MARP: Multi-Agent Route Planning

2010-12 Susan van den Braak (UU)  
Sensemaking software for crime analysis

2010-13 Gianluigi Folino (RUN)  
High Performance Data Mining using Bio-inspired techniques

2010-14 Sander van Splunter (VU)  
Automated Web Service Reconfiguration

2010-15 Lianne Bodenstaff (UT)  
Managing Dependency Relations in Inter-Organizational Models

2010-16 Sicco Vervier (TUD)  
Efficient Identification of Timed Automata, theory and practice

2010-17 Spyros Kotoulas (VU)  
Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications

2010-18 Charlotte Gerritsen (VU)  
Caught in the Act: Investigating Crime by Agent-Based Simulation

2010-19 Henriette Cramer (UvA)  
People’s Responses to Autonomous and Adaptive Systems

2010-20 Ivo Swartjes (UT)  
Whose Story Is It Anyway? How Improv Informs Agency and Authorship of Emergent Narrative

2010-21 Harold van Heerde (UT)  
Privacy-aware data management by means of data degradation

2010-22 Michel Hildebrand (CWI)  
End-user Support for Access to Heterogeneous Linked Data

2010-23 Bas Steunebrink (UU)  
The Logical Structure of Emotions

2010-24 Dmytro Tykhonov  
Designing Generic and Efficient Negotiation Strategies

2010-25 Zulfiqar Ali Memon (VU)  
Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective

2010-26 Ying Zhang (CWI)  
XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines

2010-27 Marten Voulon (UL)  
Automatisch contracteren

2010-28 Arne Koopman (UU)  
Characteristic Relational Patterns

2010-29 Stratos Idreos(CWI)  
Database Cracking: Towards Auto-tuning Database Kernels

2010-30 Marieke van Erp (UvT)  
Accessing Natural History - Discoveries in data cleaning, structuring, and retrieval

2010-31 Victor de Boer (UVA)  
Ontology Enrichment from Heterogeneous Sources on the Web

2010-32 Marcel Hiel (UvT)  
An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems

2010-33 Robin Aly (UT)  
Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval

2010-34 Teduh Dirghahayu (UT)  
Interaction Design in Service Compositions

2010-35 Dolf Trieschnigg (UT)  
Proof of Concept: Concept-based Biomedical Information Retrieval

2010-36 Jose Janssen (OU)  
Paving the Way for Lifelong Learning; Facilitating competence development through a learning path specification

2010-37 Niels Lohmann (TUE)  
Correctness of services and their composition
2010-38 Dirk Fahland (TUE)
From Scenarios to components

2010-39 Ghazanfar Farooq Siddiqui (VU)
Integrative modeling of emotions in virtual agents

2010-40 Mark van Assem (VU)
Converting and Integrating Vocabularies for the Semantic Web

2010-41 Guillaume Chaslot (UM)
Monte-Carlo Tree Search

2010-42 Sybren de Kinderen (VU)
Needs-driven service bundling in a multi-supplier setting - the computational e3-service approach

2010-43 Peter van Kranenburg (UU)
A Computational Approach to Content-Based Retrieval of Folk Song Melodies

2010-44 Pieter Bellekens (TUE)
An Approach towards Context-sensitive and User-adapted Access to Heterogeneous Data Sources

2010-45 Vasilios Andrikopoulos (UvT)
A theory and model for the evolution of software services

2010-46 Vincent Piijpers (VU)
e3alignment: Exploring Inter-Organizational Business-ICT Alignment

2010-47 Chen Li (UT)
Mining Process Model Variants: Challenges, Techniques, Examples

2010-48 Withdrawn

2010-49 Jahn-Takeshi Saito (UM)
Solving difficult game positions

2010-50 Bouke Huurnink (UVA)
Search in Audiovisual Broadcast Archives

2010-51 Alia Khairia Amin (CWI)
Understanding and supporting information seeking tasks in multiple sources

2010-52 Peter-Paul van Maanen (VU)
Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention

2010-53 Edgar Meij (UVA)
Combining Concepts and Language Models for Information Access

2011-01 Botond Cseke (RUN)
Variational Algorithms for Bayesian Inference in Latent Gaussian Models

2011-02 Nick Tinne Ame(ER) (UU)
Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language

2011-03 Jan Martijn van der Werf (TUE)
Compositional Design and Verification of Component-Based Information Systems

2011-04 Hado van Hasselt (UU)
Insights in Reinforcement Learning: Formal analysis and empirical evaluation of temporal-difference learning algorithms

2011-05 Base van der Raadt (VU)
Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline

2011-06 Yiwen Wang (TUE)
Semantically-Enhanced Recommendations in Cultural Heritage

2011-07 Yujia Cao (UT)
Multimodal Information Presentation for High Load Human Computer Interaction

2011-08 Nieske Vergunst (UU)
BDI-based Generation of Robust Task-Oriented Dialogues

2011-09 Tim de Jong (OU)
Contextualised Mobile Media for Learning

2011-10 Bart Bogaert (UvT)
Cloud Content Contention

2011-11 Dhaval Vyas (UT)
Designing for Awareness: An Experience-focused HCI Perspective

2011-12 Carmen Bratosin (TUE)
Grid Architecture for Distributed Process Mining
2011-13 Xiaoyu Mao (UvT)  
*Airport under Control. Multiagent Scheduling for Airport Ground Handling*

2011-14 Milan Lovric (EUR)  
*Behavioral Finance and Agent-Based Artificial Markets*

2011-15 Marijn Koolen (UvA)  
*The Meaning of Structure: the Value of Link Evidence for Information Retrieval*

2011-16 Maarten Schadd (UM)  
*Selective Search in Games of Different Complexity*

2011-17 Jiyin He (UvA)  
*Exploring Topic Structure: Coherence, Diversity and Relatedness*

2011-18 Mark Ponsen (UM)  
*Strategic Decision-Making in complex games*

2011-19 Ellen Rusman (OU)  
*The Mind’s Eye on Personal Profiles*

2011-20 Qing Gu (VU)  
*Guiding service-oriented software engineering - A view-based approach*

2011-21 Linda Terlouw (TUD)  
*Martularization and Specification of Service-Oriented Systems*

2011-22 Junte Zhang (UvA)  
*System Evaluation of Archival Description and Access*

2011-23 Wouter Weerkamp (UvA)  
*Finding People and their Utterances in Social Media*

2011-24 Herwin van Welbergen (UT)  
*Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior*

2011-25 Syed Waqar ul Qounain Jaffry (VU)  
*Analysis and Validation of Models for Trust Dynamics*

2011-26 Matthijs Aart Pontier (VU)  
*Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots*

2011-27 Aniel Bhulai (VU)  
*Dynamic website optimization through autonomous management of design patterns*

2011-28 Rianne Kaptein (UvA)  
*Effective Focused Retrieval by Exploiting Query Context and Document Structure*

2011-29 Faisal Kamiran (TUE)  
*Discrimination-aware Classification*

2011-30 Egon van den Broek (UT)  
*Affective Signal Processing (ASP): Unraveling the mystery of emotions*

2011-31 Ludo Waltman (EUR)  
*Computational and Game-Theoretic Approaches for Modeling Bounded Rationality*

2011-32 Nees-Jan van Eck (EUR)  
*Methodological Advances in Bibliometric Mapping of Science*

2011-33 Tom van der Weide (UU)  
*Arguing to Motivate Decisions*

2011-34 Paolo Turrini (UU)  
*Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations*

2011-35 Maaike Harbers (UU)  
*Explaining Agent Behavior in Virtual Training*

2011-36 Erik van der Spek (UU)  
*Experiments in serious game design: a cognitive approach*

2011-37 Adriana Burtuliu (RUN)  
*Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference*

2011-38 Nyree Lemmens (UM)  
*Bee-inspired Distributed Optimization*

2011-39 Joost Westra (UU)  
*Organizing Adaptation using Agents in Serious Games*

2011-40 Viktor Clerc (VU)  
*Architectural Knowledge Management in Global Software Development*

2011-41 Luan Ibraimi (UT)  
*Cryptographically Enforced Distributed Data Access Control*

2011-42 Michal Sindlar (UU)  
*Explaining Behavior through Mental State Attribution*

2011-43 Henk van der Schuur (UU)  
*Process Improvement through Software Operation Knowledge*
2011-44 Boris Reuderink (UT)  
Robust Brain-Computer Interfaces

2011-45 Herman Stehouwer (UvT)  
Statistical Language Models for Alternative Sequence Selection

2011-46 Beihei Hu (TUD)  
Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work

2011-47 Azizi Bin Ab Aziz(VU)  
Exploring Computational Models for Intelligent Support of Persons with Depression

2011-48 Mark Ter Maat (UT)  
Response Selection and Turn-taking for a Sensitive Artificial Listening Agent

2011-49 Andreea Niculescu (UT)  
Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality

2012

2012-01 Terry Kakeeto (UvT)  
Relationship Marketing for SMEs in Uganda

2012-02 Muhammad Umair(VU)  
Adaptivity, emotion, and Rationality in Human and Ambient Agent Models

2012-03 Adam Vanya (VU)  
Supporting Architecture Evolution by Mining Software Repositories

2012-04 Jurriaan Souer (UU)  
Development of Content Management System-based Web Applications

2012-05 Marijn Plomp (UU)  
Maturing Interorganisational Information Systems

2012-06 Wolfgang Reinhardt (OU)  
Awareness Support for Knowledge Workers in Research Networks

2012-07 Rianne van Lambalgen (VU)  
When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions

2012-08 Gerben de Vries (UVA)  
Kernel Methods for Vessel Trajectories

2012-09 Riccardo Neisse (UT)  
Trust and Privacy Management Support for Context-Aware Service Platforms

2012-10 David Smits (TUE)  
Towards a Generic Distributed Adaptive Hypermedia Environment

2012-11 J.C.B. Rantham Prabhakara (TUE)  
Process Mining in the Large: Preprocessing, Discovery, and Diagnostics

2012-12 Kees van der Suijs (TUE)  
Model Driven Design and Data Integration in Semantic Web Information Systems

2012-13 Suleman Shahid (UvT)  
Fun and Face: Exploring non-verbal expressions of emotion during playful interactions

2012-14 Evgeny Knutov(TUE)  
Generic Adaptation Framework for Unifying Adaptive Web-based Systems

2012-15 Natalie van der Wal (VU)  
Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes

2012-16 Fiemke Both (VU)  
Helping people by understanding them - Ambiant Agents supporting task execution and depression treatment

2012-17 Amal Elgamal(UvT)  
Towards a Comprehensive Framework for Business Process Compliance

2012-18 Elijo Poort(VU)  
Improving Solution Architecting Practices

2012-19 Helen Schonenberg(TUE)  
What’s Next? Operational Support for Business Process Execution

2012-20 Ali Bahramisharif (RUN)  
Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing

2012-21 Roberto Cornacchia (TUD)  
Querying Sparse Matrices for Information Retrieval
2012-22 Thijs Vis (UvT)
Intelligence, politie en veiligheidsdienst: verenigbare grootheden?

2012-23 Christian Muehl (UT)
Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction

2012-24 Laurens van der Werf (UT)
Evaluation of Noisy Transcripts for Spoken Document Retrieval

2012-25 Silja Eckartz (UT)
Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application

2012-26 Emile de Maat (UVA)
Making Sense of Legal Text

2012-27 Hayrettin Gürkök (UT)
Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games

2012-28 Nancy Pascall (UvT)
Engendering Technology Empowering Women

2012-29 Almer Tigelaar (UT)
Peer-to-Peer Information Retrieval

2012-30 Alina Pommeranz (TUD)
Designing Human-Centered Systems for Reflective Decision Making

2012-31 Emily Bagarukayo (RUN)
A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure

2012-32 Wietske Visser (TUD)
Qualitative multi-criteria preference representation and reasoning

2012-33 Rory Sie (OUN)
Coalitions in Cooperation Networks (COCOON)

2012-34 Pavol Jancura (RUN)
Evolutionary analysis in PPI networks and applications

2012-35 Evert Haasdijk (VU)
Never Too Old To Learn – On-line Evolution of Controllers in Swarm- and Modular Robotics

2012-36 Denis Ssebugwawo (RUN)
Analysis and Evaluation of Collaborative Modeling Processes

2012-37 Agnes Nakakawa (RUN)
A Collaboration Process for Enterprise Architecture Creation

2012-38 Selmar Smit (VU)
Parameter Tuning and Scientific Testing in Evolutionary Algorithms

2012-39 Hassan Fatemi (UT)
Risk-aware design of value and coordination networks

2012-40 Agus Gunawan (UvT)
Information Access for SMEs in Indonesia

2012-41 Sebastian Kelle (OU)
Game Design Patterns for Learning

2012-42 Dominique Verpoorten (OU)
Reflection Amplifiers in self-regulated Learning

2012-43 Withdrawn

2012-44 Anna Tordai (VU)
On Combining Alignment Techniques

2012-45 Benedikt Kratz (UvT)
A Model and Language for Business-aware Transactions

2012-46 Simon Carter (UVA)
Exploration and Exploitation of Multilingual Data for Statistical Machine Translation

2012-47 Manos Tsagkias (UVA)
Mining Social Media: Tracking Content and Predicting Behavior

2012-48 Jorn Bakker (TUE)
Handling Abrupt Changes in Evolving Time-series Data

2012-49 Michael Kaisers (UM)
Learning against Learning - Evolutionary dynamics of reinforcement learning algorithms in strategic interactions

2012-50 Steven van Kervel (TUD)
Ontology driven Enterprise Information Systems Engineering

2012-51 Jeroen de Jong (TUD)
Heuristics in Dynamic Scheduling; a practical framework with a case study in elevator dispatching
<table>
<thead>
<tr>
<th>Year</th>
<th>Student</th>
<th>Institution</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013-01</td>
<td>Viorel Milea (EUR)</td>
<td>News Analytics for Financial Decision Support</td>
<td></td>
</tr>
<tr>
<td>2013-02</td>
<td>Erietta Liarou (CWI)</td>
<td>MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing</td>
<td></td>
</tr>
<tr>
<td>2013-03</td>
<td>Szymon Klorman (VU)</td>
<td>Reasoning with Contexts in Description Logics</td>
<td></td>
</tr>
<tr>
<td>2013-04</td>
<td>Chetan Yadati (TUD)</td>
<td>Coordinating autonomous planning and scheduling</td>
<td></td>
</tr>
<tr>
<td>2013-05</td>
<td>Dulce Pumareja (UT)</td>
<td>Groupware Requirements Evolutions Patterns</td>
<td></td>
</tr>
<tr>
<td>2013-06</td>
<td>Romulo Goncalves (CWI)</td>
<td>The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience</td>
<td></td>
</tr>
<tr>
<td>2013-07</td>
<td>Giel van Lankveld (UvT)</td>
<td>Quantifying Individual Player Differences</td>
<td></td>
</tr>
<tr>
<td>2013-08</td>
<td>Robbert-Jan Merk (VU)</td>
<td>Making enemies: cognitive modeling for opponent agents in fighter pilot simulators</td>
<td></td>
</tr>
<tr>
<td>2013-09</td>
<td>Fabio Gori (RUN)</td>
<td>Metagenomic Data Analysis: Computational Methods and Applications</td>
<td></td>
</tr>
<tr>
<td>2013-10</td>
<td>Jeewanie Jayasinghe Arachchige (UvT)</td>
<td>A Unified Modeling Framework for Service Design</td>
<td></td>
</tr>
<tr>
<td>2013-11</td>
<td>Evangelos Pournaras (TUD)</td>
<td>Multi-level Reconfigurable Self-organization in Overlay Services</td>
<td></td>
</tr>
<tr>
<td>2013-12</td>
<td>Marian Razavian (VU)</td>
<td>Knowledge-driven Migration to Services</td>
<td></td>
</tr>
<tr>
<td>2013-13</td>
<td>Mohammad Safiri (UT)</td>
<td>Service Tailoring: User-centric creation of integrated IT-based homecare services to support independent living of elderly</td>
<td></td>
</tr>
<tr>
<td>2013-14</td>
<td>Jafar Tanha (ÜVA)</td>
<td>Ensemble Approaches to Semi-Supervised Learning Learning</td>
<td></td>
</tr>
<tr>
<td>2013-15</td>
<td>Daniel Hennes (UM)</td>
<td>Multiagent Learning - Dynamic Games and Applications</td>
<td></td>
</tr>
<tr>
<td>2013-16</td>
<td>Eric Kok (UU)</td>
<td>Exploring the practical benefits of argumentation in multi-agent deliberation</td>
<td></td>
</tr>
<tr>
<td>2013-17</td>
<td>Koen Kok (VU)</td>
<td>The PowerMatcher: Smart Coordination for the Smart Electricity Grid</td>
<td></td>
</tr>
<tr>
<td>2013-18</td>
<td>Jeroen Janssens (UvT)</td>
<td>Outlier Selection and One-Class Classification</td>
<td></td>
</tr>
<tr>
<td>2013-19</td>
<td>Renze Steenhuizen (TUD)</td>
<td>Coordinated Multi-Agent Planning and Scheduling</td>
<td></td>
</tr>
<tr>
<td>2013-20</td>
<td>Katja Hofmann (UvA)</td>
<td>Fast and Reliable Online Learning to Rank for Information Retrieval</td>
<td></td>
</tr>
<tr>
<td>2013-21</td>
<td>Sander Wubben (UvT)</td>
<td>Text-to-text generation by monolingual machine translation</td>
<td></td>
</tr>
<tr>
<td>2013-22</td>
<td>Tom Claassen (RUN)</td>
<td>Causal Discovery and Logic</td>
<td></td>
</tr>
<tr>
<td>2013-23</td>
<td>Patricio de Alencar Silva (UvT)</td>
<td>Value Activity Monitoring</td>
<td></td>
</tr>
<tr>
<td>2013-24</td>
<td>Haitham Bou Ammar (UM)</td>
<td>Automated Transfer in Reinforcement Learning</td>
<td></td>
</tr>
<tr>
<td>2013-26</td>
<td>Alireza Zarghami (UT)</td>
<td>Architectural Support for Dynamic Homecare Service Provisioning</td>
<td></td>
</tr>
<tr>
<td>2013-27</td>
<td>Mohammad Huq (UT)</td>
<td>Inference-based Framework Managing Data Provenance</td>
<td></td>
</tr>
<tr>
<td>2013-28</td>
<td>Frans van der Sluis (UT)</td>
<td>When Complexity becomes Interesting: An Inquiry into the Information eXperience</td>
<td></td>
</tr>
<tr>
<td>2013-29</td>
<td>Iwan de Kok (UT)</td>
<td>Listening Heads</td>
<td></td>
</tr>
<tr>
<td>2013-30</td>
<td>Joyce Nakatumba (TUE)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Title</td>
<td>Author</td>
<td>Institution</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Blueprint Model and Language for Engineering Cloud Applications</td>
<td>Kamakshi Rajagopal</td>
<td>OUN</td>
<td></td>
</tr>
<tr>
<td>Networking For Learning: The role of Networking in a Lifelong Learner’s Professional Development</td>
<td>Qi Gao</td>
<td>TUD</td>
<td></td>
</tr>
<tr>
<td>User Modeling and Personalization in the Microblogging Sphere</td>
<td>Kien Tjin-Kam-Jet</td>
<td>UT</td>
<td></td>
</tr>
<tr>
<td>Distributed Deep Web Search</td>
<td>Abdallah El Ali</td>
<td>UvA</td>
<td></td>
</tr>
<tr>
<td>Minimal Mobile Human Computer Interaction</td>
<td>Than Lam Hoang</td>
<td>TUE</td>
<td></td>
</tr>
<tr>
<td>Pattern Mining in Data Streams</td>
<td>Dirk Börner</td>
<td>OUN</td>
<td></td>
</tr>
<tr>
<td>Ambient Learning Displays</td>
<td>Eelco den Heijer</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Autonomous Evolutionary Art</td>
<td>Joop de Jong</td>
<td>TUD</td>
<td></td>
</tr>
<tr>
<td>A Method for Enterprise Ontology based Design of Enterprise Information Systems</td>
<td>Pim Nijssen</td>
<td>UM</td>
<td></td>
</tr>
<tr>
<td>Monte-Carlo Tree Search for Multi-Player Games</td>
<td>Jochem Liem</td>
<td>UVA</td>
<td></td>
</tr>
<tr>
<td>Supporting the Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning</td>
<td>Jurriaan van Reijsen</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Knowledge Perspectives on Advancing Dynamic Capability</td>
<td>Damian Tamburri</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Supporting Networked Software Development</td>
<td>Arya Adriansyah</td>
<td>TUE</td>
<td></td>
</tr>
<tr>
<td>Aligning Observed and Modeled Behavior</td>
<td>Samur Araujo</td>
<td>TUD</td>
<td></td>
</tr>
<tr>
<td>Data Integration over Distributed and Heterogeneous Data Endpoints</td>
<td>Philip Jackson</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language</td>
<td>Ivan Salvador Razo Zapata</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Service Value Networks</td>
<td>Janneke van der Zwaan</td>
<td>TUD</td>
<td></td>
</tr>
<tr>
<td>An Empathic Virtual Buddy for Social Support</td>
<td>Willem van Willigen</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Look Ma, No Hands: Aspects of Autonomous Vehicle Control</td>
<td>Arlette van Wissen</td>
<td>VU</td>
<td></td>
</tr>
<tr>
<td>Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains</td>
<td>Yangyang Shi</td>
<td>TUD</td>
<td></td>
</tr>
<tr>
<td>Language Models With Meta-information</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2014

<table>
<thead>
<tr>
<th>Title</th>
<th>Author</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies in Learning Monotone Models from Data</td>
<td>Nicola Barile</td>
<td>UU</td>
</tr>
<tr>
<td>Combining System Dynamics with a Domain Modeling Method</td>
<td>Fiona Tulayano</td>
<td>RUN</td>
</tr>
<tr>
<td>Information Retrieval for Children: Search Behavior and Solutions</td>
<td>Sergio Raul Duarte Torres</td>
<td>UT</td>
</tr>
<tr>
<td>Websites for children: search strategies and interface design - Three studies on children's search performance and evaluation</td>
<td>Hanna Jochmann-Mannak</td>
<td>UT</td>
</tr>
<tr>
<td>Knowledge Perspectives on Advancing Dynamic Capability</td>
<td>Jurriaan van Reijsen</td>
<td>UU</td>
</tr>
<tr>
<td>Supporting Networked Software Development</td>
<td>Damian Tamburri</td>
<td>VU</td>
</tr>
<tr>
<td>Aligning Observed and Modeled Behavior</td>
<td>Arya Adriansyah</td>
<td>TUE</td>
</tr>
<tr>
<td>Toward Human-Level Artificial Intelligence: Representation and Computation of Meaning in Natural Language</td>
<td>Ivan Salvador Razo Zapata</td>
<td>VU</td>
</tr>
<tr>
<td>Service Value Networks</td>
<td>Janneke van der Zwaan</td>
<td>TUD</td>
</tr>
<tr>
<td>An Empathic Virtual Buddy for Social Support</td>
<td>Willem van Willigen</td>
<td>VU</td>
</tr>
<tr>
<td>Look Ma, No Hands: Aspects of Autonomous Vehicle Control</td>
<td>Arlette van Wissen</td>
<td>VU</td>
</tr>
<tr>
<td>Agent-Based Support for Behavior Change: Models and Applications in Health and Safety Domains</td>
<td>Yangyang Shi</td>
<td>TUD</td>
</tr>
<tr>
<td>Language Models With Meta-information</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2014-15 Natalya Mogles (VU)
Agent-Based Analysis and Support of Human Functioning in Complex Socio-Technical Systems: Applications in Safety and Healthcare

2014-16 Krystyna Milian (VU)
Supporting trial recruitment and design by automatically interpreting eligibility criteria

2014-17 Kathrin Dentler (VU)
Computing healthcare quality indicators automatically: Secondary Use of Patient Data and Semantic Interoperability

2014-18 Mattijs Ghijsen (VU)
Methods and Models for the Design and Study of Dynamic Agent Organizations

2014-19 Vinicius Ramos (TUE)
Adaptive Hypermedia Courses: Qualitative and Quantitative Evaluation and Tool Support

2014-20 Mena Habib (UT)
Named Entity Extraction and Disambiguation for Informal Text: The Missing Link

2014-21 Kassidy Clark (TUD)
Negotiation and Monitoring in Open Environments

2014-22 Marieke Peeters (UU)
Personalized Educational Games - Developing agent-supported scenario-based training

2014-23 Eleftherios Sidireurgos (UvA/CWI)
Space Efficient Indexes for the Big Data Era

2014-24 Davide Coolin (VU)
Trusting Semi-structured Web Data

2014-25 Martijn Lappenschaar (RUN)
New network models for the analysis of disease interaction

2014-26 Tim Baarslag (TUD)
What to Bid and When to Stop

2014-27 Rui Jorge Almeida (EUR)
Conditional Density Models Integrating Fuzzy and Probabilistic Representations of Uncertainty

2014-28 Anna Chmielowiec (VU)
Decentralized k-Clique Matching

2014-29 Jaap Kabbedijk (UU)
Variability in Multi-Tenant Enterprise Software

2014-30 Peter de Cock (UvT)
Anticipating Criminal Behaviour

2014-31 Leo van Moergestel (UU)
Agent Technology in Agile Multiparallel Manufacturing and Product Support

2014-32 Naser Ayat (UvA)
On Entity Resolution in Probabilistic Data

2014-33 Tesfa Tegegne (RUN)
Service Discovery in eHealth

2014-34 Christina Manteli (VU)
The Effect of Governance in Global Software Development: Analyzing Transactive Memory Systems.

2014-35 Joost van Oijen (UU)
Cognitive Agents in Virtual Worlds: A Middleware Design Approach

2014-36 Joos Buijs (TUE)

2014-37 Maral Dadvar (UT)
Experts and Machines United Against Cyberbullying

2014-38 Danny Plass-Oude Bos (UT)
Making brain-computer interfaces better: improving usability through post-processing.

2014-39 Jasmina Maric (UvT)
Web Communities, Immigration, and Social Capital

2014-40 Walter Omona (RUN)
A Framework for Knowledge Management Using ICT in Higher Education.

2014-41 Frederic Hogenboom (EUR)
Automated Detection of Financial Events in News Text

2014-42 Carsten Eijckhof (CWI/TUD)
Contextual Multidimensional Relevance Models

2014-43 Kevin Vlaanderen (UU)
Supporting Process Improvement using Method Increments

2014-44 Paulien Meesters (UvT)
Intelligent Blauw. Met als ondertitel: Intelligence-gestuurde politiezie in gebiedsgebonden eenheden.

2014-45 Birgit Schmitz (OUN)
Mobile Games for Learning: A Pattern-Based Approach

2014-46 Ke Tao (TUD)
### 2015

<table>
<thead>
<tr>
<th>Year</th>
<th>Title</th>
<th>Author/Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-01</td>
<td>Niels Netten (UvA) Machine Learning for Relevance of Information in Crisis Response</td>
<td></td>
</tr>
<tr>
<td>2015-02</td>
<td>Faiza Bukhsh (UvT) Smart auditing: Innovative Compliance Checking in Customs Controls</td>
<td></td>
</tr>
<tr>
<td>2015-03</td>
<td>Twan van Laarhoven (RUN) Machine learning for network data</td>
<td></td>
</tr>
<tr>
<td>2015-04</td>
<td>Howard Spoelstra (OUN) Collaborations in Open Learning Environments</td>
<td></td>
</tr>
<tr>
<td>2015-05</td>
<td>Christoph Bösch(UT) Cryptographically Enforced Search Pattern Hiding</td>
<td></td>
</tr>
<tr>
<td>2015-06</td>
<td>Farideh Heidari (TUD) Business Process Quality Computation - Computing Non-Functional Requirements to Improve Business Processes</td>
<td></td>
</tr>
<tr>
<td>2015-07</td>
<td>Maria-Hendrike Peetz (UvA) Time-Aware Online Reputation Analysis</td>
<td></td>
</tr>
<tr>
<td>2015-08</td>
<td>Jie Jiang (TUD) Organizational Compliance: An agent-based model for designing and evaluating organizational interactions</td>
<td></td>
</tr>
<tr>
<td>2015-09</td>
<td>Randy Klaassen (UT) HCI Perspectives on Behavior Change Support Systems</td>
<td></td>
</tr>
<tr>
<td>2015-10</td>
<td>Henry Hermans (OUN) OpenUL: design of an integrated system to support lifelong learning</td>
<td></td>
</tr>
<tr>
<td>2015-11</td>
<td>Yongming Luo (TUE) Designing algorithms for big graph datasets: A study of computing bisimulation and joins</td>
<td></td>
</tr>
<tr>
<td>2015-12</td>
<td>Julie M. Birkholz (VU) Modi Operandi of Social Network Dynamics: The Effect of Context on Scientific Collaboration Networks</td>
<td></td>
</tr>
<tr>
<td>2015-13</td>
<td>Giuseppe Procaccianti(VU) Energy-Efficient Software</td>
<td></td>
</tr>
<tr>
<td>2015-14</td>
<td>Bart van Straalen (UT) A cognitive approach to modeling bad news conversations</td>
<td></td>
</tr>
<tr>
<td>2015-15</td>
<td>Klaas Andries de Graaf (VU) Ontology-based Software Architecture Documentation</td>
<td></td>
</tr>
<tr>
<td>2015-16</td>
<td>Changyuan Wei (UT) Cognitive Coordination for Cooperative Multi-Robot Teamwork</td>
<td></td>
</tr>
<tr>
<td>2015-18</td>
<td>Holger Pirk (CWI) Waste Not, Want Not! - Managing Relational Data in Asymmetric Memories</td>
<td></td>
</tr>
<tr>
<td>2015-19</td>
<td>Bernardo Tabuenca (OUN) Ubiquitous Technology for Lifelong Learners</td>
<td></td>
</tr>
<tr>
<td>2015-20</td>
<td>Lois Vanhee (UU) Using Culture and Values to Support Flexible Coordination</td>
<td></td>
</tr>
<tr>
<td>2015-21</td>
<td>Sibren Fetter (OUN) Using Peer-Support to Expand and Stabilize Online Learning</td>
<td></td>
</tr>
<tr>
<td>2015-22</td>
<td>Zhemin Zhu (UT) Co-occurrence Rate Networks - Towards separate training for undirected graphical models</td>
<td></td>
</tr>
<tr>
<td>2015-23</td>
<td>Luit Gazendam (VU) Cataloguer Support in Cultural Heritage</td>
<td></td>
</tr>
<tr>
<td>2015-25</td>
<td>Steven Woudenberg (UU) Bayesian Tools for Early Disease Detection</td>
<td></td>
</tr>
<tr>
<td>2015-26</td>
<td>Alexander Hogenboom (EUR) Sentiment Analysis of Text Guided by Semantics and Structure</td>
<td></td>
</tr>
<tr>
<td>2015-27</td>
<td>Sándor Héman (CWI)</td>
<td></td>
</tr>
</tbody>
</table>
SIKS Dissertation list

Updating compressed column-stores
Janet Bagorogoza (TiU)
Knowledge Management and High Performance; The Uganda Financial Institutions Model for HPO
2015-29 Hendrik Baier (UM)
Monte-Carlo Tree Search Enhancements for One-Player and Two-Player Domains
2015-30 Kiavash Bahreini (OUN)
Real-time Multimodal Emotion Recognition in E-Learning
2015-31 Yakup Koç (TUD)
On Robustness of Power Grids
2015-32 Jerome Gard (UL)
Corporate Venture Management in SMEs
2015-33 Frederik Schadd (UM)
Ontology Mapping with Auxiliary Resources
Summary

Daily life is full of location-related decisions: where to go on vacation, which house to buy, which job to apply for, etc. These decisions are not only influenced by the characteristics of this holiday home, house, or company, but also by the region it is located in. What is the distance to the beach, the nearest train station, or the schools and kindergartens? In this thesis, that is inspired by the case of an online holiday home broker, we introduce the geosocial recommender system GeoSoRS: a system that supports people in their decision-making process for location-bound objects, such as holiday homes or real estate, using public data only, to keep the threshold for using GeoSoRS to a minimum.

Current location-based recommender systems are focused on the recommendation of a single point-of-interest (POI), based on the characteristics of that POI only. In this thesis, we combine this information about a region in its geoprofile: a description of those characteristics of a region that are relevant to the decision to be made. We try to find the match between users and location-bound objects through a user profile and a geoprofile respectively, and look at the shared interests that users have and regions can satisfy.

We find this match by answering four main research questions. First of all, we present a software architecture suitable for the combination of web content, social media data, user-generated content (UGC) and several other sources that provide useful information for recommendation selection.

In the second, and largest, part of the thesis, we detect which places are visited by which people through a three-step process: (1) POI collection from the web, (2) estimation of their shape and size, called their polygon-of-interest, using public data only, and (3) matching such two-dimensional polygons-of-interest with user trajectories to detect true visits.

Thirdly, we find a way to assess the quality of UGC, based on trajectory characteristics of their creators. We introduce a method for human pattern recognition in trajectory data, and show how the outcome of this pattern detection can be used for the creation of UGC quality prediction models.

With the fourth and final research question, we focus on knowledge-
based recommendations, combining user interests with the interests that can be satisfied at certain locations. We combine social media data, public generic knowledge-bases and tagged item sets to recommend items from multiple domains, among which holiday homes.
Samenvatting

Het dagelijks leven zit vol locatie-gerelateerde beslissingen: waar op vakantie te gaan, welk huis te kopen, op welke baan te solliciteren, etc. Deze beslissingen worden niet alleen beïnvloed door de karakteristieken van dit vakantiehuisje, huis of bedrijf, maar ook door de regio waarin deze zich bevindt. Wat is de afstand tot het strand, het dichtstbijzijnde treinstation, of de kleuterscholen? In dit proefschrift, dat is geïnspireerd door de casus van een online vakantiehuisjesportaal, introduceren wij het geosociaal aanbevelingssysteem GeoSoRS: een systeem dat mensen ondersteund in het maken van beslissingen omtrent locatie-gebonden objecten, zoals vakantiehuisjes of vastgoed, op basis van enkel publiek beschikbare data, om het gebruik van GeoSoRS zo laagdrempelig mogelijk te houden.

Bestaande locatie-gebaseerde aanbevelingssystemen zijn erop gericht om een enkel point-of-interest (POI) aan te bevelen, op basis van de karakteristieken van dat POI. In dit proefschrift combineren wij de informatie over een regio in haar geoprofiel: een beschrijving van die karakteristieken van een regio die relevant zijn voor de beslissing die gemaakt dient te worden. We zoeken een match tussen gebruikers en locatie-gebaseerde objecten op basis van respectievelijk een gebruikersprofiel en een geoprofiel, en kijken naar de gedeelde interesses die een gebruiker heeft en waar een regio in kan voorzien.

We zoeken deze match door vier hoofdonderzoeksvragen te beantwoorden. Ten eerste presenteren we een software architectuur die geschikt is voor het combineren van informatie van het web, informatie van sociale media in het specifiek, user-generated content (UGC) en meerdere andere bronnen van nuttige informatie voor het selecteren van aanbevelingen.

In het tweede en grootste deel van dit proefschrift, detecteren we welke plaatsen zijn bezocht door mensen op basis van hun GPS geschiedenis door middel van drie stappen: (1) het verzamelen van POIs van het web, (2) het schatten van hun vorm en formaat, ook wel polygon-of-interest genoemd, op basis van enkel publiek beschikbare data, en (3) het matchen van zulke twee-dimensionale polygons-of-interest met de GPS data van gebruikers om POI bezoeken te herkennen.
In het derde deel onderzoeken we de mogelijkheid om de kwaliteit van UGC te schatten, op basis van de karakteristieken van de route en snelheid van de makers. We introduceren een methode voor gedragsherkenning door mensen op basis van trajectgegevens, en laten zien hoe de uitkomst hiervan gebruikt kan worden om modellen voor het voorspellen van UGC kwaliteit te maken.

Met de vierde en laatste onderzoeksvraag focussen we op aanbevelingen op basis van kennis: we combineren de interesses van gebruikers met de interessebehoeften waarin kan worden voorzien op bepaalde locaties. We combineren gegevens van social media, publieke, generieke kennisbanken en geannoteerde itemsets van meerdere domeinen, waaronder vakantiehuisjes.