A Thesis Submitted for the Degree of PhD at the University of Warwick

Permanent WRAP URL:
http://wrap.warwick.ac.uk/91939

Copyright and reuse:
This thesis is made available online and is protected by original copyright.
Please scroll down to view the document itself.
Please refer to the repository record for this item for information to help you to cite it.
Our policy information is available from the repository home page.

For more information, please contact the WRAP Team at: wrap@warwick.ac.uk

warwick.ac.uk/lib-publications

Predicting Context and Locations from Geospatial Trajectories

by

Alasdair Thomason

A thesis submitted to The University of Warwick

in partial fulfilment of the requirements

for admission to the degree of

Doctor of Philosophy

Department of Computer Science

The University of Warwick

April 2017

Abstract

Adapting environments to the needs and preferences of their inhabitants is becoming increasingly important as the world population continues to grow. One way in which this can be achieved is through the provision of timely information, as well as through the personalisation of services. Providing personalisation in this way requires an understanding of both the historical and future actions of individuals. Using geospatial trajectories collected from personal location-aware hardware, e.g. smartphones, as a basis, this thesis explores the extent to which we can leverage the latent knowledge in such trajectories to understand the historic and future behaviours of individuals.

In this thesis, several machine learning tools for the task are presented, including the development of a novel clustering algorithm that can identify locations where people spend their time while disregarding noise. The knowledge exposed by such a system is then enhanced with a procedure for identifying geographic features that the person was interacting with, providing information on what the user may have been doing at that time. Interactions with these features are subsequently used as a basis for understanding user actions through a new contextual clustering approach that identifies periods of time where the user may have been performing similar activities or have had similar goals.

Combined, the presented techniques provide a basis for learning about the actions of individuals. To further enhance this knowledge, however, the research presented in this thesis concludes with the presentation of a new machine learning model capable of summarising and predicting the future context of individuals where only geospatial trajectories are required to be collected from the user. Throughout this work, the potential benefits offered by geospatial trajectories are explored, with thorough explorations and evaluations of the proposed techniques made alongside comparisons to existing approaches.

Acknowledgements

The work contained in this thesis has been influenced by many people, both academically through guidance and suggestions, but also through support, encouragement and friendship. Without these people, this thesis would not exist.

Firstly, my thanks go to my supervisors, Dr Nathan Griffiths and Dr Victor Sanchez, who have provided unparalleled support and guidance throughout my time as a postgraduate student. It is their feedback and ideas that have guided the work over the past 3.5 years into a coherent thesis and helped me develop many skills along the way, and I am indebted to them for this.

Secondly, I would like to thank my fellow postgraduate students, both past and present, for their endless supply of lunchtime distractions, amusement and discussion. In addition, many other members of the department have helped me through the years, providing advice, resources and assistance: my advisors, Dr Abhir Bhalerao and Dr Mike Joy, the technical team of Richard Cunningham, Adam Hancox, Rod Moore, Roger Packwood and Paul Williamson, and the administration team of Jane Clarke, Ruth Cooper, Sharon Howard, Lynn McLean, Catherine Pillet and Gillian Reeves-Brown along with many others.

Finally, but by no means least importantly, I wish to thank my family and friends outside the department: my parents for providing me with the education and encouragement to go to university, as well as support throughout my time here and invaluable proof-reading services; my siblings, Elly and Dave, for providing much needed distraction from work; Christian, Danielle, and Louise, amongst others, for support and friendship whenever it was needed.

Although I will not list their names to protect their anonymity, I am also extremely grateful to the individuals who provided me with data for this work by allowing me to track their movements through their smartphones. The data they provided became the basis for much of the work contained in this thesis.

Declarations

The research presented in this thesis is the work of the author and has not been submitted for consideration at another institution. The work was funded by an EPSRC Doctoral Training Partnership Studentship, and parts of this thesis have been published in the following journals, collections and conference proceedings:

- Thomason, Alasdair; Griffiths, Nathan; and Leeke, Matthew. 2015a. Extracting Meaningful User Locations from Temporally Annotated Geospatial Data. In *Internet of Things: IoT Infrastructures*, volume 151 of *LNICST*, pages 84–90. Springer. doi: 10.1007/978-3-319-19743-2_13

  This paper presents the initial form of the Gradient-based Visit Extractor (GVE) algorithm, which forms the basis for Chapter 4.

- Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2015b. Parameter Optimisation for Location Extraction and Prediction Applications. In *Proceedings of the 2015 IEEE International Conference on Pervasive Intelligence and Computing*, pages 2173–2180, Liverpool. doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.322

  An exploration of automatic parameter optimisation for extracting and predicting locations over geospatial trajectories, included in Chapter 4, Section 4.5

- Thomason, Alasdair; Leeke, Matthew; and Griffiths, Nathan. 2015c. Understanding the Impact of Data Sparsity and Duration for Location Prediction Applications. In *Internet of Things: IoT Infrastructures*, volume 151 of *LNICST*, pages 192–197. Springer. doi: 10.1007/978-3-319-19743-2_29

This paper presents an investigation into understanding how properties of geospatial data impact predictions made over the same data. Although not part of the core focus of this thesis, the results are briefly mentioned in Chapter 4, Section 4.5.2.

- Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016a. Identifying Locations from Geospatial Trajectories. *Journal of Computer and System Sciences*, 82(4):566–581. doi: 10.1016/j.jcss.2015.10.005

  An extension of [Thomason et al., 2015a], that presents an expanded Gradient-based Visit Extractor (GVE) algorithm and significantly more in-depth evaluation, again forming the basis for Chapter 4.

- Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016b. Context Trees: Augmenting Geospatial Trajectories with Context. *ACM Transactions on Information Systems*, 35(2):14:1–14:37. doi: 10.1145/2978578

  This work presents and evaluates the Context Tree data structure. Part of the generation procedure forms the basis for Chapter 5, with the remainder of the work being included in Chapter 6.

- Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016c. Predicting Interactions and Contexts with Context Trees. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 46:1–46:4, San Francisco. doi: 10.1145/2996913.2996993

  This paper presents the Predictive Context Tree (PCT) model, a hierarchical classifier that predicts future contexts of individuals. The work also presents a foundation for predicting the geographic features a person will interact with, providing the basis for Chapter 7, as well as reinforcing the work in Chapter 5.

Additional parts of this thesis are contained in manuscripts currently under review for publication:

- Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016d. The Predictive Context Tree: Predicting Contexts and Interactions. *arXiv:1610.01381 (pre-print)*

  This work is an extension to [Thomason et al., 2016c], and thus parts of it are included in Chapters 5 and 7.

Implementations of all algorithms and tools referenced in this thesis have been made available at: `github.com/csukai/position`.

Data

Portions of the research in this thesis used the Nokia Mobile Data Challenge (MDC) Dataset made available by Idiap Research Institute, Switzerland and owned by Nokia [Kiukkonen et al., 2010; Laurila et al., 2012], in addition to data collected from members of the Department of Computer Science, University of Warwick.

Abbreviations

ANN	Artificial Neural Network
AUROC	Area Under the Receiver Operating Characteristic Curve
DBN	Dynamic Bayesian Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
GLONASS	Global Navigation Satellite System
GPS	Global Positioning System
GVE	Gradient-based Visit Extractor
HCD	Hybrid Contextual Distance Metric
HMM	Hidden Markov Model
LUI	Land Usage Identification Procedure
MAE	Mean Absolute Error
MDC	Nokia Mobile Data Challenge Dataset
OSM	OpenStreetMap
PCA	Principal Component Analysis
PCT	Predictive Context Tree
STA	Spatio-Temporal Activity
SVM	Support Vector Machine

Notation

T	Geospatial trajectory: $T = \{p(1),\ p(2),\ p(3),\ ...,\ p(n)$
p	Trajectory point: $p(i) = (x(i),\ y(i),\ t(i),\ \sigma(i))$
$x(i)$	The x coordinate of point i, typically degrees of longitude
$y(i)$	The y coordinate of point i, typically degrees of latitude
$t(i)$	The time component of point i
$\sigma(i)$	The accuracy value of point i, typically maximum likely deviation measured in metres
V	Set of visits: $V = \{v(1),\ v(2),\ v(3),\ ...,\ v(n)\}$
v	Individual visit: $v(i) = (p(i),\ t(i),\ d(i))$; a period of low mobility extracted solely from geospatial trajectories
$\Phi(i)$	The position of visit i
$d(i)$	The duration of visit i
l	Significant location: a cluster of *visits* based on geographical proximity.
f	Geographic feature: a physical entity in the world that has some purpose, e.g. a building, road, public amenity
e	Element: a representation of a geographical feature from a dataset
i	Interaction: a period of time spent interacting with, or within, an element

Contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1
Introduction

Adapting environments to the needs and preferences of their inhabitants is becoming increasingly important as the world population continues to grow. With the majority of people now living in urban environments[1], the provision of smart services and utilities has an unprecedented ability to improve the lives of individuals and societies. This can be realised through the provision of timely and useful information, control of automated systems, and even utilities and transit management at city-scale. In order to provide personalised services, we first require the ability to model and understand the behaviours, interactions and patterns of city inhabitants. One way in which this can be achieved is to collect vast amounts of data from individuals, or the devices they carry. However, the collection of such data would typically be invasive and require additional sensor devices to be carried. Instead, we focus our work on geospatial trajectories that can be collected from smartphones, routinely carried by the majority of the population[2].

1.1 Understanding People from Data

Understanding and modelling the behaviour of individuals can be performed in various ways, from monitoring how people interact with their devices [LiKamWa et al., 2011, 2013; Shye et al., 2010; Wang et al., 2014] and the internet [Alhindi et al., 2015; Ashman et al., 2009; Gossen et al., 2013; Popescu, 2010; Steichen et al., 2012; Zhang, 2013], through to analysing footage from video cameras [Brax, 2008; Janoos et al., 2007; Kim et al., 2010; Sillito and Fisher, 2008], or

[1] `data.worldbank.org/indicator/SP.URB.TOTL.IN.ZS`
[2] `services.google.com/fh/files/blogs/our_mobile_planet_us_en.pdf`

data from heart-rate monitors and other low-level sensors [Choudhury et al., 2008; Lee and Mase, 2002; Lester et al., 2005; Morris and Trivedi, 2011; Pirttikangas et al., 2006; Ravi et al., 2005]. In addition to these sources, online profiling has increasingly been used to understand the preferences of individuals, however, many activities are conducted off-line and therefore require additional data in order to characterise. Such data could come from video and other low-level sensors, which may be able to identify physical activities the user conducts, but such data is not always available, and so would cover only limited parts of a person's day.

Aiming for more continuous data collection, in a manner that does not significantly inconvenience the user, we instead focus our attention on geospatial trajectories. These trajectories are sequences of data points that link an entity (e.g. a person, device, vehicle, etc.) to a specific geographic location at a specific time, and can be collected from any manner of sources. Many devices are now location-aware, being capable of sensing their current location. Such devices include smartphones, in-car navigation devices, and even watches. The continuous collection and use of geospatial trajectories is becoming increasingly possible, with the goal of better personalising the services these devices offer to their users. Furthermore, trajectories can also be collected without any action required of the user though cell tower connections (i.e. by monitoring the towers in range of a cellular device), or through credit card usage, amongst many other methods. These are all minimally invasive to the user, but provide information on which to better learn and understand the person to whom the device belongs.

1.2 Geospatial Trajectories

Regardless of their source or how they are collected, geospatial trajectories are sequences of data points associated with a person or other entity. Each data point relates that entities' location to a specific time, often with an *accuracy* value that represents the uncertainty in the location measurement:

$$T = \{p(1),\ p(2),\ p(3),\ ...,\ p(n)\} \quad p(i) = (x(i),\ y(i),\ t(i),\ \sigma(i)) \qquad (1.1)$$

Where T is a geospatial trajectory consisting of n points. $p(i)$ is the ith trajectory point comprising of location, $x(i)$ and $y(i)$, time $t(i)$ and accuracy $\sigma(i)$, representing the maximum likely deviation from the recorded location, usually measured in metres. While it is not strictly necessary for the location to be recorded as latitude and longitude, this is the most convenient way to represent a physical location, and so we assume that all geospatial trajectories are recorded in this manner. A vast amount of latent knowledge is present in such trajectories; it is the task of the remainder of this thesis to discuss existing approaches of leveraging such knowledge, as well as presenting new methods.

1.3 Problem Statement and Contributions

This thesis aims to explore the potential offered by geospatial trajectories to the task of understanding individuals through machine learning. Specifically, the problem statement is: **Can a technique be developed to predict the future actions, in terms of location and context, of individuals that makes use of data available after collection, but requires only geospatial trajectories to be collected from the individuals themselves?** While exploring this problem, the following contributions are made by this thesis:

1. **Improving on current algorithms for identifying periods of low mobility (i.e. when little motion is present) in geospatial trajectories for the purpose of identifying locations meaningful to the individual.**
   The **Gradient-based Visit Extractor (GVE)** is an algorithm for identifying periods of low mobility from geospatial trajectories, for the purpose of identifying places where people, or other entities, have spent time. The

algorithm is an improvement on existing techniques as it is more resilient to noise and places fewer limitations on the extracted time periods. This algorithm is combined with an analysis of the properties of the extracted visits and the impact these properties have on a sample application, that of location prediction. Such an evaluation is lacking in previous work. Furthermore, additional techniques to aid in visit extraction and prediction are proposed, including a method for automatic parameter optimisation for extraction and prediction.

2. **Developing a technique for the identification of geographic features with which an entity interacts (e.g. specific buildings).**
   While extracting locations from geospatial trajectories produce arbitrary shaped clusters, we propose the **Land Usage Identification (LUI)** procedure, as a method for augmenting trajectories with information on geographic features extracted from a dataset, referred to as *land-usage* data. These augmented trajectories are then subjected to a filtering procedure to identify geographic features with which an individual, or other entity, interacted, providing a basis for understanding the type of places at which a person spends their time.

3. **Establishing a data structure for identifying and summarising contexts from augmented geospatial trajectories to identify periods of time with similar goals, desires and intentions.**
   The **Context Tree** is a hierarchical data structure with a generation procedure that identifies contexts based on the semantics of elements encountered and properties of the interaction with these elements. The Context Tree itself provides a summary of the contexts an individual has been immersed within.

4. **Evaluating the data structure as a predictive model for forecasting the future contexts and location interactions of individuals.**
   Based on the Context Tree, the **Predictive Context Tree (PCT)** is

a hierarchical classifier which is capable of predicting the future location of interactions of individuals, achieving competitive accuracies when compared with existing techniques. In addition to this, the PCT is able to predict the *context* in which a person will be immersed to a high degree of accuracy, offering a platform on which to understand the future actions of people, and thus provide useful and personalised services.

1.4 Code and Algorithms

Throughout this thesis, several tools, techniques and algorithms are developed and applied to real-world geospatial trajectories. In order to encourage the use of these techniques, the code used to generate results for this thesis, including concrete implementations for each contribution, has been released under the *GNU GPL License*[3] and is located at: `github.com/csukai/position`.

1.5 Thesis Structure

The remainder of this thesis is structured as follows:

Chapter 2 presents background knowledge relevant to the task of understanding people from geospatial trajectories. General machine learning topics are discussed, and existing applications of geospatial systems are summarised.

Chapter 3 discusses available datasets for this work, including specifics of the datasets selected for use in this thesis. Additionally, this chapter sets out data collection methodologies for the *Warwick Dataset*, which was collected to overcome drawbacks of existing data available for research purposes.

Chapter 4 explores the potential for improving upon existing visit and location extraction techniques, where the aim is to identify periods of low mobility in geospatial trajectories and cluster these interactions into locations. The chapter primarily presents the *Gradient-based Visit Extractor (GVE)* algorithm

[3]`gnu.org/licenses/gpl-3.0.en.html`

that expands on previous algorithms for this task. The algorithm is then evaluated thoroughly with respect to the properties of the identified interactions and under the sample application of predicting future interactions. This chapter also includes an exploration of the task of optimising parameters for extracting locations using GVE, and provides a discussion on how this can be achieved.

Chapter 5 investigates and proposes a method for identifying geographic features being interacted with from geospatial trajectories augmented with land usage data. This approach produces interactions that are similar in structure to those identified by GVE, but map to single geographic features, thus providing additional information in the form of knowledge about the type and properties of the feature being interacted with. The interactions are evaluated using existing machine learning techniques that have been demonstrated to be applicable to location prediction, with a comparison presented between extracted locations and identified elements.

Chapter 6 uses identified land usage elements as a basis for identifying contexts through hierarchical clustering. The proposed *Context Tree* is a hierarchical data structure and generation procedure that uses semantics and properties of interactions with land usage elements to summarise user contexts. This chapter presents the data structure, metrics, and algorithms for constructing Context Trees, along with a technique for reducing the size of a Context Tree through pruning.

Chapter 7 builds upon the foundation offered by Chapter 6, by presenting a model for predicting the future interactions and contexts of an individual, namely the *Predictive Context Tree (PCT).* This model is then evaluated with respect to both predictions from extracted locations and predictions from identified land usage elements (Chapter 5).

Chapter 8 concludes the thesis with a summary of the contributions made, alongside a discussion of possible future work in this area.

CHAPTER 2

Background and Related Work

Geospatial trajectories form the backbone of many location-aware systems and services, and an increasing amount of research has focused on developing techniques to leverage the knowledge latent in such trajectories. When coupled with the now pervasive nature of location-sensing hardware, such as smartphones, trajectories are an ideal basis for understanding, modelling, and predicting human behaviour.

In existing work, trajectories have been collected from dedicated devices [Ashbrook and Starner, 2003], smartphones [Laurila et al., 2012], WiFi devices [Burbey and Martin, 2008], social networks [Comito et al., 2016], vehicles [Hu et al., 2015], smart buildings [Petzold et al., 2006; Roy et al., 2003], and mobile phone cell networks [Bayir et al., 2009; Farrahi and Gatica-Perez, 2008b, 2009]. Using these trajectories as a basis for knowledge acquisition, research has considered many possible applications, including identifying locations meaningful to individuals [Bamis and Savvides, 2011; Montoliu and Gatica-Perez, 2010], predicting the future location of people [Ashbrook and Starner, 2003; Assam and Seidl, 2013], determining transport methods [Patterson et al., 2003; Zheng et al., 2008a], forecasting the destinations of journeys [Karimi and Liu, 2003; Liao et al., 2007b], and even identifying similarities [Assam and Seidl, 2014; Xiong and Lin, 2012] and anomalies between users [Chen et al., 2011a; Zhang et al., 2011].

The remainder of this chapter explores these topics in depth, along with providing the necessary background information for this thesis.

2.1 Managing and Collecting Data

With various methods of collecting trajectories, ranging from manually writing locations into a travel diary through to automatic logging from a portable device, the properties of data available will vary. For this work, we focus primarily on trajectories collected about individuals, typically from portable devices, but they could also come from vehicular data recorders or from services such as credit card or cellular telephone usage. This section discusses some challenges relating to geospatial data and its collection.

2.1.1 Resource Utilisation

Determining the current location in a portable device can be achieved through many technologies, such as the Global Positioning System (GPS), Global Navigation Satellite System (GLONASS), WiFi positioning, and cell-tower triangulation. The most accurate of these, GPS and GLONASS, have an accuracy of approximately 5-10m in perfect conditions [GMV, 2011; Grimes, 2008], but also requires the most power to determine position, and so existing work has considered balancing the accuracy of collected data with the available resources on the collection device.

In order to balance the requirement for accurate measurement with that of preserving power, research has focused on optimising the data collection process. Kiukkonen et al. [2010] present a state-based machine that transitions between different collection rates and location determination methods based on sensor readings, using WiFi base stations to indicate locations when available, and adjusting the collection rate based on accelerometer readings (i.e. if the user is moving, the collection rate is increased) at other times. Similarly, Chon et al. [2011] present SmartDC, an application that aims to estimate when a user will leave a current area and increase collection around this time, using a Markov predictor, although this system is heavily reliant on the availability of WiFi networks.

2.1.2 Privacy

Another concern when collecting geospatial data is privacy, the preservation of which has long been the subject of research [Ackerman et al., 2001; Beresford and Stajano, 2003; Kaasinen, 2003; Ljungstrand, 2001]. Methods for preserving privacy in location-aware systems include those focusing on the selective obfuscation of originating users. An example of this is the application of 'mix zones' where users can only be identified within certain regions, with user data mixed together at other times [Beresford and Stajano, 2004], or providing the users with fine control over what data can be shared and using intelligent algorithms to determine how privacy would be reduced by sharing the user's location at any time [Boutsis and Kalogeraki, 2016]. Other solutions to the privacy preservation issue focus on enabling privacy-compromising computation to be performed on client devices, eliminating the risk of intercept and location inference [Marmasse and Schmandt, 2000], or the reduction of accuracy of data, for example, the truncation of latitude and longitude values in the Nokia Mobile Data Challenge (MDC) dataset [Laurila et al., 2012], discussed later in Chapter 3. However, since some services necessitate the transmission of location data, are too computationally intensive for client devices, or require the unambiguous identification of a user, the problem of preserving location privacy in location-aware systems persists [Kaasinen, 2003].

Focusing on the reverse of these techniques, that of demonstrating the privacy implications of geospatial data, Rossi et al. [2015a; 2015b] explore techniques for identifying users based on social media check-ins and GPS data. The authors discover that check-ins to certain types of location reduce privacy more than others, and that users have high uniqueness, thereby requiring very little data when attempting to identify a user.

2.1.3 Synthetic Data

With the challenges associated with collecting real-world trajectory data, some work chooses to generate synthetic data for evaluation instead [Giannotti et al., 2007; Karimi and Liu, 2003; Lei et al., 2011; Wolfson and Yin, 2003; Zheng et al., 2010c]. Creating synthetic data does, however, have its own problems. Firstly, the ability for synthetic data to represent the distribution and patterns of real data are limited, without a real-world dataset on which to base a probabilistic model. Additionally, even with such data, creating a model that accurately represents the movement patterns and characteristics of individuals is a significant challenge. Several papers use synthetic data to evaluate location prediction techniques [Bilurkar et al., 2002; Thanh and Phuong, 2007] and visit extraction techniques [Bamis and Savvides, 2010], but fail to demonstrate the applicability of the data they generate.

2.1.4 Ground Truths

For many techniques relating to extracting knowledge from data, collecting a concrete ground truth is infeasible or dependant upon specific applications. Significant location extraction, for example, can extract locations of different sizes and scales and so no single ground truth can exist. Existing literature addresses this by comparing the outputs from such techniques against certain metrics and expectations. For instance, Guidotti et al. [2015] create synthetic trajectories with known properties and devise metrics to compare extracted locations with these properties. Much existing work additionally uses partial ground truths constructed a posteriori from manually analysing the data and cross-referencing with other data sources, such as maps and land usage information [Assam and Seidl, 2014; Comito et al., 2016; Hoh et al., 2010; Lee et al., 2015; Siła-Nowicka et al., 2015; Yan et al., 2013], or analysing video data about the study participants to manually determine activities conducted [Lester et al., 2005]. In addition to this, analysis has been performed in the absence of a ground truth by considering expected properties of the procedure with relation to input parameters [Bao et al., 2011].

2.2 Trajectory Processing

Once geospatial trajectories have been collected, they can be processed and analysed to better understand people and their actions. This section presents several existing methods for processing raw geospatial trajectories to provide a foundation for understanding behaviour.

2.2.1 Reducing Uncertainty

Due to the different methods of collection of trajectories, each data point typically carries some amount of uncertainty. Reducing this uncertainty can be achieved through filtering, outlier detection or tailored approaches such as map-matching that uses known information about the environment to estimate the true location of the entity [Qiu et al., 2013; Zheng, 2015]. Typical filtering approaches include the Kalman filter [Cooper and Durrant-Whyte, 1994; Mohamed and Schwarz, 1999; Zheng and Zhou, 2011], impulse response filter [Ge et al., 2000], particle filter [Giremus et al., 2004; Wang et al., 2007], and moving average filters [Tsai et al., 2004] to smooth out noisy data.

While most useful for vehicular trajectories, map-matching techniques aim to reduce uncertainty by utilising additional information about the world to determine the likely real location the trajectory point was recorded from. This can be achieved by simply mapping the recorded point to the closest road [White et al., 2000], or using more advanced filtering and estimation techniques (e.g. the Kalman filter mentioned earlier) [Goh et al., 2012; Ochieng et al., 2003; Pink and Hummel, 2008; Quddus et al., 2003].

2.2.2 Change-point Detection

Change-point detection can be applied to trajectories to identify the point at which significant change occurs with the goal of partitioning the trajectory into subtrajectories. Depending on the goal of the process, the criteria for selecting change-points will vary, but typically includes monitoring for rapid changes in speed, acceleration, or direction. Subtrajectories segmented in this manner have been used for travel method identification, where the goal is to determine what transportation mode (e.g. walking, cycling, driving) was in use for different components of a journey [Liao et al., 2007b; Patterson et al., 2003; Zheng et al., 2008a,b].

2.2.3 Visit Extraction

Visits, also referred to as *stops* or *stays*, are periods of a trajectory where the entity is likely to have remained in a single location, for example a shop or house for trajectories associated with individuals [Ashbrook and Starner, 2003], or a parking garage or traffic queue for trajectories associated with vehicles [Yang et al., 2013]. The identification of these visits enables applications to reason about behaviour as a sequence of interactions with the environment [Andrienko et al., 2011; Ashbrook and Starner, 2002, 2003; Bamis and Savvides, 2011; Montoliu and Gatica-Perez, 2010]. After such interactions have been identified, we are left with a sequence of visits performed by the entity:

$$V = \{v(1),\ v(2),\ v(3),\ ...,\ v(n)\} \quad v(i) = (\phi(i),\ t(i),\ d(i)) \tag{2.1}$$

Where V is a set of visits, with $v(i)$ being an an individual visit associated with a position, time and duration ($\phi(i)$, $t(i)$, $d(i)$ respectively). For some applications, the visits themselves can be ignored and only the periods of time between them are considered. This may be useful in applications such as exercise trackers where stationary periods are not of interest.

One of the earliest visit extraction techniques is proposed in an investigation conducted by Ashbrook and Starner [2002; 2003] into identifying locations meaningful to a user. From the collected data, Ashbrook and Starner observed that the data loggers used did not function well indoors, as a GPS signal was rarely available, and therefore treated periods of missing data as visits. This approach is limited in that it assumes that all missing data is caused by a visit, and visits cannot occur when data was collected. Indeed, the authors note that the data logging devices were prone to run out of battery power, also causing a lack of data. Building on this work, but assuming a constant flow of data, even when indoors, algorithms have been proposed that aim to identify periods of low mobility from within trajectories. Relying on time and distance thresholds, such algorithms typically operate by identifying subtrajectories that contain points such that the subtrajectory, or visit, is smaller than a specified radius (or, sometimes, that no consecutive points can be greater than a specified distance apart) and the duration of the subtrajectory exceeds some threshold [Andrienko et al., 2011, 2013; Hariharan and Toyama, 2004; Kang et al., 2004; Li et al., 2008; Zheng et al., 2010b, 2009; Zhou et al., 2014]. Montoliu and Gatica-Perez [2010] extend this technique, by adding an additional constraint that the time between consecutive data points in the same visit must be bounded, with the aim of preventing periods of missing data from being contained within a visit. If data became unavailable at one time, and became available at a nearby coordinate some time later, it is not possible to state with certainty that the user remained stationary for the missing period. Another approach considered for visit extraction makes use of the speed or velocity of the user, where low speeds are considered indicative of a visit occurring [Lee et al., 2015; Palma et al., 2008].

Although these techniques may overcome the issues caused by assuming that a loss of GPS signal is equivalent to a visit, they all suffer from a lack of resilience to noise. In the *thresholding* approach, a single noise point outside the visit radius will end a visit prematurely, and when considering velocity, it is likely that noise points will artificially increase the reported velocity of the user, thus also causing visits to be ended.

Aiming to overcome the drawbacks of existing approaches, by assuming noise in the dataset, Bamis and Savvides [2010] present the Spatio-Temporal Activity (STA) extraction algorithm. While the authors were specifically motivated by identifying activities that repeat in cycles through extraction and clustering, the first step of the algorithm, STA extraction, uses a definition of an activity that is identical to our definition of a visit, and thus performs visit extraction. The algorithm is similar to existing approaches in that it iterates over the trajectory points, but uses a weighted averaging filter over the spatial component to reduce the impact of noise before considering an activity to have ended. This technique, however, does have several drawbacks and assumptions relating to the data, for example, requiring evenly time-sliced data and a full data buffer before consideration of a visit can occur, consequently imposing a minimum bound on visit duration.

The topic of visit extraction is considered again later in Chapter 4, where an algorithm is proposed that aims to overcome the drawbacks of the approaches identified here. It is also considered in Chapter 5, where a novel approach to identifying land usage elements interacted with by users is presented, designed to replace traditional visit extraction for some domains.

2.3 Significant Locations

Significant, or *meaningful*, locations form the backbone of many geospatial services as they identify locations that have some meaning to the user. These applications include predicting future visits to locations [Ashbrook and Starner, 2002, 2003; Fukano et al., 2013; Wang and Prabhala, 2012], predicting how long a user will stay at a given location [Liu et al., 2013], as well as labelling locations with their likely meaning [Krumm and Rouhana, 2013]. Literature has also considered the problem of predicting locations in which people will meet [Yu et al., 2015b], and providing recommendations of places to visit to users new to a city based on the locations visited by others [Bao et al., 2015; Zheng and Xie, 2010].

2.3.1 Extracting Locations

Section 2.2.3 discusses methods of identifying visits from geospatial trajectories, resulting in a sequence of such visits that each represents a period of time in which a user remained in one place. The majority of existing work in extracting significant locations makes use of these identified visits by clustering them together to determine visits that belong to the same location. This results in a sequence of visits to locations, where unlike sets of trajectories or visits, repeated visits to the same location, $l(i)$, are possible, for example:

$$l(1) \rightarrow l(2) \rightarrow l(1) \rightarrow l(3) \rightarrow l(4) \rightarrow l(2) \rightarrow ... \tag{2.2}$$

Grouping visits into locations has been performed using unsupervised learning techniques such as clustering. Such algorithms are categorised into two main types: *Hierarchical* and *partitional*. The aim of a hierarchical algorithm is to create a tree-like structure of clusters with a single root cluster and different scales of sub-cluster below. Partitional algorithms, in contrast, cluster the entire dataset into discrete partitions at a single scale.

These types are further broken down into two categories of algorithms: *agglomerative* and *divisive*. Agglomerative algorithms start with each point as a singleton cluster and continually perform rounds of merging until a termination condition is met (bottom-up clustering). Divisive clustering, on the other hand, starts with a single cluster containing all points and repeatedly splits the clusters until some termination criterion is met (top-down clustering) [Jain et al., 1999].

K-means are a family of iterative relocation algorithms that perform expectation maximisation to split the dataset into k clusters [MacQueen, 1967]. The algorithms start with a random initial assignment of points to clusters and continually make changes until the associated error ceases to change significantly [Jain et al., 1999]. There are many different algorithms that use the K-means approach, with MacQueen's [1967] algorithm being the most widely used. K-means is differentiated from K-medoid algorithms in that the clusters are described by their centre, while in K-medoid the clusters are described by an existing point in the dataset that is as close as possible to the centre [Kaufman and Rousseeuw, 1987]. This technique has been used for clustering visits into locations in [Ashbrook and Starner, 2002; MacQueen, 1967]. However k-means requires a value for k, the number of clusters, to be known a priori, which is typically not the case. Ashbrook and Starner [2003] provided a technique for selecting a value for k by performing clustering for several values and observing the results of plotting the number of clusters extracted on a graph.

Without needing the number of clusters to be known a priori, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [Ester et al., 1996], a density-based clustering algorithm that determines clusters according to parameters, has also been shown to be effective for visit clustering [Andrienko et al., 2013; Montoliu and Gatica-Perez, 2010]. DBSCAN does not, however, apply a maximum cluster size, instead allowing arbitrarily large clusters providing that a sufficient density of visits exists.

Hierarchical algorithms such as CLARANS [Han, 2002] are also used for unsupervised learning problems such as this, where some similarity cutoff is typically used to select a level of cluster for a specific application. CLARANS is based on K-medoid, and builds upon PAM and CLARA [Kaufman and Rousseeuw, 1987]. Another such hierarchical clustering algorithm is BIRCH [Zhang et al., 1997]. BIRCH is designed to run on large datasets with limited memory available, where not all data is available at the start. It operates by creating a new data structure, a CF-Tree, and summarising clusters by their centroid, radius and diameter, updating leaf nodes as new points are brought into the system.

Montoliu and Gatica-Perez [2010] employ the grid-based clustering algorithm proposed by Zheng et al. [2010b]. This algorithm works by overlaying a grid on extracted visits, where the length of each square is a user-specified parameter. Squares containing more visits than a threshold are merged with all of their neighbours that have not already been assigned to a location, under the constraint of the maximum location size being defined as 3×3 squares. The clustering stops once no unassigned square exists with greater than the threshold number of visits. While this approach constrains the maximum location size, unlike DBSCAN, the shapes of the locations extracted are more regular, and therefore may not represent the shapes of the real-world locations as accurately.

Although having a slightly different aim, namely that of grouping activities that occur at the same place and at similar times of day, Bamis and Savvides propose a clustering algorithm, STA Agglomerator, that clusters based on both time and location [Bamis and Savvides, 2010]. The STA Agglomerator operates by summarising visits as their 3-dimensional bounding box (along the latitude, longitude and time dimensions) and progressively merging visits into clusters based on a similarity function, with weightings given to longitude, latitude and time. With a weighting to the time dimension of zero, this has the effect of performing visit clustering. However, the algorithm has both space and time complexities of $O(n^2)$, far exceeding those of both k-means and DBSCAN.

Location extraction as a topic is revisited in Chapter 4, where locations are clustered using DBSCAN from visits extraction using both existing and new approaches.

Alternative Approaches

In contrast to the approach of splitting a temporally ordered dataset into visits, there are methods of extracting visits and locations using a single algorithm. Literature that uses a single technique has focused on using existing and modified clustering algorithms to identify dense groups of points without first performing visit extraction [Guidotti et al., 2015; Zhou et al., 2014]. This has the drawback of identifying dense groups of points that happen to be together through chance (e.g. along a road travelled frequently but at which the user did not stop). While this may offer utility to certain applications, several applications of extracted locations are only interested in places where the individual spent time, for example, location prediction.

Considering both spatial and temporal proximity, ST-DBSCAN [Birant and Kut, 2007] and DJ-Cluster [Zhou et al., 2007] are density-based clustering algorithms that are designed to cater for spatio-temporal data by extracting clusters that are similar in both space and time. While they overcome the problem of performing only two-dimensional clustering directly to trajectories (that of extracting dense groups of points without considering time), these approaches are more computationally intensive than performing visit extraction followed by clustering, as the number of visits is typically far lower than the number of trajectory points. Furthermore, only locations are identified and not visits through this technique, so it becomes harder to reason about behaviour.

2.3.2 Labelling and Recommending Locations

Once extracted, locations can be used as a basis for location-aware systems and services, but they can also be enriched with additional information and used, for example, to guide individuals around a city or other point of interest. To this end, research has been conducted into extracting locations from visits and using them to offer suggestions to new users who may be visiting an unfamiliar area, achieved by matching the interests of users based on the locations they have visited [Zheng et al., 2010a,b,c]. This idea has been extended by personalising the recommendations even more towards the user, where the type of location a user wishes to visit next is predicted, and a recommendation that fits this category given [Zhao et al., 2015]. Research has also been conducted into automatic labelling of locations by treating the labels as a supervised learning problem, where a dataset is created with manually assigned labels, and classifiers used to label the remaining locations [Andrienko et al., 2011, 2013; Do and Gatica-Perez, 2013; Farrahi and Gatica-Perez, 2009]. Yan et al. [2013] extend this by providing locations with annotations of semantic information extracted from a land usage dataset. In a related area, Gong et al. [2011] use locations to characterise user similarity based on how much time people spend at the same places and, under the assumption that users with similar preferences have similar movement patterns, use historical information from correlated users to predict where others are going to visit in the future.

2.4 Location Prediction

Location prediction was initially studied for the purposes of predicting to which cell tower connections should be handed over while people were moving [Akoush and Sameh, 2007; Bilurkar et al., 2002; Gong et al., 2011]. More recently, with the increased availability of such data, the task has changed to predicting the future location of an individual from geospatial trajectories [Ashbrook and Starner, 2003; Assam and Seidl, 2013; Chon et al., 2012; Hariharan and Toyama, 2004], or rooms in smart buildings [Petzold et al., 2006; Vintan et al., 2004]. This section explores methods to achieve location prediction, and the different goals they have.

Although research has previously been conducted into predicting exact longitude and latitude values for a future time [De Domenico et al., 2013], the vast majority of predictors work by predicting one of a predefined set of coordinate clusters, typically called *Locations*, as discussed in Section 2.3. Predicting the future location of an individual limited to a discrete set of locations has the advantage that the predicted output is typically more meaningful, offering the ability to say when the user will return to somewhere they have been before, or regularly visit (e.g. their home or place of work).

In addition to the different aims of predicting from a set of discrete locations and predicting the exact continuous longitude and latitude values, location prediction can be split into the categories of *next location* and *future location* prediction. Next location prediction takes a sequence of location transitions, for example $l(1) \rightarrow l(2) \rightarrow l(1) \rightarrow l(3)$, or a single *current* location, and aims to predict the next location to be visited in the sequence. These transitions are often extracted using location extraction techniques and represent places where the individual or other entity remained above some threshold amount of time (e.g. 10 minutes). Future location prediction, by contrast, aims to predict the location of an individual or other entity at a given future time by using the future time as an input parameter.

The techniques discussed in this section, specifically next location prediction, and to a lesser extent, future location prediction, are used as sample applications throughout this thesis. Chapter 4 uses the techniques to predict locations to be visited over locations extracted using a new visit extraction technique as a basis. Chapter 5 uses the same techniques, but over interactions with geographic features represented by land usage elements, extracted by a proposed algorithm. Finally, Chapter 7 proposes a new technique for predicting locations and contexts, using the approaches discussed here as a basis.

Predicting locations in this manner is typically treated as a supervised learning problem in existing literature (e.g. [Akoush and Sameh, 2007; Vintan et al., 2004]). Supervised learning problems are a subclass of machine learning where *training* data is provided as a set of input values along with a single output value, often referred to as the *class* value. It is the goal of supervised learning techniques to derive a function that maps from the input values (e.g. sequences of visits to locations) to the correct output (the next or future location to be visited).

2.4.1 Cell-tower Handover

Initial research into next cell prediction (predicting the next cell tower to which a mobile device will connect) started by simply looking at the movement direction of the user (e.g. north-west) and predicting the next cell tower that the user would encounter on this path. More recently, however, historical information about the user's connections has been considered to provide a higher level of predictive accuracy [Lei et al., 2011; Vukovic et al., 2009]. Bilurkar et al. [2002] propose using neural networks to perform next cell prediction. The neural networks in this case are trained through backpropagation with output nodes representing different possible future cell towers, and input nodes representing the current tower connected to and properties such as time of day. The issue with this work is that the authors assume that areas covered by cell towers are square in shape and evenly distributed, with a transition between towers occurring at most once every 20 minutes. In reality, however, the cells are not fixed in shape at all — a stationary device may have several towers to choose from depending on factors such as the weather — and transitions between cells may occur more frequently than the authors' assumption of every 20 minutes.

Also using neural networks for next cell prediction, Akoush and Sameh [2007] use Bayesian inference to learn the weights in the network, arguing that this reduces the complexity of training the model, thereby speeding up the process when compared to backpropogation. The authors conclude that predicting the specific cell to be connected to is challenging, and so opt for predicting blocks of 6 cells, achieving 57% predictive accuracy. Although an improvement over previous work, especially with regards to evaluating the technique, the accuracy of prediction could still be improved. One technique for this was proposed by Gong et al. [2011], who propose using data from multiple users to improve the predictive accuracy for a single user. They achieve this by ranking users against each other based on a *social correlation* metric, identifying users who spend time together, under the assumption that users who are similar will follow similar patterns. Predictions are then determined by selecting the user with the highest social correlation to the one for whom the prediction is requested, achieving 30% accuracy for predicting the specific cell. However, the authors note that in some cases using a simpler 2nd-order Markov predictor performed better than their proposed approach.

2.4.2 Next Location Prediction

In addition to next cell prediction, next location prediction has also been utilised inside smart homes [Helal et al., 2003; Roy et al., 2003] and office buildings [Hazas et al., 2004]. Prediction inside buildings aims to detect the room that a specific person will visit next, typically motivated by the desire to automatically forward phone calls when an employee leaves their office, or providing electronic signs on the doors of offices to indicate where the usual occupant may be or when they may return. Inter-office prediction has been performed using neural networks, achieving accuracies as high as 92% [Vintan et al., 2004]. Continuing from this work, Petzold et al. [2006] provide a comparison of different machine learning techniques and quantify them based on how accurate they are, how quick they are to build, and how long it takes them to learn a new pattern when a routine changes. The authors consider neural networks (multilayer perceptron and Elman network), Dynamic Bayesian Networks (DBNs), Markov models and state predictors. Concluding, they demonstrate that the highest accuracy (82%) was achieved by the state predictor, although it only returned predictions in 74% of cases. Of those techniques that always returned a prediction, Elman networks performed best with 80% accuracy, although they were characterised by the authors as ‘slow’ to learn and relearn. This work is limited in that it does not evaluate different parameters for each approach. In fact, the authors state that they selected parameters that required similar amounts of memory, but were not optimised for accuracy.

As with the neural networks described in Section 2.4.1, these classification techniques, as well as the stochastic Markov model, were trained using certain properties such as time of day and current location as input parameters, with the output class representing which room or location a person will visit next. The models each therefore take an instance representing the *current* conditions, and select a class that represents the next location the user will likely visit. Evaluation occurs by comparing the output class label against the actual location visited next in the dataset.

A major problem with existing works is that of validation. While research has been published that claims accuracies between 80-90% [Noulas et al., 2012; Petzold et al., 2006; Vintan et al., 2004], the numbers reported are heavily dependent upon factors such as the amount and properties of training data used. When dealing with enclosed environments, the possible locations are typically well defined (e.g. rooms in a building) and any evaluation conducted uses these known locations. However, when dealing with unbounded environments, the selection of possible locations plays an extremely important part in providing meaning to the evaluation conducted. While the extraction of locations is common [Ashbrook and Starner, 2002, 2003; Kang et al., 2004; Liu et al., 2013; Zhou et al., 2007] as a precursor step to location prediction, no research that employs this technique for location prediction provides a thorough analysis of properties the locations extracted.

Focusing on prediction using extracted locations, Ashbrook and Starner [2002; 2003] use Markov models to model a user's transitions between extracted locations. The approach adopted assigns probabilities to each possible transition, where a change in routine (e.g. the user changing shifts in their job or a student having a new schedule) would require an equal number of transitions to occur in the new routine as in the old before the model would adapt. Additionally, the model only contained transition probabilities based on historical transitions, but did not consider factors such as the time of day. Considering time as well as current location, Wang and Prabhala [2012] propose using a combined model, where transitions are used alone if their confidence is high, and a second periodicity-based model is trained and used when transition confidence is low.

Next location prediction over extracted locations has more recently used Support Vector Machines (SVMs) [Wang and Prabhala, 2012], blockmodels [Fukano et al., 2013], Markov models [Assam and Seidl, 2013; Gong et al., 2011; Hariharan and Toyama, 2004; Mathew et al., 2012], and DBNs [Dash et al., 2015]. In this case, the SVMs are used in a one-vs-all approach, where an SVM is trained for each possible next location and optimised to return an answer to the question of whether this class or any other class is the correct classification. Focusing on these techniques, Assam and Seidl [2013] make use of multiple users' data to return a prediction for a single user, by discovering correlations between user transitions.

2.4.3 Future Location Prediction

Although similar to next location prediction, future location prediction typically takes a set of historical trajectories along with a future time and reasons about where the user will be at that specific time. While many of the same machine learning tools are used for both types of prediction, changes must be made to account for the temporal nature of future location prediction. In strictly next location prediction, interactions can be modelled as transitions between states, where time is not required, or often considered. For future location prediction, time must be accounted for in order to reason about it, achieved using Hidden Markov Models (HMMs) [Qiu et al., 2013], state predictors [Xiong and Lin, 2012], and reachability-based approaches [2013]. Other approaches exist that aim to achieve higher accuracy than using standard machine learning techniques. An example of this is work conducted by Burbey and Martin [2008], who instantiate multiple Markov models of different orders, and select the model that returns a result that uses the most historical information (highest order). Burbey and Martin's approach is also capable of estimating the time that the user will arrive at the location they are going to visit, with demonstrated accuracy as high as 91% when allowing for ±20 minutes.

Finally, Gao *et al.* [2012] investigate the reverse of the standard problem. Instead of taking a future time and asking where the user will be, they take a location and ask when the user will next visit. This is achieved by modelling the probability distribution of a user visiting each location for all possible times of day using Bayesian inference, and selecting the time calculated to be most probable from historical information.

2.4.4 Destination Prediction

In addition to determining the next and future location of a user, specific investigations have been conducted into determining the end point of a journey once it has commenced [Chen et al., 2010; Karimi and Liu, 2003; Liao et al., 2007b]. Although related to the problem of next location prediction, destination prediction differs in that it can also consider properties such as the route taken by the driver at the beginning of a journey. With applications in smart vehicles, this work has received increased focus in recent years and is achieved through classification techniques [Cho, 2016], by observing matching similar starting trajectories from one or more users [Chen et al., 2011b; Trasarti et al., 2015; Xu et al., 2016; Xue et al., 2013] and through statistical models [Alvarez-Garcia et al., 2010; Krumm and Horvitz, 2006].

2.5 Contexts and Activities

Identifying and predicting the locations that people wish to visit goes some way to understanding human behaviour, but it does not consider what the person wished to achieve, only where they were. For this, we turn to exploring identifying *activities* and *contexts*, where activities are low-level actions performed by the user, and contexts represent times when a user had a specific goal or task to achieve. For example, a context could represent periods of time throughout which a person was exercising, but the activity being performed would specifically be *jogging* or *playing football*.

This topic is revisited later in this thesis, in Chapters 6 and 7, where the identification and prediction of contexts respectively are considered.

2.5.1 Identifying Activities

Identifying the activities being performed by individuals has been considered as a hierarchical learning problem that can discover activities at multiple scales from video data [Kim et al., 2010]. There is little distinction between activity extraction and activity labelling in existing work, where a group of sensor readings or part of a video are provided and the task is to classify which activity from a set is being performed. This includes identifying the *current* activity from video [Brand et al., 1997; Messing et al., 2009; Morris and Trivedi, 2011], accelerometers [Choudhury et al., 2008; Lee and Mase, 2002; Ravi et al., 2005], accelerometers and heart rate sensors [Lester et al., 2005; Pirttikangas et al., 2006], accelerometers and GPS devices [Subramanya et al., 2006], and custom sensor networks [Van Kasteren et al., 2008]. Relating specifically to trajectories, the existing work is focused on 2D movement trajectories extracted from video data [Bashir et al., 2006, 2007; Brand et al., 1997]. Once extracted, these trajectories have been split into subtrajectories, typically based on changing velocity, where Principal Component Analysis (PCA) and Markov models have been used to detect the activity being performed [Bashir et al., 2007; Brand et al., 1997]. Geospatial trajectories have been considered as a source of activity identification, but this typically entails identifying periods of time spent at specific locations (e.g. [Huang et al., 2015]), which we consider a different problem and discuss in Section 2.3. Using time and features of the locations, Yu et al. [2015a] propose identifying activities from the types of locations visited, and Liao et al. [2007a] propose a hierarchical activity model for individuals that describe the significant locations that a person visits and the activities performed at each of these locations, where activities are determined by assigning labels to grid cells on a map based on the speed of travel and proximity to transit routes in each cell.

Although more broad than activity identification, labelling of individuals has also been considered by using trajectories to classify students based on the course they study [Farrahi and Gatica-Perez, 2008a]. Expanding further on this, Farrahi *et al.* [2008b] label transitions in data in an attempt to summarise behaviour by identifying users with similar lifestyles. The labels they add take a form similar to 'heading home at 10 p.m.', however it is limited in that it only considers three class labels for locations, namely *Home*, *Work* or *Other*.

2.5.2 Identifying Contexts

Context identification, in contrast, aims to discover periods of time in which a person is likely to have had similar goals or performed similar actions but the process is not necessarily concerned with the specific activity being performed. Identifying contexts has been considered from the locations visited by users, where properties of the interactions are used to determine whether a location is likely to have a single purpose (e.g. a restaurant), or multiple purposes (e.g. a shopping centre with restaurants and shops) [Assam and Seidl, 2014].

Identifying the contexts of the user, rather than the location, has been explored using entropy-based clustering [Bao et al., 2011], and sequence-based approaches that consider the transitions between contexts [Lemlouma and Layaida, 2004]. Utilising contexts, research has also focused on developing architectures and applications that adapt devices based on the current context [Anagnostopoulos et al., 2006; Lemlouma and Layaida, 2004]. Situation and intention awareness are related areas that have a greater focus on developing tools and techniques to aid a person in conducting a particular task to achieve some goal [Howard and Cambria, 2013; Vinciarelli et al., 2015], with specific examples in defence [Howard, 2002] and aviation [Endsley, 1995, 2000].

2.5.3 Predicting Future Contexts

Similarly to location prediction, the task of context prediction has been considered in the literature, where context and location prediction sometimes overlap. Using beacons placed around a smart home to identify different contexts, Seo and Lim [2016] predict the future context of occupants using classification techniques, aiming to identify what the user wishes to do in the house next. Additionally, Assam et al. [2014] proposes using identified contexts of locations as a basis for location prediction. Separating out the context and location prediction stages, Yu et al. [2015a] and Bhyri et al. [2015] employ two-step approaches that first aims to predict the context a user will be in and then aims to identify the specific location that the user will visit to fulfil the context, achieved through classification and statistical techniques. Contexts have also formed the basis for recommender systems, with Le et al. [2015] using the context history of users to predict a bundle of locations that the user may like to visit.

2.6 Applications of Geospatial Systems

In [Musolesi, 2014], the author delves into the possible applications of big mobile data, with a specific focus on data associated with location. The article reveals a wide variety of benefits that such data can, or could, afford to individuals. While many such applications are currently waiting to be realised, many more have concrete procedures and applications in place already, including the prediction of crime locations [Gerber, 2014], the provision of targeted advertisements [Wu et al., 2016], and the automatic detection of residence changes [Matekenya et al., 2015]. While these applications have specific goals, there are many more that aim to instead provide general techniques, such as the grouping of repeating patterns [Cao et al., 2005, 2007; Eagle and Pentland, 2009; Giannotti et al., 2007; Gudmundsson et al., 2004], used as a precursor to user similarity identification [Xiao et al., 2012]. The remainder of this section focuses on other tools and techniques that are based on geospatial trajectories.

2.6.1 Transport

In addition to transport destination prediction, discussed in Section 2.4, other applications of trajectories have been considered with transport and transit in mind. Such applications consider the detection of locations that people park in [Yang et al., 2013], and the provision of intelligent route guidance and traffic anomaly detection [Liu et al., 2011; Yuan et al., 2011].

2.6.2 Trajectory Similarity Identification

Trajectories belonging to entities enable the discovery of similarities between individuals and routes travelled. Existing work has focused on identifying overlapping components of journeys from geospatial trajectories [Cao et al., 2005], a technique used to inform destination prediction [Liao et al., 2007b]. Similarity between users has also been investigated using pattern discovery in routines (i.e. finding users who visit similar sequences of locations) [Assam and Seidl, 2014; Shen and Cheng, 2016; Xiong and Lin, 2012], or common behaviours (e.g. 'user typically leaves the house late on weekends') [Eagle and Pentland, 2009]. With groups of similar users determined, techniques are able to utilise data from multiple people to reason about these groups. Examples that utilise such techniques include the use of historical locations of multiple users to determine travel times using neural networks [Chen et al., 2009]. Focusing on a much larger scale, Oliveira et al. [2016] investigate the similarities and differences of users in different cities, finding that there are many commonalities between the patterns of users, regardless of the city.

Calculating the similarity of trajectories is often performed by measuring the distance between two trajectories or the difference between sequences of location interactions. Metrics to achieve this include looking for the closest points and measuring their distance [Zheng, 2015], using statistical measures such as the Hausdorff distance [Lee et al., 2007], models such as the HMM [Porikli, 2004], and calculating the differences between bounding boxes surrounding two trajectory components [Zheng and Zhou, 2011]. In addition, Zheng et al. [2016] propose a technique that looks at the semantic similarity of locations visited as part of a trajectory, and uses this as a basis for identifying similar trajectories.

On an individual basis, Farrahi et al. [2010] have devised a technique, *bag of multimodal behaviour*, that aims to identify when users are performing activities with other people, a possible precursor step to predicting when people will meet or perform some activity together again in the future.

2.6.3 Anomaly Identification

Existing work has also explored the potential that geospatial trajectories from different sources offer to the task of anomaly detection. This has been achieved using isolation-based outlier detection to determine anomalous subtrajectories from vehicle tracking data [Chen et al., 2011a; Zhang et al., 2011], and statistical measures, such as the Hausdorff distance, to identify general trajectories that differ from an expected pattern [Laxhammar and Falkman, 2011, 2014; Rosen and Medvedev, 2012]. In addition to these approaches, which aim to identify specific anomalies in individual data streams, anomaly detection has been considered on data from multiple sources in the form of hotspot detection. Specifically, with vehicular data, Liu et al. [2010; 2011] propose the *outlier tree* that stores relationships between outliers at different scales into a data structure, and demonstrate its utility to identify heavy traffic along routes modelled as a graph. Hotspot detection techniques have also been demonstrated to apply to crime data for the purpose of identifying high crime areas [Shiode and Shiode, 2014; Shiode et al., 2015], and correlation-based clustering for detecting mass population movement [Liu et al., 2014].

The previous examples of anomaly identification are specific to geospatial data, but there are many general approaches that while not necessarily having been used for this purpose before, may well perform effectively in this context. Previously considered techniques that have shown to be promising include detecting data points that do not fall within expected clusters [Breunig et al., 2000; Chandola et al., 2009; Gogoi et al., 2011; He et al., 2003], classification-based approaches [Abe et al., 2006; Chandola et al., 2009], and distance-based techniques [Angiulli and Fassetti, 2009; Chandola et al., 2009; Gogoi et al., 2011].

2.7 Summary

This chapter has provided a background for the work in the remainder of this thesis, and introduced many concepts that are built upon later. Specifically, Chapter 4 builds upon the visit extraction and clustering techniques discussed in Sections 2.2.3 and 2.3.1 respectively. Chapters 5 and 6 again discuss visit extraction (Section 2.2.3), but also consider identifying contexts, as discussed in Section 2.5. Finally, Chapter 7 makes use of classification techniques, as introduced in Section 2.4, to perform location and context prediction (Sections 2.4 and 2.5.3 respectively). The remaining background material is referenced throughout this thesis and provides a grounding in the area of understanding individuals and entities from geospatial data.

CHAPTER 3
Datasets

This chapter describes the publicly available datasets that are used throughout this thesis, in addition to presenting the *Warwick* dataset. Existing datasets available for research purposes have limitations in the form of both the data collected, for example not covering continuous spans of time or having locations obfuscated for privacy reasons, and also with licences that prevent publication of specifics of the datasets. To overcome these, the Warwick dataset was collected from members of the University of Warwick, and consists of geospatial trajectories recorded by smartphones.

3.1 Geospatial Trajectories

As discussed in Chapter 1, Section 1.2, geospatial trajectories are sequences of data points that relate the location of an individual, or other entity, to a specific time. In order to explore the potential these trajectories offer to understand individuals, we first require the availability of such trajectories for analysis. Several trajectory datasets have been collected by institutions and made available for research. This section presents the available datasets and discusses their differences and limitations, categorised by the type of entity to which the data relates.

3.1.1 Individuals

Trajectory datasets about individuals are perhaps the most common, as they afford insight into the movement patterns of people throughout the world. There are two broad categories of collection methodologies for these trajectories: *active* and *passive*. Actively collected trajectories are those collected using any methodology in which the individual decides to record their location, such as using a device to track a period of exercise (e.g. a bike ride), or sharing their location through social media. Conversely, passive trajectory generation can occur inadvertently and without the user having to perform any specific action. Examples of passive data collection include carrying a cellular-connected mobile phone, where the network provider logs the cellular tower to which the phone is connected, or using a credit card to make purchases, where the credit card provider records the location of the card each time it is used.

Datasets available for research are typically collected using an active methodology, where the participant has control over whether to provide data or not at any given time. In early literature, such data collection occurred using dedicated GPS tracking devices (e.g. [Ashbrook and Starner, 2002]), but more recently the use of smartphones to collect data has become commonplace, with several datasets made available for research:

Reality Mining [Eagle and Pentland, 2005] is a dataset collected in 2004 from Nokia 6600 mobile phones carried by 75 students and faculty members of the MIT Media Laboratory and 25 students of the MIT Sloan Business School. The data was collected over the course of a year, with approximately 450,000 hours of total coverage. The goal of the investigation was to explore real-world behaviour from mobile telephones, and includes information such as time, call logs, Bluetooth devices in proximity, mobile phone cell ID, application usage and charge status of devices belonging to the participants. The inclusion of cell ID provides a mechanism to locate the individual geospatially, although the accuracy of such a location is likely to be poor as the range of a connection between a cellular device and tower can be up to 35km [ETSI, 1996].

GeoLife Trajectories [Zheng et al., 2008a, 2009, 2010c] is a dataset collected by Microsoft Research Asia, which instead of using cellular ID, uses GPS to determine location. The dataset includes data from 172 users, totalling

50,000 hours of coverage, and collected throughout 2007-2012. The goal of the collection was to record the movement patterns of city inhabitants through their smartphones and other location-aware devices. However, the data collection methodology was only interested in periods of time in which the users were moving, and therefore only requested data collection be enabled when the person was in motion, resulting in non-continuous data.

The Yonsei Dataset [Chon and Cha, 2011] is a set of geospatial trajectories collected in 2011 from 8 students at Yonsei University in Seoul over a period of 2 months. Location was determined using a combination of cell-tower triangulation, WiFi fingerprinting and GPS. Specifically, an application, SmartDC, was employed to balance location accuracy and collection frequency against resource utilisation [Chon et al., 2011]. Although some of the data will, therefore, contain high-accuracy locations determined by GPS, other parts of trajectories will have locations determined by lower-accuracy techniques when the system favoured power conservation over accuracy.

The Nokia Mobile Data Challenge (MDC) Dataset [Kiukkonen et al., 2010; Laurila et al., 2012] also uses GPS to determine location, and features data from nearly 200 individuals over the span of 2 years. The full dataset includes social interaction data (e.g. call and SMS data) and phone application usage in addition to geospatial trajectories. This data was collected with the aim of providing continuous coverage of the participants' days, unlike GeoLife. However, in order to protect the anonymity of the participants, detail for the geographic regions around their home or work locations have been removed from the data by truncating latitude and longitude values, artificially reducing the variance of these periods. The 200 users who participated in the study were each given a Nokia N95 smartphone, and the users range from 18 to 62 (mean 29) years of age, with approximately 62% of participants being male.

In addition to data collected from smartphones, social media has also been used as a source of trajectories. Due to their nature, however, these trajectories are extremely sparse. Such datasets include Brightkite [SNAP, 2008] and Gowalla [Cho et al., 2011; SNAP, 2010], along with the Foursquare dataset [Yang et al., 2016].

3.1.2 Other Entities

While most focus on trajectories considers the movement patterns of individuals, trajectory datasets exist for other entities, including animals and vehicles:

Movebank [Movebank, 2017] is an online database of tracking data collected about animals from researchers throughout the world, aiming to archive animal movement data for future use.

T-Drive [Yuan et al., 2010, 2011] is a subset of a dataset collected by Microsoft Research that contains one week of trajectories collected from 10,357 taxis in Beijing in 2008. The data has a collection rate of approximately one point every 3 minutes.

3.1.3 The Warwick Dataset

For the work in this thesis, we are interested in the behaviour of people and therefore limit ourselves to datasets concerning human trajectories, collected using GPS to achieve the highest accuracy. We are also interested in continuous behaviour, which excludes the *GeoLife* dataset, leaving the *Yonsei* and *MDC* Datasets as options to support this research. The Yonsei data is limited in that it only spans 2 months for 8 users, and employs a variety of techniques for determining location. The MDC data, on the other hand, contains a vast amount of data, but has large periods where latitude and longitude are truncated; the impact of this on applications is unclear. Due to these considerations, we opt to employ the MDC dataset, but in order to understand how the truncated periods impact results, we also opt to collect our own data for comparative purposes.

The *Warwick Dataset* was collected from 17 members of the University of Warwick over a period of 6 months, with a methodology that aimed to replicate the MDC collection procedure. The dataset contains 627,983 trajectory points in total. Properties of some of these trajectories are summarised in the following section, in Table 3.1.

3.1.4 Trajectories Selected for Evaluation

For privacy reasons, we are unable to publish the Warwick dataset, and so for the remainder of this thesis, we employ both the Warwick and MDC datasets, discussing the differences between results from each dataset. Using both sources of data for evaluation provides us with the ability to demonstrate the applicability of the techniques proposed in this thesis over real-world datasets, while ensuring reproducibility by performing some of our evaluation on a publicly available dataset. Using our own dataset also provides us with the ability to interview participants in order to validate results where appropriate, a task that is not possible with any existing research dataset.

It is important to note, however, that both of the datasets used are collected from groups of people with similar demographics, although in different countries. The users are students and faculty of two universities, primarily of computer science or related departments. These individuals are therefore likely to have properties in common that make the results similar across both datasets, and while we have attempted to mitigate this by using two different datasets, we are unable to say how well the results contained in this thesis will generalise to other demographics and people with vastly different patterns of life.

Before making use of the datasets, we perform two pre-processing steps:

- With the MDC data, as discussed in Section 3.1.1, the creators opted to truncate the latitude and longitude values around the locations where individuals live, typically down to 2 decimal places, reducing the precision of the location to approximately 1.2km^2, and removing all variance for these time periods. For our analysis we opt to exclude these points, and treat them as missing data. The algorithms that we discuss in this thesis assume variance in data and, therefore, testing them on datasets with this artificial property may produce unexpected results. On the other hand, the expectation of periods of missing data is natural for all geospatial datasets recorded through devices carried by the user, and so no loss of generality should occur by treating such data as missing. As these periods usually relate to times when a user was at home and work, the impact of removing such significant places from the data is unknown, and so we employ the Warwick dataset, which contains coverage of times spent at home and work, as well. Chapter 7 and Appendix D explore the impact of using the MDC data with and without these truncated periods.

- The second pre-processing step is applied to both datasets and ensures that only one trajectory point can occur per time instance. In the case of both the MDC and Warwick datasets, time is measured to the nearest second. This check prevents more than one point from being associated with the same second, where multiple points were recorded by the measuring device erroneously. In cases where the points contain different location recordings, the one with the best accuracy value is selected for storage and the remainder discarded.

In total we select 1.2 million trajectory data points, covering 20,965 hours of human mobility patterns, for our evaluation. These data points come from 20 individuals, 10 from each dataset, selected to ensure a wide variety of trajectory lengths. A trade-off is made between the number of users whose data is analysed and the sparsity of the data. While additional users would be desirable, this work focuses on understanding behaviour at high resolution and thus data from fewer users but with reduced sparsity is prioritised.

Summaries of the users selected can be found in Table 3.1, where the *Coverage* is calculated by counting the number of hours covered by each trajectory,

Table 3.1: Summary of users selected from the Warwick and MDC datasets.

User	**Points**	**Rate**	**Coverage (Hrs)**	**Accuracy (m)**
MDC_5927	93909	1.7	901.7	97.5
MDC_5938	192343	0.9	3427.6	89.2
MDC_5947	108333	1.6	1110.2	85.1
MDC_5948	31438	1.5	350.3	98.7
MDC_5966	90510	2.4	634.8	83.0
MDC_5976	150551	2.4	1067.5	73.1
MDC_5990	49817	1.7	501.0	89.0
MDC_6051	38762	2.0	321.5	99.8
MDC_6104	23448	2.1	184.6	87.4
MDC_6109	117411	1.9	1032.4	100.7
War_08	14471	0.5	456.7	36.9
War_1c	33172	0.4	1483.4	57.4
War_1d	12637	0.3	682.2	34.8
War_24	45199	0.8	906.5	141.2
War_61	41571	0.3	2223.9	112.7
War_6b	24571	0.7	616.6	9.9
War_6c	23993	0.2	1750.2	86.2
War_85	35179	0.5	1133.0	67.4
War_87	12411	0.7	307.2	51.5
War_95	44592	0.4	1873.4	60.0

excluding periods of missing data greater than 1 hour from the totals. In the table, each user is given an identifier where *MDC* and *War* refer to users of the MDC and Warwick datasets respectively. MDC users are assigned their 4-digit identifier from the dataset, while Warwick users are given a randomly generated 2-character identifier. Both of these datasets are employed to evaluate the main contributions of this thesis, and therefore feature in Chapters 4-7.

3.2 Land Usage Data

In addition to geospatial trajectories, the work in Chapters 5-7 also requires data on real-world entities in order to provide additional meaning to the trajectories. This section details some of the available datasets for identifying land usage and individual entities.

LMC2007 [CEH, 2007] is a dataset covering the United Kingdom that has classified the land in the country using satellite images. Although it has coverage of the whole country, the labels applied are broad (e.g. grassland, freshwater, urban) and individual entities, such as buildings, are not identified.

The **OS MasterMap** [Ordnance Survey, 2017] is a highly-detailed map of the United Kingdom that includes all roads, buildings, and other features (e.g. trees, fences, fields) in the country. Unfortunately, this solution is proprietary and not freely available for research.

OpenStreetMap (OSM) [OpenStreetMap, 2017a] is an open-source mapping solution similar to the OS MasterMap, but covering the entire world. Although the data is provided by users, the framework for submitting changes to the map ensures that the features labelled are consistent, and the data includes details on roads, buildings and many other types of feature. Furthermore, each *element* in OSM is associated with a set of *tags* that provide information about the element in question, often including details such as the purpose of buildings, or the speed limit of roads.

Due to its global coverage, and goal of mapping each individual feature, we select a static snapshot of the OSM dataset for this work. This snapshot was taken on 2nd September 2015, a date after the collection of both the MDC and Warwick datasets had been completed. The dataset can be queried through the Overpass API [OpenStreetMap, 2017b]. To reduce the complexity of processing certain elements, the API limits the types of feature that can be extracted by using a *within* query (i.e. when queried with the request "what elements are this point contained within?") to those that have been assigned a name manually in the data. Unfortunately, not everything relevant in the database has been given a name (most houses, for instance, have a 'building' tag of 'residential', but not a specific name). Knowing the user is in a house is extremely useful, even if the house does not have a name, and so to get around this limitation, we modify the API to allow the selection of elements that form closed polygons regardless of whether or not they have an assigned name[1]. Data from OSM is featured as part of the evaluation of the contributions presented in Chapters 5-7.

[1]Details of this modification can be found here: `github.com/csukai/osm`

3.3 Summary

This chapter has presented a discussion of existing trajectory and land usage datasets, along with any limitations they may have. From the available datasets, we select the MDC dataset for evaluating the techniques proposed in this thesis. However, the MDC dataset carries some limitations in the form of reduced accuracy of data, to protect participants' privacy, and licensing terms that prevent publication of maps showing any part of the data. In order to work around these limitations, we have also collected our own data, the Warwick dataset, and we make use of trajectories from 10 users of both the MDC and Warwick datasets for the remainder of the work in this thesis. Additionally, we have selected OSM as a source of data for land usage, as it is freely available and contains details on vast numbers of geographic features, such as buildings, roads, and recreation areas.

CHAPTER 4
Identifying Visits from Geospatial Trajectories

Harnessing the latent knowledge present in geospatial trajectories allows for the potential to revolutionise our understanding of behaviour. One part of achieving this is the identification of periods of time when the user is likely to have remained in the same region, allowing for the identification of repeated visits to the same place. Existing work has made much use of extracted locations (as discussed in Section 2.3), but this is far from a solved problem as these approaches either fail to handle noisy data, or have practical limitations. This chapter presents a new algorithm for the identification of periods of low mobility from geospatial trajectories, the Gradient-based Visit Extractor (GVE) algorithm. We evaluate the algorithm with respect to existing approaches, demonstrating its applicability to the task of visit extraction from noisy data, as well as demonstrating the ability for locations to be extracted from identified visits and for these locations to be used as a basis for prediction.

4.1 Introduction

Visit extraction is an important part of location-aware research as it allows us to reason about the behaviour of individuals by identifying places where they have historically spent their time. Furthermore, the procedures for visit extraction require only geospatial trajectories, which are becoming increasingly available due to the rise in prevalence of location-aware hardware. However, the identification of such visits is not a solved problem as existing approaches are typically fairly primitive and come with many limitations. Sections 2.3.2 and 2.4 introduced the potential applications of identified visits and the locations that can be clustered from them. Such applications include location prediction,

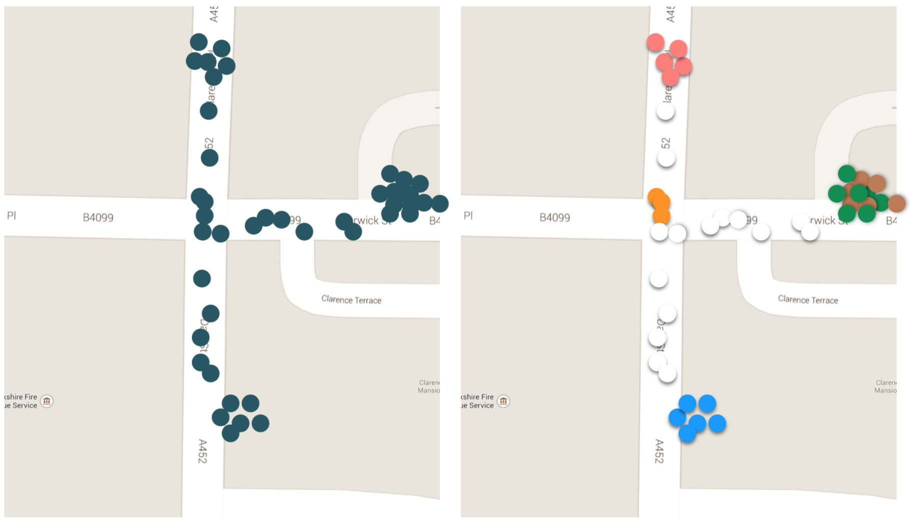


Figure 4.1: Visit extraction is performed over geospatial trajectories (left) to identify periods of low mobility (right).

typically relying on extracted locations to discretise the set of possible predictive outcomes, or context-aware services such as recommender systems or digital assistants that use location to provide a greater level of personalisation.

The remainder of this section discusses the problem of location extraction, with related work summarised in Section 4.2. Section 4.3 presents the Gradient-based Visit Extractor (GVE) algorithm for extracting periods of low mobility from geospatial trajectories. The algorithm is compared to existing approaches in Section 4.3.2, and used as a basis for prediction in Section 4.5. Finally, the chapter concludes with Section 4.6.

4.1.1 Visit and Location Identification

The extraction of locations meaningful to users is achieved by analysing the datapoints found within trajectories and identifying the regions where the user has spent time. Although a variety of techniques have been used in the literature to extract locations from data, they are typically used as a precursor step to performing another activity, such as location prediction, and have not been investigated or evaluated in depth. While it is possible to perform location extraction using a single clustering algorithm, the process is typically performed in two distinct clustering steps [Andrienko et al., 2011, 2013; Ashbrook and Starner, 2002; Bamis and Savvides, 2011; Montoliu and Gatica-Perez, 2010; Zheng et al., 2010b]. The first step, *visit extraction*, is responsible for partitioning a temporally ordered dataset into periods of low mobility, referred to as *visits*, or sometimes *stops* or *stays*. During each visit, the individual or other entity is expected to have remained in one geographic location. Points recorded while moving (i.e. those not contained in a visit) are classed as noise. Clustering the extracted visits by their spatial proximity can then be performed to identify locations repeatedly visited, referred to as *visit clustering*.

Extracting significant locations by first using visit extraction techniques, as depicted in Figure 4.1, has several advantages, namely:

- Visit extraction can be performed in linear time (i.e. $O(n)$), summarising vast portions of the dataset, thus reducing the input size for the clustering step (typically $O(n^2)$).
- By considering their temporal nature, individual data points that occur when an entity is moving are ignored. In traditional clustering, if several points were to be recorded in close proximity on different occasions, for instance along a road, an erroneous location would be identified.
- Extracted visits consist of contiguous points and thus characterise a period of time in which the user remained at the location, providing a basis for modelling a user's time.

The disadvantages of identifying locations through visit extraction relate to the location clusters extracted at the visit clustering step. In order to reduce the complexity of this stage, extracted visits are typically summarised into a single point (e.g. centre of mass or centroid), and consequently the shape of overall locations (i.e. convex hull of the associated points) extracted are not likely to be represented. Depending upon the goal of location clustering, this could be problematic, and is explored as part of this work.

4.1.2 Use Cases

The uses for extracted locations and visits are varied as they provide a foundation for modelling behaviour. However, for this chapter we consider some representative examples as motivation, and refer back to them throughout the chapter, these examples applying equally well to trajectories from various sources. The first use case, referred to as *accurate visits*, considers the visits as a source of context, aiming to accurately summarise periods of time into a single geographic place. Locations can be optionally used here to tie together the visits that occur to the same geographic region, but it is the visits themselves that are of primary interest. An *accurate visit* is therefore a visit that has been accurately identified in terms of space and time. The second use case, *location properties*, is less focused on visits, but rather considers the accurate identification of the properties (i.e. shape and position) of locations. Accurately identified locations are essential to certain services, such as creating geofences, where the visits are of less importance. It is important to note, however, that although the *location properties* use case does not strictly require the accurate extraction of visits, the runtime of visit clustering algorithms is severely impacted as the number of visits increases.

In addition to these use cases, which focus on visit extraction, we employ a sample application, that of location prediction, to demonstrate the utility of the extracted visits and locations. This application is used throughout this thesis, specifically in Chapters 5 and 7, in addition to Section 4.5, as a basis for exploring the utility of different procedures.

4.2 Related Work

In Chapter 2 we provided background and related work for the topics in this chapter, with a recap of the most relevant topics included here for ease of understanding. Specifically, Section 2.2.3 discusses the algorithms used for visit extraction, Section 2.3 discusses clustering together visits to identify locations, and Section 2.4 discusses methods of location prediction. In this chapter we also present a technique for the automatic selection of parameters for location extraction and prediction, where relevant approaches and background reading is discussed in Section 4.2.4.

4.2.1 Visit Extraction

The most common approach for identifying visits uses time and distance thresholds to construct subtrajectories such that each subtrajectory, or visit, is smaller than a specified radius (or, sometimes, that no consecutive points can be greater than a specified distance apart) and the duration of the subtrajectory exceeds some threshold [Andrienko et al., 2011, 2013; Hariharan and Toyama, 2004; Kang et al., 2004; Li et al., 2008; Zheng et al., 2010b, 2009; Zhou et al., 2014]. Montoliu and Gatica-Perez [2010] extend this technique by adding an additional constraint that the time between consecutive data points in the same visit must be bounded, proposed to prevent periods of missing data from being contained within a visit. This approach does not, however, allow for noise in the dataset where a single point recorded outside the radius of a visit would cause a visit to be ended, even if this point was erroneous. Aiming to overcome this issue, by assuming that there may be noise in the dataset, Bamis and Savvides [2010] present the Spatio-Temporal Activity (STA) extraction algorithm. This approach is similar to existing approaches in that it iterates over the trajectory points, but uses a weighted averaging filter to reduce the impact of noise before considering an activity as having ended. However, this technique does bring with it several drawbacks and assumptions relating to the data, for example, requiring evenly time-sliced data and a fixed number of data points before consideration of a visit can occur, consequently imposing a minimum bound on visit duration.

4.2.2 Visit Clustering

Clustering visits into locations has been performed using the k-means algorithm [Ashbrook and Starner, 2002; MacQueen, 1967]. However, k-means requires a value for k, the number of clusters, to be known a priori, which is typically not the case. Although approaches exist for selecting an appropriate value of k [Ashbrook and Starner, 2003], the use of DBSCAN, a density-based clustering algorithm that determines clusters according to parameters, is more common [Andrienko et al., 2013; Montoliu and Gatica-Perez, 2010]. The primary drawback of DBSCAN is that it does not limit the size of clusters, instead allowing arbitrarily large clusters providing a sufficient density of visits exists. While Zheng et al. [2010b] have proposed an approach for location clustering that limits the size of clusters; it produces clusters that are regular in shape, and therefore will likely not be applicable for the *location properties* use case introduced in Section 4.1.2.

4.2.3 Location Prediction

Predicting the locations to be visited by an individual can be split into the categories of *next location* and *future location* prediction, where the former aims to identify where the user will visit *next*, upon leaving their *current* location, and the latter aims to predict where a person will be at a given future time. Predicting the next location of a user has applications in smart transit and advertising, where the possible destination could be automatically selected by a vehicle's navigation equipment, or an offer for a particular shop or restaurant could be provided to entice a user to visit a specific location in a shopping centre. Such prediction has also been considered in cellular networks [Lei et al., 2011; Vukovic et al., 2009] or for rooms in a smart office building [Hazas et al., 2004; Petzold et al., 2006]. Next location prediction over extracted locations is typically performed using Support Vector Machines (SVMs) [Wang and Prabhala, 2012], blockmodels [Fukano et al., 2013], and Markov models [Assam and Seidl, 2013; Gong et al., 2011; Hariharan and Toyama, 2004; Mathew et al., 2012].

4.2.4 Parameter Optimisation

Algorithms for visit and location extraction, as well as location prediction, require the selection of appropriate parameters in order to function effectively. In many cases, this problem is non-trivial as knowing the correct parameters to produce good results is not possible. In order to reduce this burden on the application developer, we also explore the potential for automatic parameter selection through optimisation techniques. Existing research has considered cases in which a metric can be devised to rank two sets of parameters quantitatively, presenting techniques for locating *optimal* values. Locating an optimal or near-optimal solution from an n-dimensional search space can then be achieved using optimisation algorithms. One such example is *hill climbing*, which begins at a random point in the search space and repeatedly moves to adjacent states until it reaches a maxima [Russell and Norvig, 2009]. There is no guarantee, however, that such a maxima would be global as hill climbing is prone to detecting local maxima in non-convex search spaces. To avoid this issue, heuristic-based approaches have been presented, including simulated annealing [Bertsimas and Tsitsiklis, 1993; Kirkpatrick et al., 1983] and evolutionary approaches such as genetic algorithms [Russell and Norvig, 2009] and memetic algorithms [Moscato, 1989]. Given enough time, these algorithms can find the global minima, however, in most scenarios this is not practical and therefore they can be used to find an approximation.

Automated selection of parameters has, additionally, been considered for a few select domains, specialised for each task at hand. This includes optimising the parameters for SVMs using online Gaussian process models [Frohlich and Zell, 2005], genetic algorithms [Saini et al., 2010], and heuristic-based approaches [Rubio et al., 2010]. Also considered for parameter optimisation have been Hidden Markov Models (HMMs), using length modelling [Zimmermann and Bunke, 2002], and neural networks, using genetic algorithms [Cook et al., 2000;

Whitley et al., 1990]. We make use of simulated annealing in this chapter to perform the task of automatic parameter selection for location extraction and prediction.

4.3 The Gradient-based Visit Extractor (GVE)

This section presents the Gradient-based Visit Extractor (GVE) algorithm, which extracts visits from geospatial trajectories and addresses the drawbacks of existing approaches, making it capable of extracting visits from a wider variety of datasets. Sections 2.2.3 and 4.2.1 discussed existing approaches to visit extraction, with the STA extraction algorithm identified as the current state-of-the-art [Bamis and Savvides, 2010]. Although an improvement on previous techniques, this algorithm has several limitations. Firstly, the algorithm assumes that the data is sampled at regular intervals. While some domains have regular and predictable data recording, this is not always the case when aiming to extract visits from data collected from devices carried by users. Indeed, it is likely that conditions such as signal loss or lack of battery power will result in periods of time with varying collection rates, and periods with missing data entirely, thus resulting in non-evenly timesliced data. Secondly, the STA extraction algorithm requires that a buffer of points is filled before consideration of a visit can occur, both imposing a delay on points being considered and specifying a minimum bound on visit duration. The result of this is that the minimum visit length to be extracted must be known a priori and used to select appropriate parameters.

Aiming to overcome these drawbacks, we propose the GVE algorithm, shown as Algorithm 1. The remainder of this section discusses the algorithm, as well as demonstrating its applicability to the task of visit extraction, achieved through an exploration of the parameter space of the algorithm, and analysis of properties of the visits extracted. The Gradient-based Visit Extractor identifies visits from temporally-annotated geospatial trajectories by continually building

Algorithm 1 Gradient-based Visit Extractor (GVE) algorithm.

```
$n_{points}, \alpha, \beta, t_{max}$ // Input parameters
visits ← [ ] // Empty array, to be filled with visits
visit ← [ $p_0$ ] // Array containing the first point in the dataset
buffer ← [ $p_0$ ] // Buffer over which to consider trend of motion

function Process(point)

    buffer.append(point)
    if buffer.length > $n_{points}$ then
        buffer.shift
    end if

    // A visit has ended if there is missing data or the user is moving away
    if TDist(visit.last, point) > $t_{max}$ or MovingAway?(visit, buffer) then
        if TDist(visit.first, visit.last) > 0 then
            visits.append(visit) // If the visit has some duration, store it
        end if
        visit ← [ point ]
        buffer ← [ point ]
    else
        visit.append(point) // If the visit hasn't ended, add the new point
    end if

end function

// Is the gradient of the buffer greater than the threshold?
function MovingAway?(visit, buffer)

    if Gradient(visit, buffer) > Threshold($\alpha$, $\beta$, buffer.length) then
        return True
    else
        return False
    end if

end function
```

a visit until adding another point would cause the recent trend of motion to be consistently away from the visit already extracted.

The buffer over which the trend of motion of the user is considered has a maximum size of n_{points}, but this buffer does not need to be filled for a comparison to take place. Parameters α and β are used to define a *threshold* function on the size of the buffer. If the buffer contains a small number of points, for example two points, and a third point were to arrive that is further away from the first point than the second, it could be an indication that the user is travelling away from the first point or it could be attributed to noise. This problem is mitigated by using a negative logarithmic function to ensure that the threshold for trend of motion is higher when there are fewer points in the buffer. Trend of motion is defined using a gradient that includes both spatial and temporal components, therefore allowing for the possibility of points of varying temporal distances. The *gradient* of buffer b is defined as:

$$\frac{|\mathbf{B}| \sum_{p_i \in \mathbf{B}} (D_t(p_1, p_i)\ D_s(C(\mathbf{V}), p_i)) - \sum_{p_i \in \mathbf{B}} D_t(p_1, p_i) \sum_{p_i \in \mathbf{B}} D_s(C(\mathbf{V}), p_i)}{|\mathbf{B}| \sum_{p_i \in \mathbf{B}} D_t(p_1, p_i)^2 - \left(\sum_{p_i \in \mathbf{B}} D_t(p_1, p_i) \right)^2} \tag{4.1}$$

Where $\mathbf{B} = \{p_1, p_2, ..., p_n\}$ is the current buffer, D_t is the temporal distance between two points, i.e. $D_t = |t(q) - t(p)|$, in minutes, and D_s is the spatial distance (in km) between two points. $\mathbf{V} = \{p_1, p_2, ..., p_n\}$ is the current *visit*, and $C(\mathbf{V})$ is the visit's centre of mass (i.e. the mean latitude and longitude value of all points). A gradient greater than the *threshold* indicates that the visit has ended:

$$-\alpha\ log \left(\frac{|\mathbf{B}|}{\beta} \right) \tag{4.2}$$

With the gradient summarising the movement trend of the user relative to the visit, and a threshold function that returns a value dependent upon the number of points over which the gradient was calculated, we are now able to determine when a visit has ended: namely, if the gradient is greater than the threshold for a given set of points then the visit can be marked as having ended. This ensures resilience to noise by monitoring the movement trend over a set of points, but still allows for visits with few points.

In addition to trend of motion exceeding the threshold, a visit ends if the temporal distance (D_t, in minutes) between the point being considered and the immediately preceding point is greater than the parameter t_{max} (line 14 of Algorithm 1), which is a similar technique to that used by Montoliu and Gatica-Perez [2010] to detect periods of missing data. Without this consideration, if data were to be missing for several hours or days, but by chance the first point recorded after this period is geographically close to the last point recorded, the gradient would be extremely low and is likely to be below the threshold. In this instance, the point would be considered as part of the previous visit, even though a significant period of time had elapsed and the user could easily have left the visit and simply returned to a location nearby. By introducing a maximum time between consecutive points for them to be part of the same visit, t_{max}, we mitigate this issue. Conversely, two consecutive points that fall on the same time instance (i.e. there is no time between them) could be erroneously identified as a visit. This is prevented in line 15 of Algorithm 1, where visits are only stored if they have a positive duration. A visit that consists of a single point, or a visit that consists of multiple points that fall on the same time instance would therefore be considered noise.

While parameter selection is dependent upon the specific application and dataset, we impose one primary constraint, namely $\beta > n_{points}$. If we permitted β to be less than n_{points}, it is possible for the threshold function to return a negative value when the size of the buffer exceeds β. Using a negative threshold would allow a visit to be characterised as having ended when the gradient returns a negative value, where a negative gradient indicates that the user is moving towards the centre of the visit, rather than away from it. By requiring $\beta > n_{points}$ we ensure that any value returned by the threshold function is positive. In some applications, it is possible for a minimum visit duration to be known. In these cases, selecting visits of an appropriate length can simply be performed after visit extracting has completed, by iterating through the set of visits and selecting those with duration greater than d_{min} minutes.

4.3.1 Clustering Visits into Locations

Once visits have been identified, existing applications typically cluster them to determine which visits belong to the same *location*. This can be achieved using DBSCAN, which, as discussed in Section 4.2.2, is a density-based clustering technique designed to limit the domain knowledge required to specify parameters. To this end, the algorithm takes only two parameters: *eps* and *minpts*,

Table 4.1: Parameters considered for the GVE algorithm as part of the parameter exploration.

Parameter	**Constraints**	**Values Considered**
α	≥ 0	0.1, 0.2, 0.3, . . . , 2.0
β	> 1, $> n_{points}$	6, 9, 12, . . . , 30
n_{points}	> 1	2, 5, 8, . . . , 26
t_{max}	> 1	10, 20, 30, . . . , 120

which together specify the minimum density of points, or in our case, visits, to identify a location. Visits are summarised into their centre of mass by calculating the average latitude and longitude of all points belonging to a visit, and these centres can then be clustered with DBSCAN. While existing approaches have also considered using the *centroid* of a visit for this summary, as GVE is designed to incorporate noisy points into visits without prematurely ending the visit, such points will have a magnified effect on the centroid, and so the centre of mass is used instead.

4.3.2 Properties of Extracted Visits

Understanding the GVE algorithm begins with an exploration of the parameter space, aiming to guide application developers towards appropriate parameter choices, as well as demonstrating properties of the algorithm itself. GVE takes 4 parameters: α, β, n_{points}, and t_{max}. The parameters α and β alter the threshold function, where the gradient of a buffer of points is calculated, and a gradient above the threshold for the current number of points in the buffer indicates the end of a visit, with α scaling the threshold function in the abscissa, and β in the ordinate. The value of x is the number of points over which the gradient was calculated, and y is the corresponding threshold value. The n_{points} parameter specifies the maximum size of the buffer, and t_{max} specifies the maximum amount of time between two consecutive points permitted for the points to belong to the same visit. Additionally, a constraint is placed on the parameters that requires $\beta > n_{points}$. Values of these parameters considered for this exploration are shown in Table 4.1.

Data

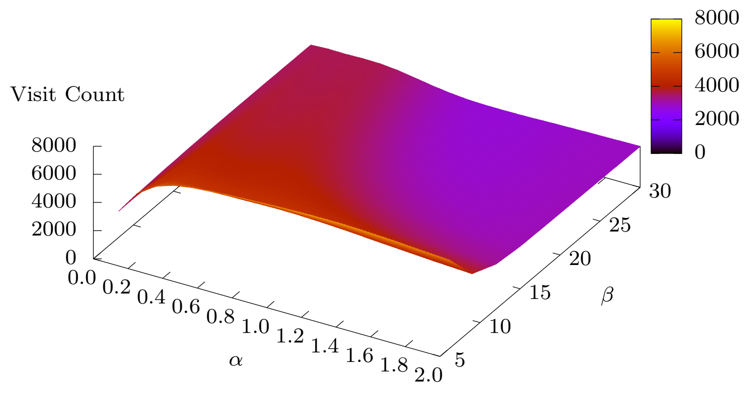
As discussed in Chapter 3, we use both the Nokia Mobile Data Challenge (MDC) and Warwick datasets for evaluation purposes, selecting 10 users from each. In this section, we perform all experiments on both datasets; however, where trends from results are the same across both, we focus on the MDC dataset and omit results from the Warwick dataset, instead presenting them in Appendix A. Both datasets have had some amount of pre-processing applied, specifically to remove periods of truncated location measurements from the MDC data, and to remove duplicate points from both, as discussed in Chapter 3.

Metrics

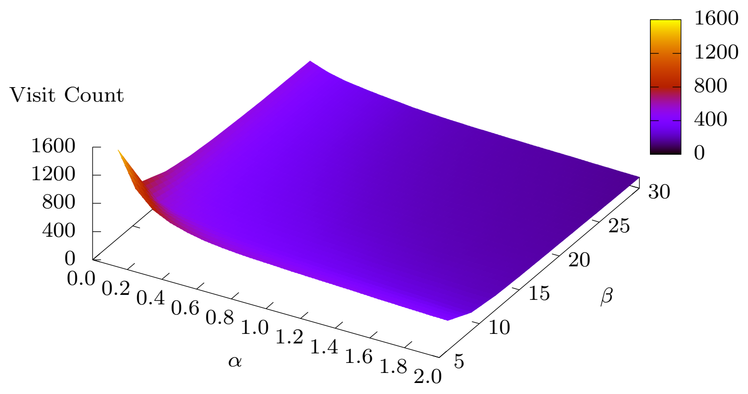
This exploration is conducted with respect to metrics including the duration of visits extracted, proportion of data points classed as noise (as noise points indicate movement, this proportion is analogous to the proportion of time spent travelling between visits), and the area of locations extracted. In these cases, while an application developer may not have specific target values, it is feasible that an acceptable range of values could be determined. As an example, if an application requires room-size locations, then the average area would be expected to be small, or if data were to be collected from a vehicle that is in motion for most of the day, the noise proportion would be expected to be high.

Results

Exploring the parameters for the GVE algorithm, Figure 4.2 shows the effect of varying α and β, which control the *threshold* function (a function on the current number of points in the buffer), while holding $n_{points} = 5$ and $t_{max} = 60$ minutes. As no optimal parameter values exist for unsupervised clustering, we opt to constrain parameters such that the resultant plots provide a representative view of the unconstrained parameters. Figure 4.2 also demonstrates the difference that the parameters have when using the two datasets, namely the MDC (Figure 4.2a) and Warwick (Figure 4.2b) data, discussed in Chapter 3. Focusing



(a) MDC: $n_{points} = 5,\ t_{max} = 60$.

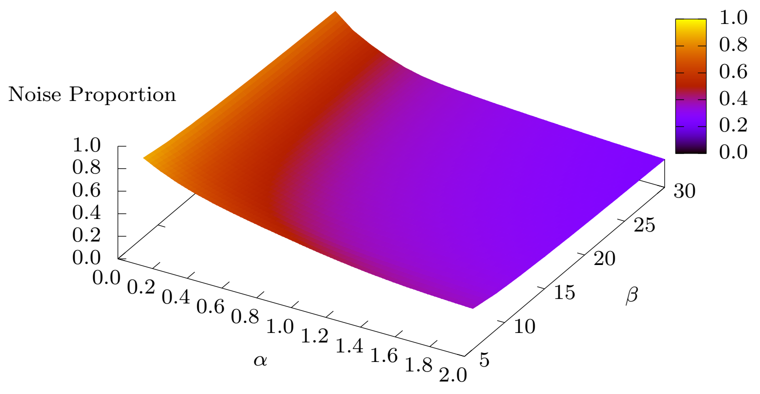


(b) Warwick: $n_{points} = 5,\ t_{max} = 60$.

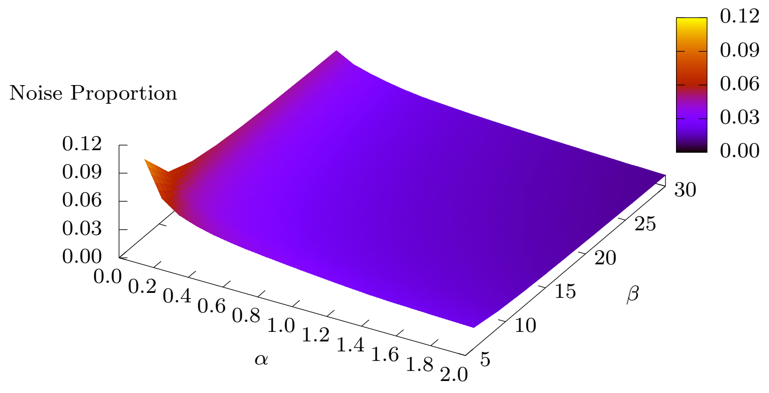
Figure 4.2: The relationship between parameters α and β and the number of visits identified for the GVE algorithm.

on the MDC data, the number of visits extracted increases when either α or β are lowered. When α becomes extremely small (< 0.4), this trend changes and the number of visits begins to decrease again. With low values of α, a lower gradient is required to mark a visit as having ended, but as α continues to decrease, fewer and fewer points are present in each visit as visits become shorter. In the context of visit extraction, visits consisting of no duration, including those that contain a single point, are considered noise and disregarded, so rather than the number of visits increasing, they actually begin to fall as α is lowered further. This effect is also shown in Figure 4.3a, where the proportion of data points classed as noise rises as the number of visits decreases. The number of visits extracted by the algorithm has a direct impact on the *accurate visits* use case, in that the correct characterisation of where a user spends their time requires the identification of the correct number of visits. If too few visits are extracted, multiple real-world visits will have been merged, causing a loss of information, thus invalidating the utility afforded by the start and end times of each visit.

Figure 4.2b shows the same results, but for the Warwick dataset instead of the MDC data, and helps to illustrate the differences that can be expected between different datasets. While the overall trend of producing a greater number of visits when lowering either α or β is present, very low values of α do not have the same effect as they do in Figure 4.2a. In fact, with the Warwick dataset, the number of visits extracted continues to increase as α is decreased. The corresponding noise proportion (Figure 4.3b) shows that although the proportion of points designated as noise increases as α is lowered, it only reaches a fraction of the value achieved for MDC (Figure 4.3a), indicating that the different properties of the Warwick data make it less susceptible to the effect that caused the visit counts to increase for low values of α with the MDC data. One possible reason for this is that the Warwick dataset has a lower collection rate (i.e. more time between data points on average) than the MDC (as shown in Table 3.1, Chapter 3), and as α is lowered over the MDC dataset, pairs of temporally close points become grouped into very short visits, which are then discarded as noise



(a) MDC: $n_{points} = 5,\ t_{max} = 60$.



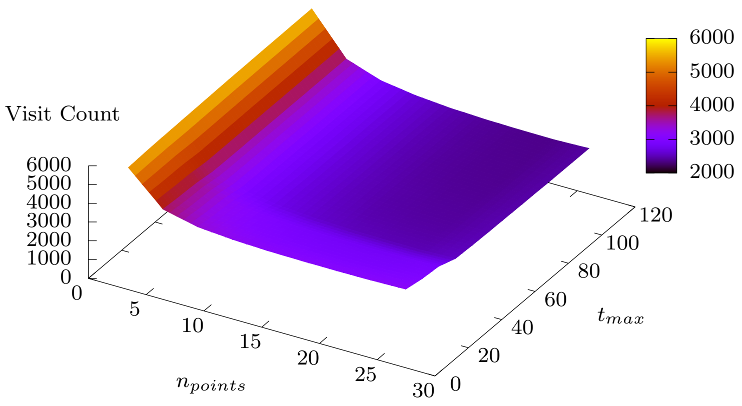
(b) Warwick: $n_{points} = 5,\ t_{max} = 60$.

Figure 4.3: The effect on proportion of trajectory points designated as *noise* when varying α and β for the GVE algorithm.

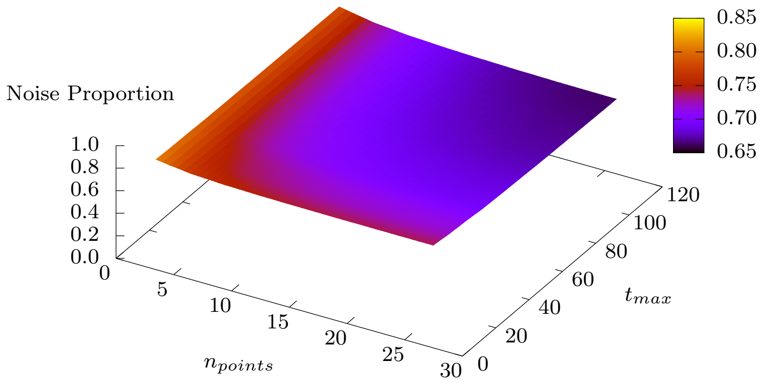
later as α is reduced further. In contrast, in the Warwick dataset, these short visits do not exist to the same extent (as evidenced by the significantly lower visit counts) and therefore there is nothing to discard, preventing the number of visits from dropping as α is lowered.

Figure 4.4 shows the effects of the remaining parameters, n_{points} and t_{max}, on the number of visits extracted and the proportion of trajectory points designated as noise when using data from the MDC dataset. In this case, values of the remaining parameters are selected as $\alpha = 0.1,\ \beta = 30$, which produce representative results. Figure 4.4a, showing the number of visits identified, demonstrates that t_{max} has minimal impact, while the effect of n_{points} is far more pronounced, where a lower value of n_{points} leads to a drastic increase in the number of visits identified. The maximum time between consecutive points permitted for them to belong to the same visit, t_{max}, has limited impact as it affects relatively few visits. With small t_{max} values, visits are split more frequently as smaller amounts of missing data cause a visit to be ended, slightly increasing the number of visits identified and also increasing the proportion of points designated as noise (Figure 4.4b). The maximum size of the buffer of trajectory points over which to calculate the gradient, n_{points}, has more of an impact as a smaller buffer leads to the gradient being considerably more susceptible to noise, prematurely ending visits and causing the number of visits extracted to increase, along with the proportion of points designated as noise. The trends depicted here are consistent across both the Warwick and MDC datasets, so we display only the MDC graphs here, and show graphs for the Warwick data in Appendix A.

The relationships between the parameters and the number of visits extracted are corroborated by Figures 4.5c and 4.5d, which show that when the number of visits extracted decreases, the average length of the extracted visits increases. While the maximum visit length (Figures 4.5e and 4.5f) also follows this general trend, it is important to note that low values of α do not have the same effect as with average visit length because the behaviour only affects very short visits.

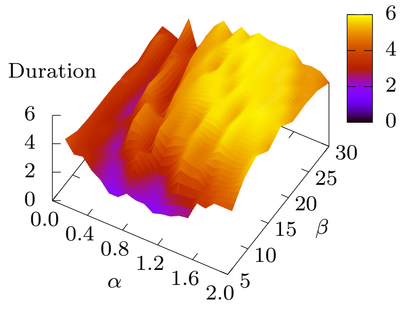


(a) Number of visits identified.

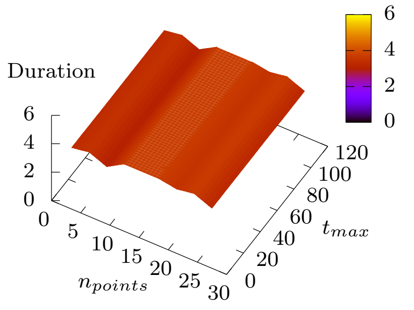


(b) Proportion of trajectory points designated as noise.

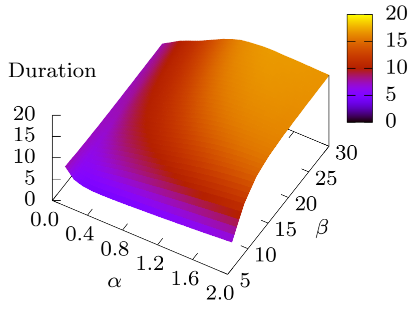
Figure 4.4: The effect of the parameters n_{points} and t_{max} on the GVE algorithm when extracting visits over the MDC dataset, with the remaining parameters fixed at $\alpha = 0.1,\ \beta = 30$.



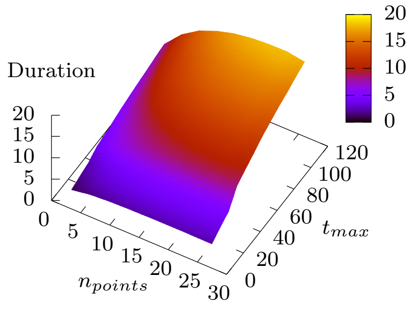
(a) Minimum visit duration (seconds).



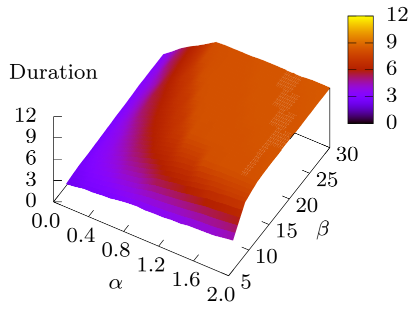
(b) Minimum visit duration (seconds).



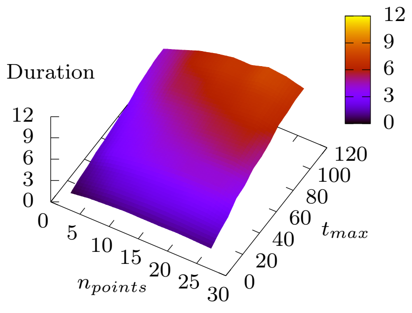
(c) Average visit duration (minutes).



(d) Average visit duration (minutes).



(e) Maximum visit duration (hours).



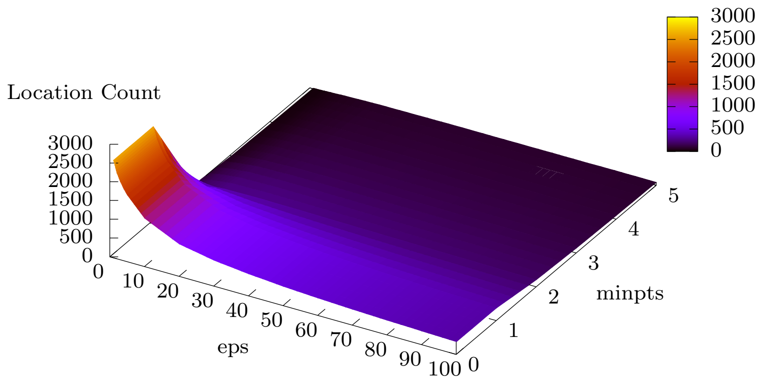
(f) Maximum visit duration (hours).

Figure 4.5: The effect of parameters on the minimum, average, and maximum length of extracted visits for the GVE algorithm on the MDC dataset, where constrained parameters were held at $\alpha = 0.1,\ \beta = 30,\ n_{points} = 5,\ t_{max} = 60$.

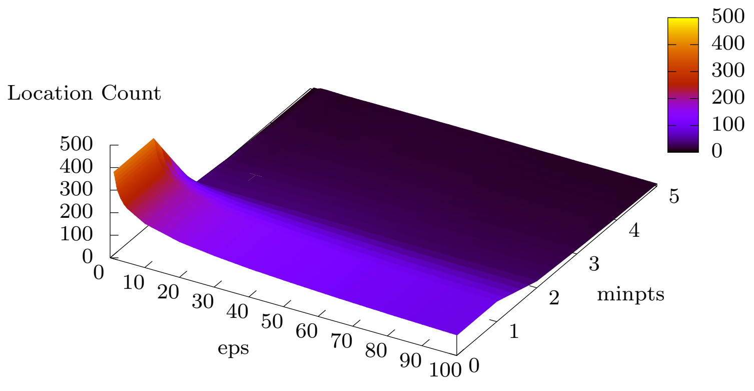
While the relationships between number of visits, average and maximum visit duration and proportion of noise are strongly correlated, minimum visit duration (Figures 4.5a and 4.5b) does not follow this trend. In fact, the length of the minimum visit extracted is not significantly dependent upon any parameter since GVE does not impose a minimum bound on visit duration. While parameters may still influence the length of the shortest visit, with a large dataset that covers a variety of mobility patterns, the effect of the parameters on minimum visit length is small. The results show that the shortest visit is extracted when both α and β are low, as this increases the likelihood of a visit being marked as having ended sooner. Additionally, lower values of n_{points} cause the length of the shortest visit to decrease, as the algorithm is only considering the trend of motion over a small number of points. Not imposing a minimum bound on duration is beneficial because it allows arbitrarily short visits to be extracted, although in some cases this may not be desirable. These figures show that extremely short visits are identified, but in many applications they may not be required. In this case, short visits can be removed easily in a single step once visit clustering has been completed by classifying the points belonging to them as noise retroactively.

Visit Clustering In existing work, the primary use for identified visits is that of determining significant locations, and so we explore properties of extracted locations based on the visits identified through GVE. For this, we select the parameters $\alpha = 0.1$, $\beta = 30$, $n_{points} = 5$, and $t_{max} = 60$, which produce an average of 488 (standard deviation: 218) visits across users in the Warwick dataset, and 3533 (standard deviation: 2469) visits across users in the MDC dataset. Employing DBSCAN, we investigate the impact of the parameters *eps* and *minpts* when performing clustering over these visits, where the values considered for these parameters are shown in Table 4.2.

Figure 4.6 shows the effects of *eps* and *minpts* on the number of locations extracted, for both the MDC (Figure 4.6a) and Warwick (Figure 4.6b) datasets.

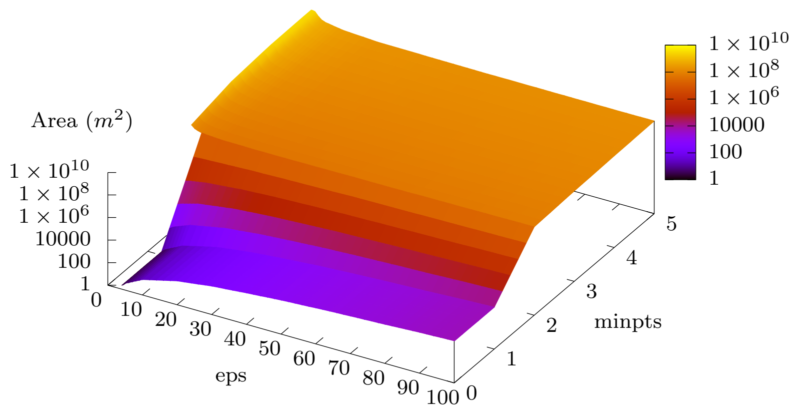


(a) MDC dataset.

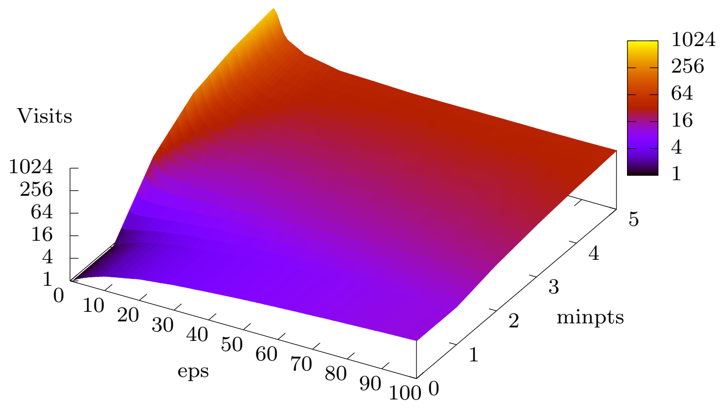


(b) Warwick dataset.

Figure 4.6: The effect of DBSCAN's *eps* and *minpts* parameters on the number of locations clustered from visits identified using GVE.



(a) Average location area.



(b) Average number of visits per location.

Figure 4.7: The effect of DBSCAN's *eps* and *minpts* parameters on the average size of locations and the average number of visits per location for the MDC dataset on visits identified using GVE.

Table 4.2: Parameters considered for DBSCAN as part of the parameter exploration.

Parameter	**Constraints**	**Values Considered**
eps	≥ 0	1, 2, 3, 4, 5, 10, 20, 30, ..., 100
minpts	≥ 0	0, 1, 2, 3, 4, 5

The trends in both datasets are identical, so we focus on the MDC data. The largest number of locations are extracted for small values of *eps* and small values of *minpts*, where *eps* specifies the distance (in metres) to consider as part of determining density. Small values require a higher density of points for a group of points to be clustered together, which therefore causes many smaller groups to be identified, while larger values cause these groups to merge into one location. Similarly, small values of *minpts* require fewer points within such distances to consider a cluster as a location. With *minpts* = 0 or 1, any single visit can become a location, resulting in vastly more locations extracted for these values. Requiring more points within the distance threshold leads to fewer locations being clustered and more visits considered noise.

Figure 4.7a shows the average area of locations extracted for different parameters over the MDC dataset, with results for the Warwick dataset in Appendix A (Figure A.3a). As *eps* is increased, locations are larger, but there are fewer of them. Similarly, increasing *minpts* also extracts fewer, but larger, locations. Finally, Figure 4.7b shows the average number of visits contained within each location, again for the MDC dataset with Warwick shown in Appendix A, Figure A.3b. For low values of *eps* and *minpts*, a single visit can be considered a location and therefore the average is close to 1. For higher values of each, the number of visits per location is increased, as the number of locations is less. Throughout this parameter exploration, we have shown how the parameters of the algorithms alter properties of the extracted visits and locations. These effects are summarised in Tables 4.3 and 4.4 for GVE and DBSCAN, respectively.

[1]Except for very low values on the MDC dataset.

Table 4.3: Summarised effects of the GVE algorithm's parameters as each parameter is increased.

Parameter	**Visit Count**	**Noise Proportion**	**Minimum Duration**	**Avg./Max. Duration**
α	Decreases[1]	Decreases	Negligible	Increases
β	Decreases	Decreases	Negligible	Increases
n_{points}	Decreases	Decreases	Negligible	Increases
t_{max}	Decreases	Decreases	Negligible	Increases

Table 4.4: Summarised effects of the DBSCAN algorithm's parameters as each parameter is increased.

Parameter	**Location Count**	**Area**	**Visits per Location**
eps	Decreases	Increases	Increases
minpts	Decreases	Increases	Increases

4.4 Evaluation: Comparison to Thresholding and the STA Extraction Algorithm

Towards our goals of demonstrating the applicability of GVE to the task of visit extraction, this section presents the results observed from performing visit extraction on real-world data in comparison to existing algorithms. Section 4.2 summarises the techniques for visit extraction discussed in Chapter 2. For the purpose of comparison, we select both the most widely used approach, *thresholding*, and the approach designed to improve upon thresholding by handling noise in data, *STA extraction.* This section explores the applicability of these two algorithms, in addition to GVE, in the context of the use cases presented in Section 4.1.2, through locations clustered with DBSCAN. In order to ensure consistency between the algorithms, all distances are measured in metres, and time in minutes. Furthermore, we define the shape of a location to be the convex hull of the centre of mass of each visit belonging to the location, as mentioned in Section 4.3.1. Summarising visits in such a way is an important part of extracting locations from identified visits as it vastly reduces the complexity of the clustering procedure, and thus is a technique we employ here [Andrienko et al., 2011, 2013; Ashbrook and Starner, 2002; Bamis and Savvides, 2011; Montoliu and Gatica-Perez, 2010; Zheng et al., 2010b].

Thresholding requires parameters r and t, specifying the maximum radius and minimum time of a visit, along with t_{max}, which performs the same purpose as in GVE, specifying the maximum amount of time between two consecutive points to consider them part of the same visit. The STA extraction algorithm requires the parameter N_{buf}, specifying the size of the buffer, and uses a weighted averaging function of window size N_d to characterise the trend of motion, with D_{thres} (measured in metres) specifying the threshold for the averaged buffers, above which the visit is considered to have ended. A relationship between N_d and N_{buf} was proposed by the authors of the STA extraction algorithm, namely $N_d = \frac{N_{buf}}{2}$, and we adopt this here [Bamis and Savvides, 2010].

4.4.1 Accurate Visits Use Case

The first use case, *accurate visits*, is concerned with characterising where a user spent their time, achieved through the accurate identification of visits. While it is not possible to state categorically the optimum parameters for achieving this goal for a given set of trajectories in the absence of a concrete ground truth, we can use the exploration of the parameter space to reason about suitable ranges of parameters. Using knowledge about the source and length of trajectories, it would be reasonable to suggest an expected minimum number of visits, thereby allowing us to constrain the available parameter space to only combinations that produce the correct number. Furthermore, the expected ratio of time that a user spent travelling against time spent at a visit could be estimated, imposing maximum values for the proportion of noise in the trajectories, as the proportion of noise is approximately equivalent to the proportion of a user's time that they have spent travelling.

In order to consider the applicability of each algorithm to the *accurate visits* use case, we investigate the properties of visits that are produced when using all three algorithms, over the Warwick dataset. Using the Warwick dataset for this evaluation ensures we have insight into the users who provided data and we are therefore able to construct a list of properties about the visits that we expect to be identified from this data:

- Users make an average of between 2 and 15 visits per day;
- The maximum length of a single visit does not exceed 2 days (2880 minutes);
- Each user made at least one short visit, for example to buy coffee or pay for a service, so the minimum visit duration must be less than 10 minutes;
- Each user spent at least 60% of their time at a visit location, and no more than 40% travelling between them (a noise proportion of 0.4 or less).

The parameter exploration conducted in Section 4.3.2 demonstrated that the GVE algorithm is capable of extracting visits that last only a few seconds. For this comparison, we are only interested in visits that have some significance to the user and so we adopt the procedure proposed in Section 4.3.2, namely that of removing extremely short visits retroactively, where we define a short visit to be one of less than 5 minutes in duration (i.e. $d_{min} = 5$). For consistency, this is applied to all algorithms considered.

While it is true these assumptions may not hold for every individual, for example it may be possible for a delivery driver to average more than 9.6 hours per day travelling (i.e. 40% of 24 hours), we believe that they apply to the majority of people and, specifically, all of the users who provided data for the Warwick dataset. Table 4.5 shows the parameter sets considered for this comparison, where only runs obeying the constraint $\beta > n_{points}$ are performed for GVE. In total, this results in 375 parameter combinations for GVE, 336 for STA and 324 for thresholding.

Focusing on the three algorithms, and using all combinations of parameters shown in Table 4.5, Table 4.6 shows details of the visits extracted that match the assumptions laid out earlier. Specifically, the table shows the percentage of parameter sets tested that match the assumptions (labelled as *% Match*), and the ranges of number of visits extracted, average visit lengths (in minutes)

Table 4.5: Parameter values used for evaluating visit extraction procedures.

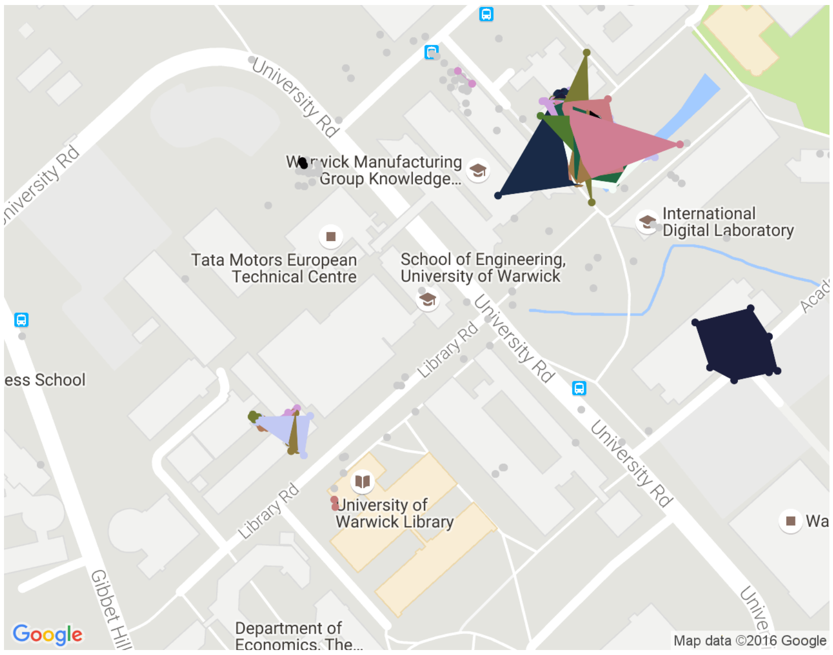
Algorithm	Parameter	Range	Values Considered
GVE	α	≥ 0	0.2, 0.6, 1.0, 1.4, 1.8
	β	> 1	6, 10, 14, 18, 22
	n_{points}	> 1	2, 6, 10, 14, 18
	t_{max}	> 0	15, 30, 45, 60, 75
STA	D_{thres}	> 0	0.1, 0.2, 0.3, 0.4, 0.5, 1, 1.5, 2, ..., 10
	N_{buf}	> 1	4, 6, 8, ..., 30
	N_d	≥ 1	$^{N_{buf}}/_{2}$
Thresholding	t	> 0	2, 6, 8, 10, 15, 20, 30, 45, 60
	r	> 1	10, 20, 30, 50, 75, 100
	t_{max}	> 0	15, 30, 45, 60, 75, 90

and noise proportions for each user. The results demonstrate that both GVE and thresholding are capable of extracting visits that match the assumptions for all users, but STA extracts no matching sets of visits for 3 of the users. This provides an indication that the STA extraction algorithm is less applicable than GVE to the accurate visits use case.

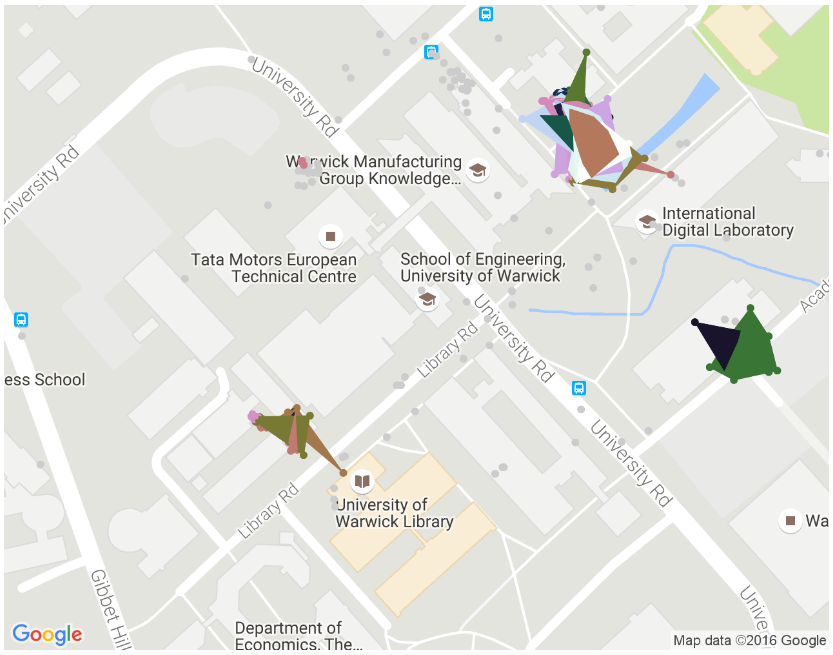
Thresholding and GVE both produce sets of visits that adhere to the assumptions; however, there are known shortcomings with thresholding regarding noise in the dataset, as noise will cause visits to be ended erroneously. Figure 4.8 shows example visits extracted for both GVE and thresholding from the same user's data. Visits are extracted in both cases from parameters that produce visit sets consistent with the assumptions outlined previously. Specifically, we select parameters by observing the results of plotting visits on a map and selecting only parameters that create visits consistent with the expectation that visits should not span multiple buildings. From this, the result with the lowest proportion of points designated as noise is selected for each algorithm. With visual observation confirming the correctness of visits, the visits with the lowest proportion of noise provide characterisation for the largest proportion of time. In the figure, visits are represented by different colours with the convex hull of the points belonging to each visit illustrated in the same colour as the points. The convex hulls have been included for clarity so as to present unambiguously which points belong to which visit, and a summary of the visits is shown in

Table 4.6: Summary of parameter sets that match expected values for different visit extraction techniques. *Match* shows the percentage of parameter sets that match the assumptions listed previously. Values are shown as a range with mean and standard deviation in brackets, and places where no parameters match are shown in bold.

Alg.	**User**	**Match**	**Visits**	**Avg. Length**	**Noise Prop.**
GVE	08	6.67	135-235 (172,29)	151-275 (218,35)	0.15-0.18 (0.16,0.01)
	1c	14.67	264-621 (390,87)	138-343 (239,54)	0.03-0.07 (0.05,0.01)
	1d	41.33	104-364 (220,73)	77-390 (186,86)	0.07-0.37 (0.27,0.09)
	24	16.0	284-556 (355,82)	89-180 (149,28)	0.03-0.06 (0.05,0.01)
	61	43.73	412-1328 (828,259)	60-321 (153,72)	0.04-0.13 (0.07,0.02)
	6b	60.0	48-369 (152,100)	53-748 (321,188)	0.01-0.08 (0.03,0.01)
	6c	20.8	645-1135 (781,86)	71-151 (112,13)	0.06-0.2 (0.1,0.03)
	85	53.33	138-724 (253,126)	90-499 (316,109)	0.01-0.07 (0.03,0.01)
	87	97.33	53-174 (82,24)	101-361 (242,62)	0.03-0.09 (0.05,0.01)
	95	1.07	733-819 (766,37)	118-133 (127,7)	0.06-0.07 (0.07,0.0)
STA	08	13.39	44-302 (159,74)	79-534 (242,118)	0.12-0.39 (0.24,0.08)
	1c	**0.0**	-	-	-
	1d	48.51	31-230 (77,34)	126-726 (438,117)	0.16-0.39 (0.28,0.06)
	24	**0.0**	-	-	-
	61	**0.0**	-	-	-
	6b	42.56	26-273 (59,34)	27-683 (438,201)	0.06-0.37 (0.16,0.08)
	6c	3.27	329-677 (528,120)	110-234 (157,41)	0.3-0.4 (0.34,0.03)
	85	23.21	145-749 (258,144)	77-431 (296,108)	0.1-0.39 (0.15,0.05)
	87	12.2	72-195 (104,30)	75-266 (192,46)	0.13-0.29 (0.16,0.03)
	95	11.61	116-972 (322,225)	57-588 (296,136)	0.1-0.39 (0.24,0.09)
Thres.	08	16.67	129-278 (194,45)	126-287 (197,49)	0.07-0.11 (0.09,0.02)
	1c	22.22	330-930 (523,186)	89-268 (185,56)	0.05-0.09 (0.06,0.01)
	1d	27.47	161-442 (238,70)	61-250 (170,54)	0.13-0.39 (0.26,0.08)
	24	11.11	302-449 (379,54)	112-167 (136,20)	0.04-0.05 (0.05,0.0)
	61	28.7	594-1471 (897,246)	51-208 (137,47)	0.08-0.22 (0.12,0.03)
	6b	31.48	106-390 (202,91)	49-335 (204,96)	0.03-0.09 (0.05,0.02)
	6c	19.14	682-1141 (900,125)	70-139 (101,18)	0.11-0.29 (0.2,0.05)
	85	11.11	361-546 (448,65)	119-184 (150,22)	0.05-0.06 (0.05,0.0)
	87	14.81	115-203 (145,29)	85-156 (126,23)	0.07-0.1 (0.08,0.01)
	95	12.65	638-1209 (826,147)	76-162 (126,24)	0.05-0.09 (0.07,0.01)


University Rd
University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European Technical Centre
School of Engineering, University of Warwick
Library Rd
University Rd
ess School
University Rd
University of Warwick Library
Library Rd
Gibbet Hill
Department of Economics, The...
Google
Map data ©2016 Google

(a) GVE ($\alpha = 0.2$, $\beta = 6$, $n_{points} = 2$, $t_{max} = 60$)


University Rd
University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European Technical Centre
School of Engineering, University of Warwick
Library Rd
University Rd
ess School
University Rd
University of Warwick Library
Library Rd
Gibbet Hill
Department of Economics, The...
Google
Map data ©2016 Google

(b) Thresholding ($t = 8$, $r = 30$, $t_{max} = 90$)

Figure 4.8: Example visits extracted for Warwick user *08*.

Table 4.7: Summaries of properties of the visits shown in Figure 4.8.

	Visits	**Avg. Visit (min)**	**Noise Proportion**
GVE	215	169	0.179
Thresholding	230	154	0.179

Table 4.7. In this case, GVE has extracted fewer visits, covering the same proportion of points and with a longer average duration, further indicating that thresholding is prone to splitting visits erroneously.

4.4.2 Location Properties Use Case

In order to understand how well each algorithm performs in the *location properties* use case, we construct a partial ground truth of buildings that we expect to see in the data and compare the extracted locations against these buildings. To achieve this, we manually identify 5 buildings on the University of Warwick campus unambiguously visited for each of our 10 test users (from the Warwick dataset). This is achieved by overlaying all points belonging to each user on satellite imagery of the university campus, and in combination with knowledge about the degree courses of each participant, 5 buildings unambiguously visited are selected and summarised into a minimum set of points, stored for later comparison. A comparison to the locations clustered through DBSCAN on visits identified by the GVE, STA and thresholding algorithms is then performed using the set-similarity measure *Dice's coefficient*:

$$QS = \frac{2|A \cap B|}{|A| + |B|} \tag{4.3}$$

Where QS is the quotient of similarity in the range $[0, 1]$. A value of 1 indicates that the sets (A and B) are identical, while 0 indicates no overlap. Although Dice's coefficient is designed for comparing the contents of two sets, it can be used over clusters by considering $|A \cap B|$ to represent the area covered by both clusters, and $|A| + |B|$ to be the sum of the areas of the clusters. This methodology provides an indication of how well the algorithms perform at extracting the

Table 4.8: Parameter increments selected for the optimisation procedure.

Algorithm	**Parameter**	**Range**	**Increments**
GVE	α	≥ 0	0.2
	β	> 1	2
	n_{points}	> 1	2
	t_{max}	> 0	10
STA	D_{thres}	> 0	0.2
	N_{buf}	> 1	1
Thresholding	t	> 0	5
	r	> 1	2
	t_{max}	> 0	10
DBSCAN	eps	> 0	5
	$minpts$	≥ 0	1

geographical shape of visited locations as described in the *location properties* use case in Section 4.1.2.

With a metric in place to quantify the correctness of locations extracted from a given set of parameters, we use a mathematical optimisation approach to select parameters that produce near-optimal scores for Dice's coefficient. Specifically, we employ *simulated annealing* as an example technique. While several techniques, as discussed in Section 4.2.4, are appropriate, we simply select simulated annealing for demonstration purposes and therefore select example functions where required. The algorithm starts at a random location in the state space, in our case the space of all possible valid parameters for the visit extraction algorithm and DBSCAN, used for clustering visits together to locations. This state is evaluated by performing visit extraction and clustering using the parameters, assigning a score to each ground truth location (i.e. the Dice's coefficient between the ground truth location and the most representative extracted location) and averaging these scores to give the set of locations an overall score. Our implementation of the simulated annealing algorithm aims to minimise *cost*, so we define cost as:

$$Cost = 1 - Score \tag{4.4}$$

Now the starting position has an associated cost, a random neighbour is selected, where a neighbour is defined to be a set of parameters where one parameter is one increment different from the current state (with increments shown in Table 4.8). This neighbour is then evaluated and if the cost is better (i.e. lower), the neighbour state is adopted as the current state, and the algorithm repeats. If the score is not better, it is taken with some probability dependent upon the *temperature*, a weighted measure of how much time has passed, and the *probability function*, a representation of how much worse the neighbour state is. For our purpose, we define these two measures by selecting values that empirically produce useful results:

$$T(r) = 0.985^{r \times 500} \tag{4.5}$$

$$PF(c, n, r) = e^{\frac{-(n-c)}{T(r)}} \tag{4.6}$$

Where r is the proportion of time expanded, c is the existing state's cost, and n is the neighbour state's cost. Regardless of whether the state is selected, the algorithm continues to repeat unless an optimal cost of 0 is encountered. At the end of the algorithm, the state observed with the lowest cost is assumed to be the solution. For this work, we select $k_{max} = 100$ and perform the algorithm on 10 randomly generated start states for each user, with the best cost selected, giving us an indication of how well each visit extraction algorithm can perform when considering the *location properties* use case.

Figure 4.9 shows the locations identified for an example user's ground truth, and Figure 4.10 shows the locations extracted for the best parameters observed for each of the three algorithms for the same user.

Table 4.9 shows the best score observed for each user and each algorithm, where the scores represent the average Dice's coefficient across all 5 ground truth locations and, therefore, a higher score is better. The results indicate that the three algorithms perform fairly consistency when it comes to extracting locations that match the ground truth locations, with scores of around 17-22%. Although a higher percentage may be desirable, this is unlikely with any technique due to the fact that a person may have only visited part of a building, or noise

Table 4.9: Summary of how well each visit extraction technique performs relative to a partial ground truth, where higher scores are more representative of the ground truth locations.

Algorithm	User	Score	Average (Std. Dev.)
GVE	08	0.237	0.218 (0.083)
	1c	0.201	
	1d	0.172	
	24	0.333	
	61	0.348	
	6b	0.132	
	6c	0.112	
	85	0.239	
	87	0.274	
	95	0.133	
STA	08	0.163	0.172 (0.093)
	1c	0.129	
	1d	0.11	
	24	0.308	
	61	0.148	
	6b	0.104	
	6c	0.114	
	85	0.216	
	87	0.354	
	95	0.074	
Thresholding	08	0.15	0.200 (0.084)
	1c	0.189	
	1d	0.146	
	24	0.314	
	61	0.231	
	6b	0.129	
	6c	0.175	
	85	0.226	
	87	0.353	
	95	0.082	

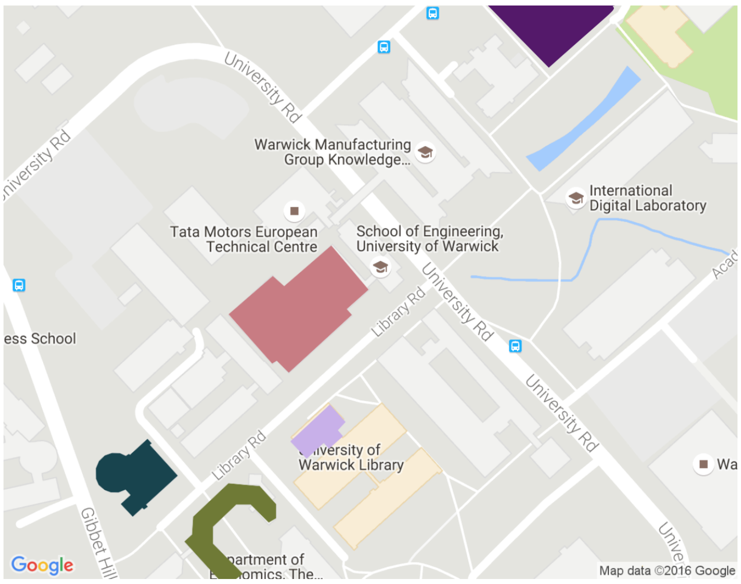
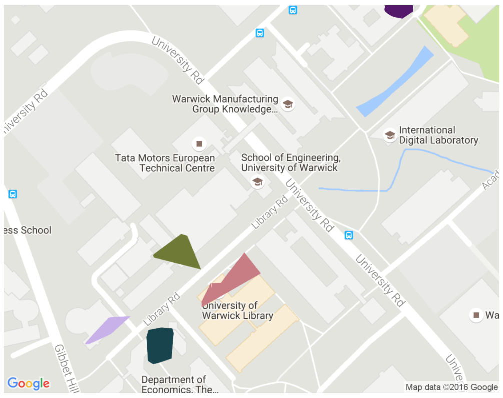

University Rd
University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European Technical Centre
School of Engineering, University of Warwick
Library Rd
ess School
University of Warwick Library
Gibbet Hill
Department of Economics, The...
Map data ©2016 Google
Google

Figure 4.9: Ground truth locations identified for Warwick user *61*.

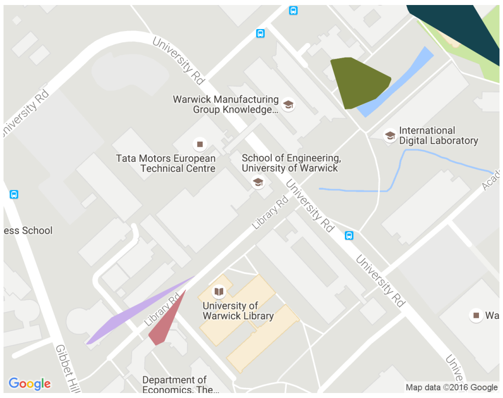
may cause identified visits, and therefore locations, to fall partly inside and partly outside the locations. In terms of the different algorithms, GVE performs slightly better than the other two. This is likely to be due to its improved handling of noise in the datasets, although the margin is close. This evaluation demonstrates that a relationship exists between the identified locations and the real-world buildings visited by the user, although the relationship is not perfect.

4.5 Predicting Over Extracted Locations

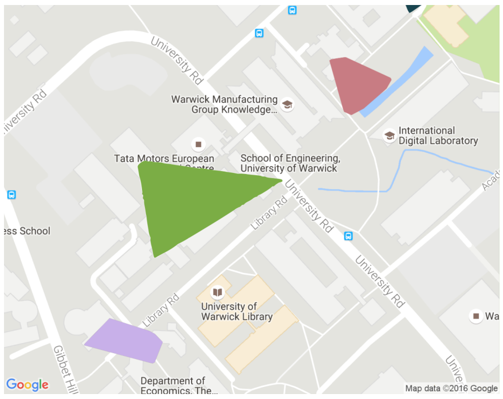
Through the parameter exploration in Section 4.3.2 and the comparisons to existing techniques in Section 4.4, we have demonstrated the applicability of the GVE algorithm to the task of identifying visits from trajectories and using these visits as a basis for extracting locations. We now turn our attention to a common application of such extracted locations, that of predicting the next location a user will visit. This section further demonstrates the applicability of the GVE algorithm through a sample application, alongside presenting a method of automatic parameter selection for location extraction and prediction


University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European Technical Centre
School of Engineering, University of Warwick
Library Rd
University of Warwick Library
Department of Economics, The...
Gibbet Hill
Google
Map data ©2016 Google

(a) GVE.


University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European Technical Centre
School of Engineering, University of Warwick
Library Rd
University of Warwick Library
Department of Economics, The...
Gibbet Hill
Google
Map data ©2016 Google

(b) STA.


University Rd
Warwick Manufacturing Group Knowledge...
International Digital Laboratory
Tata Motors European
School of Engineering, University of Warwick
Library Rd
University of Warwick Library
Department of Economics, The...
Gibbet Hill
Google
Map data ©2016 Google

(c) Thresholding.

Figure 4.10: Extracted clusters that maximise Dice's coefficient relative to the ground truth for Warwick user *61*. The different shapes are explained by the different techniques employed.

applications through a metric that characterises the goal of each of these tasks. Existing literature, as discussed in Chapter 2 and summarised in Section 4.2, has explored the task of location prediction from extracted locations, but has failed to account for properties of the extracted locations when considering the accuracy of predictions. Although evaluating locations extracted through unsupervised learning techniques is challenging due to the lack of available ground truth, the properties inherent in the locations have significant impact on the utility of the predictions. Selecting appropriate parameters for location extraction is therefore of paramount importance. However, the impact of altering parameters is typically unknown, as methods of evaluating the applicability of a set of locations to a particular task (e.g. prediction) are lacking.

The work presented here aims to overcome this problem by providing a method of parameter optimisation that selects the most appropriate parameters for both location extraction and the subsequent learning algorithm employed, in this case location prediction. The need for manual selection of parameters and review of location properties is therefore removed, and the robustness of any predications based on the locations is ensured, thereby increasing the utility afforded by such systems. Specifically, we: provide a metric for the evaluation of both extracted locations and predictions that characterises the goal of each of these tasks; frame the process of parameter selection as that of mathematical optimisation through the presented metric; and discuss characteristics of the metric while demonstrating its applicability over real-world data.

4.5.1 Evaluation Metric

In Section 4.4.2 we introduced the idea of using mathematical optimisation techniques to select parameters for location extraction, where a single metric was available for the task at hand, that of matching extracted locations against a ground truth. In this instance, optimisation is an appropriate choice because there is a clear measure of how *good* a set of parameters is. When considering location prediction, it could be considered desirable to achieve high predictive accuracies, and therefore a similar procedure could be employed aiming to maximise the accuracy of the predictor. Using locations extracted from trajectories as a basis for prediction, high predictive accuracies can be achieved simply by increasing the size of the extracted locations, as fewer locations make for simpler predictive models and are therefore likely to result in higher accuracy. This is not desirable because although the accuracy increases, the utility of predictions decreases significantly as locations become larger. For instance, knowing which building a person will visit is more useful to many applications than knowing in which city a person will be.

In order to evaluate the combined performance of location extraction and prediction, it becomes important to understand the true aims behind the process. For this work, we consider the aim of location prediction to be that of identifying the exact future location of an individual with as little uncertainty as possible. Uncertainty in this context can be considered to encompass both the size of locations and the accuracy of predictions, where very large locations can lead to high predictive accuracy with little utility, and very small locations can lead to very low predictive accuracy due to the increased complexity of the models required. It is even conceivable that, when aiming to identify the exact geographic region a user will visit, a prediction for a small, close-by, location that is incorrect may offer greater knowledge than a prediction for a vast location that encompasses the correct region, as the former case, although incorrect, identifies a position close to the correct one.

Evaluating both locations and predictions is difficult in standard approaches because of the dependency relationship between the two. The exact locations extracted will directly impact on the ability for prediction to occur, thus methods of characterising the locations and predictions independently cannot cater for this relationship. Therefore, the separate stages of location extraction and prediction must be evaluated together to get an honest representation of applicability to the given task. Towards this end, and taking the goal of location prediction to be that previously defined, the idea of aiming to identify as close as possible the region to be visited by the user, we consider *error* to be the is the distance between the centroid of the predicted location and the centre of mass of the region actually visited by the user, represented by the next visit in the data.

Intuitively, this definition of error favours small locations with accurate predictions, as wherever the actual region visited falls within such a location (i.e. an accurate prediction), the distance between the region and the centre of the location will be small. With large locations, which are undesirable for location prediction, a correct prediction may still have a high error if the distance between the location centroid and actual visited region is large. Similarly, for incorrect predictions (i.e. inaccurate ones), a small predicted location situated near to the correct region will give a low error as the distance between the predicted location's centroid and the actual visit made is small.

Under such a definition of error, while small locations are favoured, locations that are meaninglessly small (e.g. if every data point were to be classed as its own location) are prevented by the properties inherent in predictive systems. In order to predict future movements of individuals, past behaviour is analysed and patterns determined, but when considering such an extreme case, each location would have a single transition to another, unique, location, thus no repeating patterns can exist. This property ensures that accurate predictions from such training data are unachievable and therefore predictions will be little better than random, producing a high average error in the system. If the error were defined to be the distance between the centroid of the predicted location and the extracted location within which the user's next visit falls, a correct prediction would be given an error of 0 regardless of the size of the location. By using the distance between the centroid of the predicted location and the centre of mass of the actual visit, unless the location covers only this single visit, the error will be non-zero even for a correct prediction, with the magnitude dependent upon the location's size.

The error of a given set of locations and given predictor, E, can be calculated by using the Mean Absolute Error (MAE) metric, with distances calculated by the Haversine formula in kilometres [Robusto, 1957], formally:

$$E = \frac{1}{|P|} \sum_{l,v \in P} D_s(centroid(l),\ centreofmass(v)) \tag{4.7}$$

Where P is the set of all predictions, each prediction having two parts: $\{l, v\} \in P$, l is the expected next location of the user, and v is the actual visit that the user makes next. Centroid here assumes evenly distributed mass of points within the convex hull, while the centre of mass considers the distribution and takes the average latitude and longitude of all of the points.MAE is selected for its linear weighting of errors. While mean squared error is also a common metric, any large error would be weighted extremely highly and overshadow the remaining data (e.g. if the user visited a different city to the one predicted just once, even if all other predictions were correct), which is undesirable in this case. With MAE, a small number of large errors has significantly less impact and thus individual mistakes are still penalised, but not as highly. With an error metric selected whose definition, as described previously, is consistent with the expectations of the problem, comparison of sets of extracted locations and predictions can occur. Given two sets of locations and an associated prediction model, the more desirable set is the one which has the lowest associated error, E.

4.5.2 Optimisation Methodology

With an evaluation metric in place, the selection of optimal parameters for a given set of data would ideally be performed by evaluating all possible combinations of parameters and selecting those which result in the minimum error. In reality, however, performing location extraction and prediction is computationally expensive, and so it is infeasible to perform a complete search. Instead, a near optimal solution can be found using mathematical optimisation algorithms. For this task, we opt to use simulated annealing, as previously used to select parameters to optimise for a ground truth in Section 4.4.2, as it can overcome the problem of local maxima while maintaining a single state space. While other algorithms, such as evolutionary approaches, are also applicable, they typically assume that taking two states that individually produce good results and merging them will produce results at least as good. In this work, the interplay between parameters is extremely important and thus it is not immediately clear that this property will hold.

Extracting locations is performed using GVE followed by DBSCAN, as discussed previously. These locations are then used as a basis for next location prediction, i.e. aiming to identify the location out of the extracted set of locations that the user is most likely to visit upon leaving their current location. Specifically, we employ SVMs to perform the prediction, which have been shown to be an effective technique for this task [Wang and Prabhala, 2012], and use them in the same manner as discussed in Section 2.4.2. Once a set of locations has been extracted for a given user, the history of transitions between them is split into two with half used to train the predictive models, and the remaining half used as part of the evaluation procedure. An even test:train split is selected to ensure both sufficient training data for the predictive models is available and to evaluate performance. This testing data then becomes the basis for the score assigned to each set of parameters.

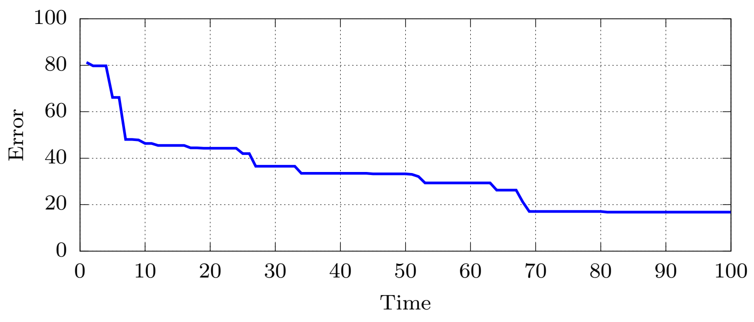
Minimising the error metric is performed using the simulated annealing algorithm and approach, as presented previously in Section 4.4.2. The neighbour, probability and temperature functions are defined as in Section 4.4.2, and the parameter increments shown in Table 4.8 are used for GVE and DBSCAN.

In order to understand the applicability of the proposed metric to the task at hand, multiple runs of the parameter optimisation approach must be performed with data from different users; in our case 10 users of the MDC and Warwick datasets. Furthermore, to understand the metric better, it is important to investigate the impact of using different segments of data from the same users, as different subsections of a user's trajectory will have inherently different properties. It has been shown that properties of data can impact the predictive accuracy attainable, through an exploration that considers sparsity and duration of data, where the duration is shown to have a greater impact than sparsity [Thomason et al., 2015c]. To this end, experiments use different amounts of data (continuous subsets of between 25% and 100% of the available data per user). With data selected and a methodology formalised, experiments can be run with different values of k_{max}, the parameter of simulated annealing that specifies the maximum number of iterations of the algorithm, where the selection of a neighbour, evaluation and possible adoption of a new parameter set is a single iteration. Terminating the algorithm occurs at this point. An alternative approach would be to use a tolerance of cost with k_{max} preventing extremely long runs, however, as this work is proposing to evaluate the cost metric, we allow the procedure to continue to find the lowest cost possible within the allowed time.

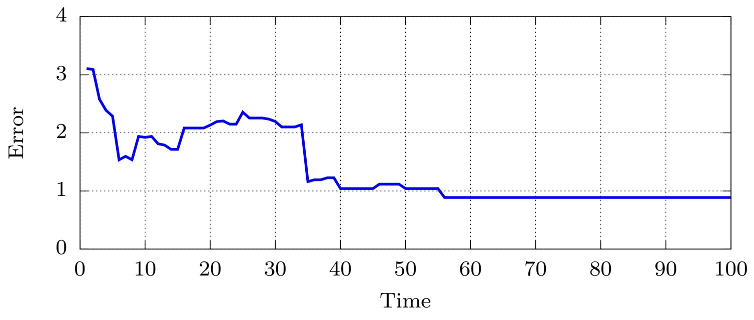
4.5.3 Results and Analysis

Figure 4.11 shows two example runs of the simulated annealing algorithm and how the error of the extracted locations and predictions varies over time. In both cases, the final error is significantly lower than the initial error (where randomly selected parameters were used to extract locations and perform predictions). In Figure 4.11a, the error is monotonically decreasing, so each iteration produces an error no worse than the one before. Conversely, Fig. 4.11b shows steps that move to a position of higher error on several occasions. This demonstrates simulated annealing's ability to overcome the local maxima problem, taking worse positions towards the beginning of the run, but converging towards a minimum as time runs out. After $t = 46$, no move to a worse position is made, instead, the error decreases before remaining constant.

Figure 4.12 provides indications of the relationship between the defined metric and the size of locations and accuracy of predictions. Both graphs are generated by taking all iterations of the runs from all users and placing the results into bins and averaging. Each bin therefore consists of different numbers



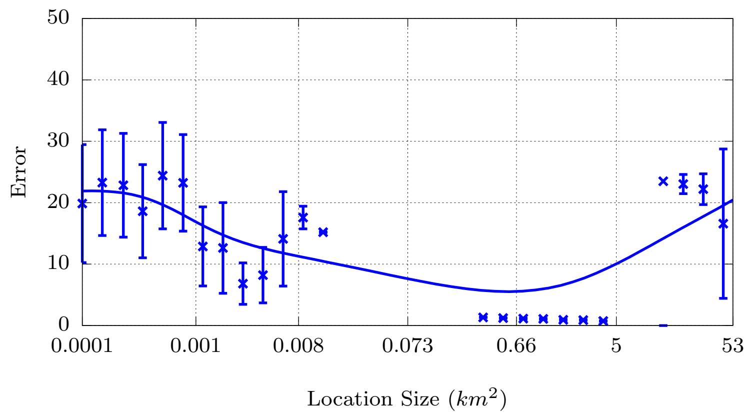
(a) Monotonically decreasing error.



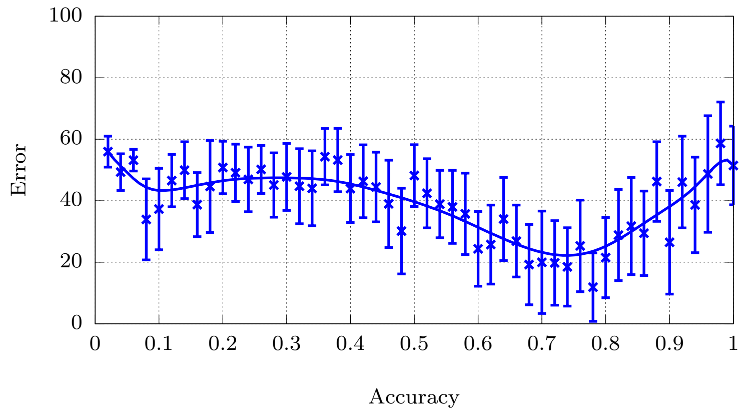
(b) Not monotonically decreasing error.

Figure 4.11: Example simulated annealing runs showing error against time.

of points, and therefore the lines only show an indication of the relationship present instead of a rigorous study of it. Specifically, Figure 4.12a shows the relationship between the proposed error metric and average location area (in km^2), and Figure 4.12b shows the relationship between the error metric and average predictive accuracy. Figure 4.12a shows that large locations typically incur a high error, which decreases as the locations get smaller up to a certain point. Once locations become extremely small in size, they encompass few visits and thus provide less training data, leading to the error increasing once again. The variation in standard deviation values shown in this figure can be explained by Figure 4.13. This shows the number of runs conducted that produce locations with average sizes that fit into each bin, demonstrating that places with errors above 10 that have low standard deviations are due to there only being a small number of runs over which to average error. Some of those with errors



(a) Average location size.



(b) Average predictive accuracy.

Figure 4.12: Indication of the relationship between the proposed error metric and properties of extracted locations and predictions

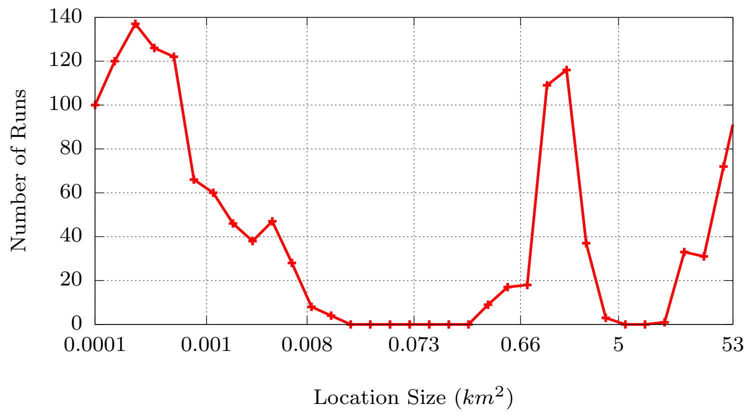
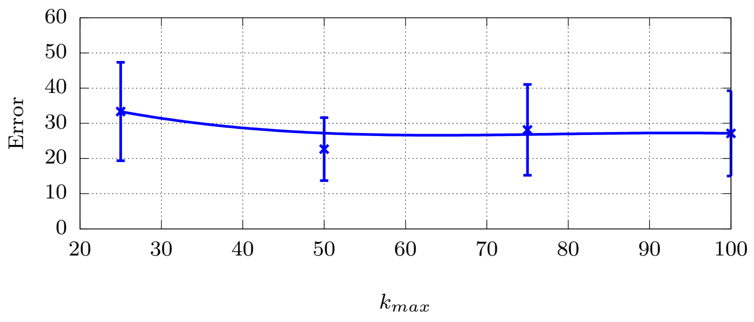


Figure 4.13: Number of runs conducted that produce locations with different average sizes.

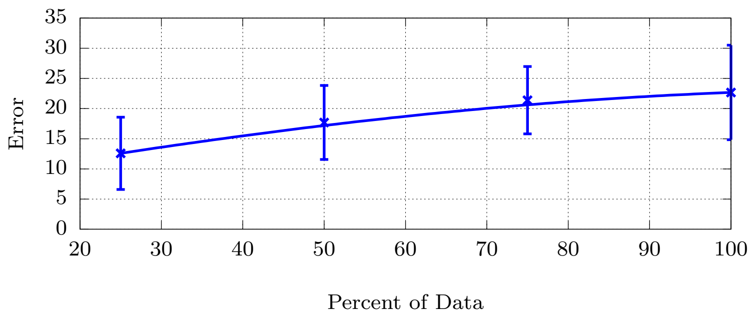
close to 0 that have low standard deviations have some of the highest numbers of runs, demonstrating that locations of between 0.3 and 4km^2 consistently produce small errors.

Figure 4.12b demonstrates that lower errors are also indicative of higher predictive accuracy, up to a point. As predictive accuracy increases towards 1, however, the associated error begins to increase. This is due to the situation where achieving such high predictive accuracy comes at the cost of location size. Due to the complexity of human mobility, achieving perfect predictions over small locations is extremely unlikely, so in the cases where predictive accuracy approached 1, the locations became larger and thus were penalised with a higher error. Combined, these properties demonstrate that the definition of error as proposed favours a balance between small locations and high predictive accuracy. These are desirable properties for this purpose as they serve to identify accurately where a user will be in the future with as little uncertainty as possible.

Finally, Figure 4.14 shows the effect on cost of the maximum number of iterations (i.e. k_{max}, Figure 4.14a) and percentage of each user's data used (Figure 4.14b), again showing results for both datasets. In the case of Figure 4.14a, the percentage of data used is held constant at 50%, and in Figure 4.14, k_{max} is held at 100, with results averaged over all users. As evidenced by the figures, the metric performs as would be expected. Specifically, as the number of iterations is increased, the average error decreases because the algorithm is allowed more moves to find the optimal position (Figure 4.14a). While k_{max} can be further extended beyond 100, it was observed that with $k_{max} = 100$, 82% of runs had converged to a stable error, indicating that the benefits of selecting a larger value for k_{max} would be minimal. However, providing more of the user's data for optimisation results in increased errors (Figures 4.14b). More data means the predictor has more information to model the user's behaviour, but it also means the user is likely to have visited additional locations for which no previous transitions exist, increasing the complexity of the required predic-



(a) Error against number of iterations.



(b) Error against percentage of data.

Figure 4.14: The effect of the optimisation procedure's parameters on minimum error encountered.

tive model, thus resulting in slightly increased errors as these new locations are likely to result in incorrect predictions. This combination of factors leads to the trends shown in the graph, where the increase in information results in a slightly higher average error.

Summary

This component of the evaluation of the GVE algorithm has considered GVE as a basis for location prediction through a proposed method for automatic parameter optimisation for location extraction and prediction that understands the aims of both tasks. While existing work has assumed the validity of extracted locations and focused on prediction alone, we argue that predictions are predicated upon the underlying locations. Therefore, ensuring the representativeness of such locations is of paramount importance when aiming to produce useful predictions. This section has presented a metric that considers both locations and predictions, and has demonstrated its utility through an evaluation of properties of extracted locations and predictions it favours. This metric provides a starting point for other domains that require parameter selection from a multi-stage procedure such as this.

4.6 Conclusion

This chapter has explored the problem of identifying visits, and subsequently locations, from geospatial trajectories. To this end, a novel algorithm, the Gradient-based Visit Extractor (GVE) has been presented that extracts periods of low mobility from geospatial data while maintaining resilience to noise and overcoming the drawbacks of existing techniques. Specifically, GVE does not place a minimum bound on visit duration, has no assumption of evenly spaced observations, and considers points as they arrive, making it amenable to visit extraction in real-time from a variety of data sources.

In addition to presenting the algorithm, this chapter has provided a comprehensive analysis of the properties of the visits extracted through an exploration of the parameter space, providing application developers with knowledge to aid in parameter selection. The applicability of GVE to the task of visit extraction has been demonstrated by a comparison to existing approaches through metrics representative of common goals of location extraction. Finally, an investigation into using extracted locations as a basis for prediction has been presented that includes a novel method of parameter selection through a metric that characterises the goals of both the extraction and prediction procedures. Through all of these investigations, results demonstrating the suitability of GVE have been achieved, with evidence indicating increased accuracy over existing approaches. This quantitative evaluation, lacking from previous work, demonstrates the applicability of the Gradient-based Visit Extractor (GVE) algorithm to the task of visit extraction and, consequently, as a precursor step to location extraction.

CHAPTER 5

Augmenting Geospatial Trajectories with Land Usage Data

Chapter 4 presented a method of identifying visits using only geospatial trajectories as a basis, and clustering these visits into arbitrary shapes that are likely to be meaningful to the users. However, as Section 4.4 showed, the extracted *locations* are not very representative of the real-world. While this method of identifying locations from trajectories is well-established in the literature, it does not take into consideration properties of the physical world.

Recently, the processing, storage, and connectivity capabilities of location-aware hardware devices have improved, allowing us to consider techniques that require additional data sources, or the ability to query a remote service for information. Making use of these developments, this chapter proposes a novel method of identifying geographical features (e.g. buildings, roads, amenities and areas) that a user has interacted with, creating a mapping between the extracted locations and the real-world. Achieved by augmenting trajectories with land usage data available after trajectory collection has occurred, the *Land Usage Identification (LUI)* procedure extracts *land usage elements*, referred to simply as *elements*, that a person has interacted with, and summarises these *interactions* in a manner consistent with the visits and locations of previous work. This chapter demonstrates the applicability of this approach through an evaluation and characterisation of the extracted elements, and through a sample application, that of predicting future interactions.

5.1 Introduction

Much existing work has focused on identifying locations from geospatial trajectories as a basis for prediction, aiming to determine the likely regions that an individual or other entity will visit in the future. While this is a useful component of many services, the identified locations do not necessarily correspond to actual geographic features, often spanning multiple buildings or areas. In contrast, this work takes the raw geospatial trajectories and augments them with information about the real-world to identify exactly which building, point of interest or geographical feature a person was likely interacting with, while maintaining compatibility with existing applications.

The Land Usage Identification (LUI) procedure places no additional burden on the user as no additional data is required to be collected from them; instead, additional information can be brought into the system in the form of land usage datasets available after collection has occurred. Through *augmentation*, *filtering* and *summarisation* techniques, the physical features that a user has interacted with are identified and their interactions summarised. This results in *elements* that replace the *locations* present in previous work, where each element has associated information describing its location and purpose. This additional information not only provides a foundation for understanding what a person may have been doing, but provides a relationship between the data and the real-world that can be leveraged by applications. This approach has the added benefit of considering periods of time, regardless of whether the person was stationary or moving. Location extraction techniques only consider time when the person was stationary, but land usage elements can be associated with trajectories regardless, identifying time spent on, for example, a road, train track, or sports field.

The utility of the LUI procedure is demonstrated through an exploration of the extracted elements and interactions, a comparison to extracted locations, and an exploration that uses the extracted elements as a basis for prediction, which is a common application of extracted locations. Utilising existing machine learning approaches, we demonstrate increased predictive accuracy over identified elements compared with using extracted locations for the purpose of next location prediction.

A discussion of relevant related work is presented in Section 5.2, and the LUI procedure is presented in Section 5.3. Section 5.4 evaluates the procedure and the utility afforded by the identified elements, with a conclusion and summary in Section 5.5.

5.2 Related Work

Identifying significant locations from geospatial trajectories is an area that has been covered before in the literature, typically split into two distinct stages. The first stage, visit extraction, identifies periods of low mobility by iterating through the trajectory, constructing visits according to some criteria. Once these visits have been extracted, they are clustered into arbitrary shapes, called locations [Andrienko et al., 2013; Li et al., 2008; Zheng et al., 2009; Zhou et al., 2014]. In addition to the techniques to achieve visit extraction and clustering discussed in Chapter 2 (Sections 2.2.3, and 2.3.1), Chapter 4 presented the Gradient-based Visit Extractor (GVE) algorithm, which aims to build upon the current state-of-the-art for visit extraction. The algorithm maintains resilience to noise lacking in many existing approaches and can handle data from a variety of sources, but does not consider geographic features when determining visits.

The idea of using land usage data to increase the meaning of extracted locations has been considered by Yan et al. [2013], who first extract visits (referred to as *episodes*) from raw trajectories, and then annotate these visits using data obtained from a land usage dataset, to create *semantic trajectories*. While this approach creates a mapping between extracted visits and geographic features, it does not consider the properties of the features when identifying visits, instead performing annotation only once the visits have been extracted.

Finally, the work in this chapter uses a sample application, that of location prediction, as a motivating example for extracting visits or interactions from geospatial trajectories. Location prediction has also been discussed previously in this thesis (in Sections 2.4 and 4.2), so a further discussion here is omitted.

5.3 Land Usage Identification (LUI) Procedure

This section presents the Land Usage Identification (LUI) procedure for identifying geographic features, represented by land usage *elements*, that a person interacted with, and summarising these *interactions*. The procedure itself uses geospatial trajectories, typically collected by location-aware devices such as smartphones or dataloggers, and augments these trajectories with data from a land usage dataset to identify all possible geographic features (e.g. buildings, roads, amenities, and areas) that a person could have been interacting with at any given time. The filtering of these identified elements is then performed to identify which element the user was likely to have been interacting with for a period of time. The benefits of this approach, instead of solely using geospatial trajectories as in existing approaches, is that it better captures the relationship between the identified locations and the real-world, with locations being representative in terms of shape, location and properties of the places actually interacted with by the user. These extracted *elements* and their *interactions* are interchangeable with *locations* and *visits* found in previous works, replacing the arbitrary clusters with meaningful elements.

The LUI procedure is split into three stages:

Augmentation — each point in a geospatial trajectory is augmented with all possible land usage elements that the user could have been interacting with.

Scoring and Filtering — the augmented trajectory is filtered to determine which element was likely being interacted with at any given time.

Summarising — the trajectory is summarised into *interactions*, associated with each element.

For this work, each point in a geospatial trajectory is assumed to have an *accuracy* value, measured in metres, that represents the confidence in the recorded location (i.e. latitude and longitude). Additionally, a land usage dataset is assumed to consist of sets of *entities* with associated information. Each entity

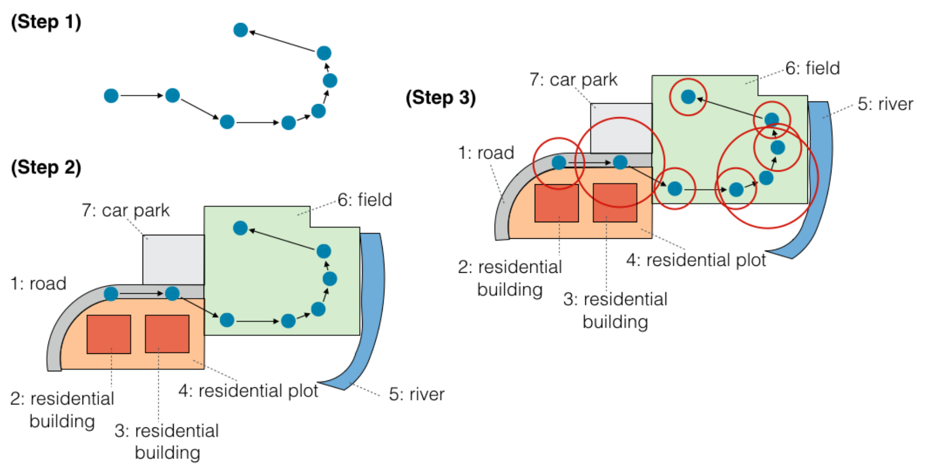


Figure 5.1: Example of the trajectory augmentation procedure.

represents a real-world object, feature or area, such as an individual post box, building or farm. Elements are assumed to have a collection of geographical coordinates that represent their location and shape, along with a set of 'key:value' pairs, called *tags*, that describe their properties. For example, a house may have the tag 'building:residential'.

5.3.1 Augmentation

The procedure for trajectory augmentation with relevant land usage elements is shown in Figure 5.1, where a raw geospatial trajectory (Step 1) enters the system and is overlaid on the land usage dataset (Step 2). The reported accuracy of each trajectory point is then used as a radius to consider (Step 3), such that all elements smaller than a specified size, *maxradius*, that are partially or wholly within the accuracy radius, are stored alongside the original point (summarised in Table 5.1). This is achieved by processing each trajectory point in turn automatically until an augmented trajectory is formed. These augmented trajectories are then subjected to a filtering procedure, as detailed in the following section.

Table 5.1: Augmented trajectory.

Point	**Time**	**Elements**
1	0	{1,2,4}
2	1	{1,3,4,6,7}
3	4	{6}
4	7	{6}
5	10	{5,6}
...	...	...

Table 5.2: Summarised trajectory.

Times	**Elem.**	**Tags**
0-1	1	road:residential
2-17	6	landuse:field
...	...	...

5.3.2 Scoring and Filtering

Once augmentation has been completed, identifying which land usage elements were most likely to have been interacted with is the task of a filtering procedure as part of a three-step process:

1. A buffer of points, and associated land usage data, is selected.
2. The land usage elements in the buffer are weighted and scored.
3. The element with the highest score is selected for association with the point at the centre of the buffer, the *point under consideration*.

Due to the nature of geospatial data collection systems, a continuous and evenly timesliced trajectory cannot be assumed, so selecting a buffer based on a fixed number of points would be inappropriate. Instead, we use a fixed temporal window for the buffer and consider all points that fall within this period. A buffer therefore consists of a *point under consideration* and the points falling within δ minutes immediately before and after this point. The pseudocode for maintaining such a buffer is presented in Algorithm 2. Each land usage element associated with any point within this buffer is then scored according to the number of points the element is associated with, the accuracy of these points, and the temporal distance from the point under consideration:

$$Score(e) = |P(e)| \sum_{p_i \in P(e)} \frac{1}{\sigma(p_i)} \left(1 - \frac{D_t(p_i, p_c)}{\delta}\right) \tag{5.1}$$

Where $P(e)$ is the set of all points that are associated with element e, $\sigma(p_i)$ is the accuracy value of point p_i, p_c is the point under consideration, δ is the width

Algorithm 2 Buffer management procedure.

```
points ← (p_1, p_2, ...) // input set
δ ← 5 // input parameter specifying width of each half of the buffer
buffer ← [ points.shift ]
output ← [ ]
index ← null

// Build the initial buffer
while points.length > 0 do
    // If index has not been set, then we are in the first half
    if index == null && TimeBetween(buffer[0], points[0]) > δ then
        // If the next point is greater than δ minutes from the first, this half is full
        index ← buffer.length − 1
    // If index has been set, then we are in the second half
    else if index ! = null && TimeBetween(buffer[index], points[0]) > δ then
        break // Exit the loop as adding the next point would exceed δ
    else
        buffer.append(points.shift)
    end if
end while

// Process the current buffer, increment index and maintain the new buffer
while points.length > 0 do
    output.append(Filter(buffer, index)) // Perform the actual filtering
    index ← index + 1

    // If the point for consideration is not in the buffer, then add it now
    if index == buffer.length then
        buffer.append(points.shift)
    end if

    // Remove any point from the first part that is not within δ minutes of buffer[index]
    while TimeBetween(buffer[0], buffer[index]) > δ do
        buffer.shift
        index ← index - 1
    end while

    // Add points until doing so would exceed δ minutes from buffer[index]
    while points.length > 0 && TimeBetween(buffer[index], points[0]) <= δ do
        buffer.append(points.shift)
    end while
end while

return output
```

Algorithm 3 Summarisation procedure.

```
trajectory ← ($p_1, p_2, ...$) // augmented trajectory
$t_{max}$ ← 5 // maximum time between consecutive points (minutes)
$d_{min}$ ← 10 // minimum visit duration (minutes)
elements ← ElementStore // store of elements and their interactions
interactionStart ← p1.timestamp
previousPoint ← p1

while currentPoint ← trajectory.shift do
    // If the elements differ or too much time has passed, end the interaction
    if (currentPoint.element != previousPoint.element or
        (currentPoint.timestamp - previousPoint.timestamp) > $t_{max}$) then

        // Only store interactions if they are long enough
        if (previousPoint.timestamp - interactionStart) > $d_{min}$ then
            interaction ← {start: interactionStart, end: previousPoint.timestamp}
            elements.addInteraction(previousPoint.element, interaction)
        end if

        // Mark the start of a new interaction
        interactionStart ← currentPoint.timestamp
    end if

    previousPoint ← currentPoint
end while
return elements
```

of half of the buffer (i.e. the number of minutes from p_c to consider) and $D_t(p, q)$ is the temporal distance between p and q (in minutes). This equation gives a higher score to elements associated with a large number of high accuracy points (where high accuracy is recorded as a small accuracy radius). The element with the highest score is then stored alongside the point under consideration. In rare cases, it is possible for two or more elements to share the same score, and in these instances, we select the element whose centroid is closest to the point under consideration.

5.3.3 Summarisation

Summarising the augmented trajectories into *interactions* is achieved through one-dimensional clustering that simply identifies neighbouring points that share the same land usage element, as shown in Algorithm 3. This procedure requires two parameters: t_{max}, which prevents periods of missing data from being included in an interaction by specifying the maximum amount of time that can exist between neighbouring points before the interaction is split, and d_{min}, the minimum duration required for an interaction to be stored. Upon completion, the procedure outputs a set of land usage elements that contain information about the element in the form of tags and coordinates, but also a set of times during which the user was interacting with the element.

5.4 Application and Evaluation

This section evaluates the utility of the land usage elements extracted through the augmentation, filtering, and summarisation procedures presented in Section 5.3. This evaluation takes the form of an exploration of the identified land usage elements, and an investigation into using such elements in lieu of extracted locations for the purpose of predicting future interactions. For consistency, in the remainder of this chapter, we adopt the following terminology:

Visit: a period of low mobility extracted solely from geospatial trajectories, as discussed in Chapter 4, indicating time when a person or entity remained in a single place.

Location: a cluster of *visits* based on geographical proximity.

Geographical Feature: a physical entity in the world that has some purpose, e.g. a building, road, public amenity or group of entities such as a university campus.

Element: a land usage element from a dataset, corresponding to a single geographical feature.

Interaction: a period of time spent interacting with, or within, an element.

Existing techniques therefore identify *locations* and *visits* to locations, while the technique proposed in this chapter instead identifies *elements* and *interactions* with elements. The output of the procedure is a set of interactions with elements performed by the user.

5.4.1 Data

For this work, and as discussed previously in Chapter 3, we use trajectories from 10 users of the Warwick dataset and 10 users of the Nokia Mobile Data Challenge (MDC) dataset for evaluation. It is important to note that the MDC dataset truncates the latitude and longitude values recorded around participants' residences and places of work, and we further treat these periods as missing. This is likely to have an impact on the elements identified through augmentation, and so we also use the Warwick dataset, which does not have the limitation. The augmentation procedure presented in this chapter also requires a dataset of land usage information in order to identify which elements were being interacted with at any given time. For this purpose, and as discussed in Chapter 3, we employ the OpenStreetMap (OSM) dataset. OSM spans the entire world and contains sets of elements with associated tags (e.g. descriptions of the functions of buildings, names of roads, etc.), and coordinate pairs that describe the shape and location of a feature, and thus forms the ideal basis for this work.

5.4.2 Exploring the Augmentation Procedure

Figure 5.2 shows sample data at each stage of the augmentation, filtering, and summarisation processes. Raw trajectory data, in the form of an ordered array of points (Step 1), enters the system. Each point has timestamp, longitude, latitude and accuracy values. Step 2 augments the trajectory with identifiers for all land usage elements that the user could have been interacting with at that time (as described in Section 5.3.1). This is achieved by extracting all land usage elements within the radius of the accuracy of the point and storing the identifier of each element. Step 3 shows the augmented trajectory once filtered (Section 5.3.2), with a single element associated with each point, representing the element likely to have been interacted with. Finally, summarisation occurs, clustering together contiguous time periods that belong to the same element (as

```
(Step 1)
    latlng: 52.3834499, −1.56026223
    timestamp: 2013−11−08 14:09:51.00 Z
    accuracy: 65.0

(Step 2)
    latlng: 52.3834499, −1.56026223
    timestamp: 2013−11−08 14:09:51.00 Z
    accuracy: 65.0
    data: [n_312873295, n_552101208, n_695942926, n_1014585845,
      n_1014585853, w_92341980, w_92342116, w_145179860,
      w_145179863, w_145179883, w_273005393, w_303748830,
      w_329376738, w_329376739, r_2437023, ...]

(Step 3)
    latlng: 52.3834499, −1.56026223
    timestamp: 2013−11−08 14:09:51.00 Z
    accuracy: 65.0
    data: [w_145179860]

(Step 4)
    w_145179860:
      tags:
        building: university
        building_levels: 3
      members: [n_1586185863, n_1586185883, ...]
      times:
        - begin: 2013−11−08 13:13:05.00 Z
          end: 2013−11−08 17:16:47.00 Z
      latlngs:
        - 52.3837765, −1.5601465
        - 52.3838285, −1.5600527
        - ...
```

Figure 5.2: Examples of the data at each stage of the augmentation, filtering, and summarisation processes.

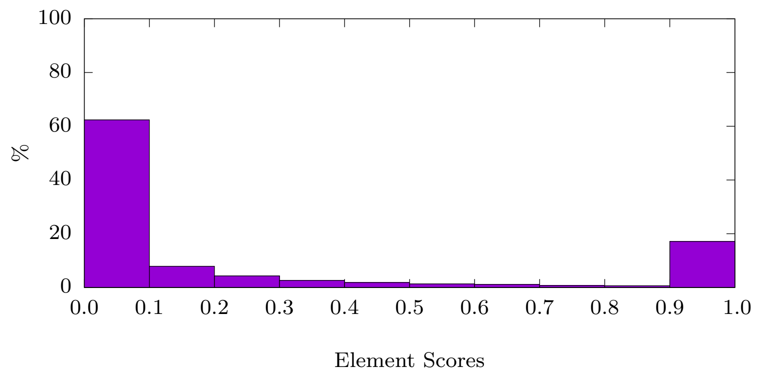


Figure 5.3: The distribution of normalised element scores before element selection takes place as part of the filtering procedure, for a sample user ($\delta = 5$).

described in Section 5.3.3), which is shown in Step 4.

The remainder of this section explores properties of the elements and interactions identified through the LUI procedure.

Element Filtering

Once trajectory points have been augmented with all possible land usage elements that the user could have been interacting with, the filtering procedure selects the element that the user was likely to have been interacting with at that time, achieved by assigning scores to each element within a buffer, and selecting the element with the highest score. We begin our evaluation by investigating the distribution of scores assigned to elements.

Element filtering and selection takes the parameter δ, which specifies the temporal distance from the *point under consideration* to consider as part of the buffer, in minutes. Holding $\delta = 5$, Figure 5.3 shows the distribution of scores for all elements in the filtering process (i.e. the values of *Score* from Equation 5.1), normalised, for all 24,571 trajectory points belonging to a sample user. This figure shows that the majority of elements are given low scores, which is likely to be due to appearing transiently in the data, while fewer elements achieve high scores during the procedure. The element with the highest score is selected and, therefore, this provides an indication that noise elements with low scores

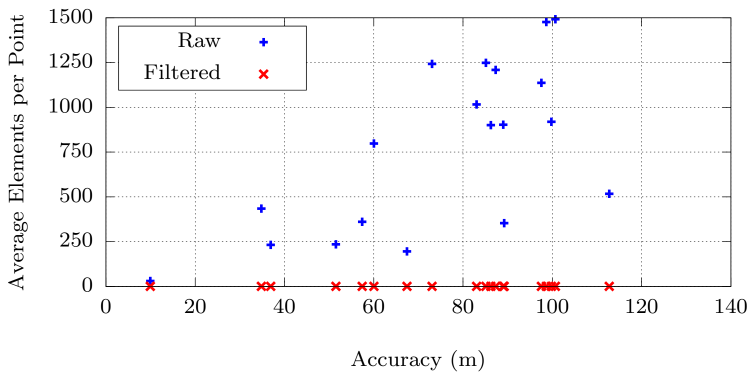


Figure 5.4: The effect of accuracy on number of elements, pre- and post-filtering, for different users' data, using $\delta = 10$, $maxradius = 50$.

are likely to be ignored.

With such a large proportion of noise elements indicated by Figure 5.3, we turn our attention to understanding how many elements are associated with each point in an augmented trajectory. Figure 5.4 shows the relationship between the accuracy of trajectory points and the number of extracted elements, where the accuracy value (in metres) determines the radius of land usage data to consider when creating the augmented trajectory. Specifically, the figure plots the average accuracy against the average number of elements per point for each of the 20 evaluation users. The figure demonstrates that a larger accuracy typically results in a larger number of elements per point and, as expected, filtering drastically reduces this number. Table 5.3 shows the same data, demonstrating the extremely large standard deviations present in the data, indicating that the accuracy and number of points pre-filtering can vary considerably amongst the data from one user. User *War_6b* has the lowest average number of elements post-filtering due to it also having a very small average accuracy, leading to an increased proportion of points when no land usage elements can be identified, due to deficits in the dataset. Figure 5.5 shows the effect of δ on the number of unique elements that make it through the filtering process. The figure demonstrates that as δ is increased, the number of elements is reduced as each buffer effectively averages over a longer period of time.

Table 5.3: Summary of point accuracy for each user of both datasets, with the average number of elements per point both pre- and post-filtering shown, along with variances in brackets.

User	**Accuracy**	**Pre Filtering**	**Post Filtering**
War_6b	9.9 (24.9)	30.8 (239.7)	0.8 (0.3)
War_1d	34.8 (139.4)	434.6 (1518.0)	1.0 (0.1)
War_08	36.9 (116.7)	232.6 (554.6)	1.0 (0.2)
War_87	51.5 (251.0)	235.4 (711.0)	1.0 (0.2)
War_1c	57.4 (181.5)	361.2 (1081.1)	1.0 (0.0)
War_95	60.0 (223.3)	798.0 (1489.0)	1.0 (0.1)
War_85	67.4 (1132.0)	196.0 (870.2)	1.0 (0.0)
MDC_5976	73.1 (58.0)	1242.2 (1405.1)	1.0 (0.0)
MDC_5966	83.0 (61.5)	1016.3 (1645.2)	1.0 (0.0)
MDC_5947	85.1 (73.4)	1248.1 (1473.2)	1.0 (0.0)
War_6c	86.2 (261.0)	900.6 (2983.4)	1.0 (0.0)
MDC_6104	87.3 (58.6)	1208.8 (1786.2)	1.0 (0.0)
MDC_5990	89.0 (70.3)	902.8 (1287.0)	1.0 (0.1)
MDC_5938	89.2 (58.4)	354.0 (1034.6)	1.0 (0.1)
MDC_5927	97.5 (70.1)	1136.8 (1914.5)	1.0 (0.0)
MDC_5948	98.7 (66.5)	1476.4 (1901.5)	1.0 (0.0)
MDC_6051	99.8 (61.3)	919.6 (1091.6)	1.0 (0.0)
MDC_6109	100.7 (67.1)	1491.7 (1904.5)	1.0 (0.0)
War_61	112.8 (379.0)	517.2 (2156.7)	0.9 (0.3)
War_24	141.2 (584.3)	334.4 (1542.3)	1.0 (0.0)

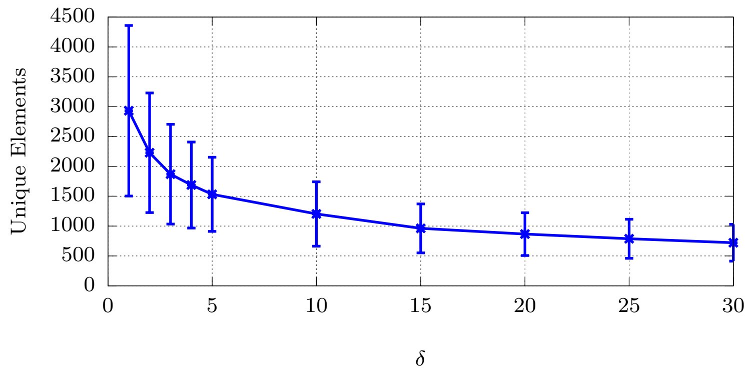


Figure 5.5: The effect of δ on the number of unique elements identified through the LUI procedure ($maxradius = 50$).

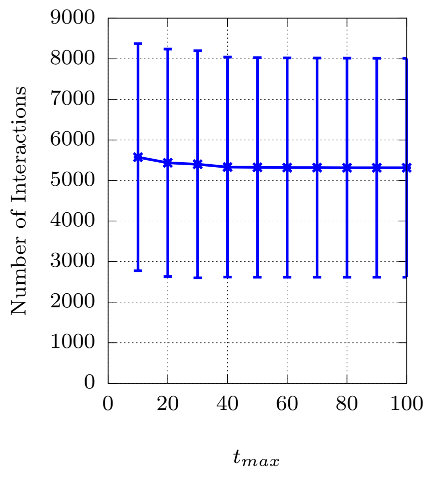
Summarisation

Once the data has been filtered, it is summarised into continuous periods of time. Two parameters, t_{max} and d_{min}, are required for this, specifying the maximum amount of time (in minutes) between consecutive points for them to be considered contiguous, and the minimum duration of an interaction for it to be stored, respectively. Using the same 20 users and the filtering parameter $\delta = 10$, Figure 5.6 shows how t_{max} affects the number and duration of interactions extracted (ignoring d_{min} by setting it to 0), and Figure 5.7 shows the effects of d_{min} on the same metrics (holding $t_{max} = 100$). The figures show that increasing either t_{max} or d_{min} causes fewer, but longer, time periods to be extracted as interactions that otherwise would have been split can remain merged. Figure 5.6a shows that this trend is less with t_{max}, and also that there is significant variance between the users, so the trend is less well defined for this parameter, which is likely to be due to t_{max} impacting on fewer visits than d_{min}.

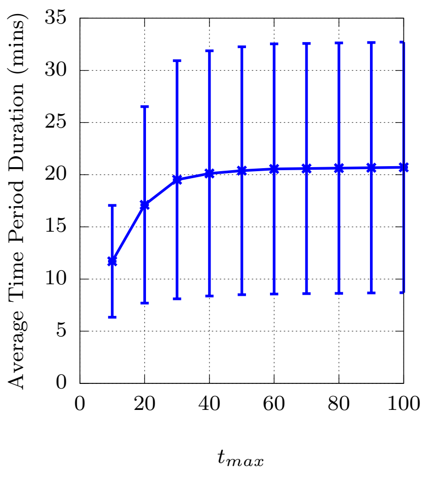
5.4.3 Comparison with Extracted Locations

With an understanding of the elements identified by the LUI procedure, we now turn our attention to comparing the identified elements against locations extracted through existing approaches. Specifically, we select thresholding, the most widely used approach, and GVE, the approach presented in Chapter 4 that builds upon the existing state-of-the-art, for comparison. Clustering identified visits into locations is then performed by DBSCAN.

Thresholding takes the parameters *radius*, *time* and t_{max}, specifying the maximum width of a visit, minimum duration of a visit and maximum time between consecutive points for them to be considered as part of the same visit, respectively. For this comparison we are aiming to identify locations no larger than a typical building, and so set the *radius* parameter to 50m; a value for t_{max} of 1 hour allows for short periods of missing data, but will prevent longer

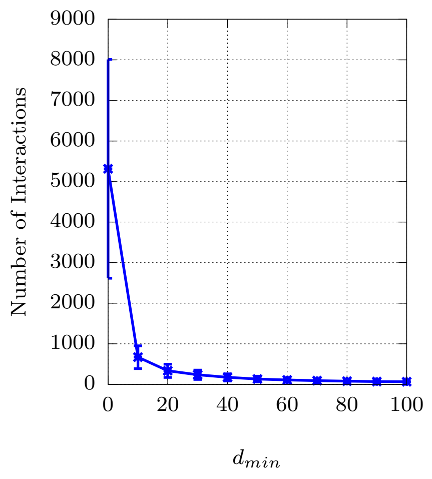


(a) Number of time periods.

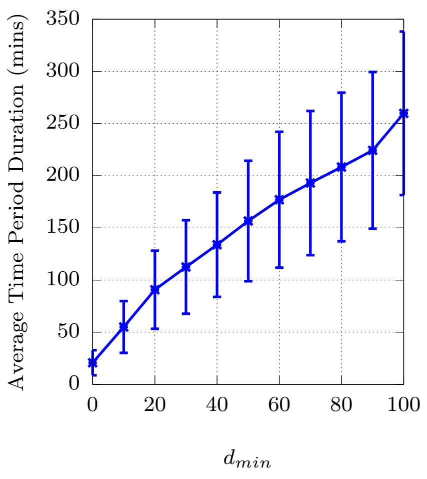


(b) Average time period duration.

Figure 5.6: The effect of t_{max} on the summarisation procedure ($d_{min} = 0$).



(a) Number of time periods.



(b) Average time period duration.

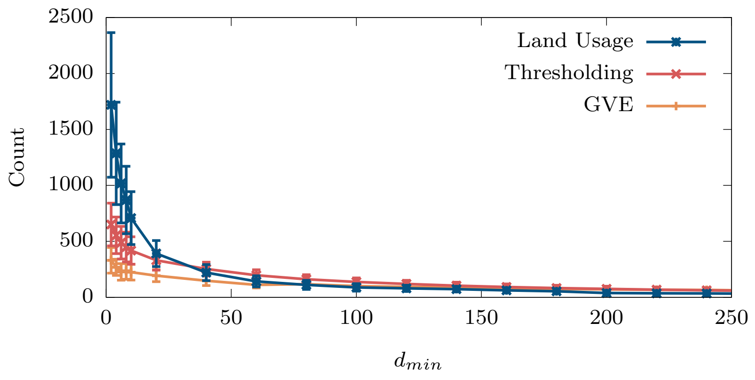
Figure 5.7: The effect of d_{min} on the summarisation procedure ($t_{max} = 100$).

periods from being included in interactions where the user may have left and returned some time later. For the MDC dataset, larger periods of missing data are expected as we have removed periods of data where latitude and longitude values were truncated (as discussed in Chapter 3, Section 3.1.4), and so we ignore the t_{max} parameter when generating results for the MDC dataset. Finally, the *time* parameter, equivalent to the d_{min} parameter in land usage extraction, is left open and its impact explored as part of the evaluation.

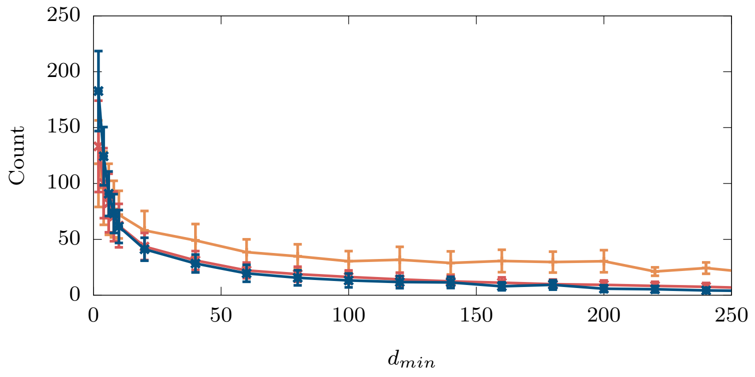
The parameters for GVE allow for tuning the algorithm, but do not map neatly to the real-world properties of the extracted interactions (e.g. size and duration). To get around this, and produce comparable results, we first extract locations using thresholding, followed by DBSCAN, and then select parameters for GVE that extract locations of approximately the same size, using the simulated annealing-based methodology proposed in Chapter 4 (Sections 4.4.2 and 4.5) for this purpose. This process uses a mathematical optimisation procedure to minimise the difference in average location sizes between the data clustered with thresholding and that clustered with GVE. DBSCAN is used in both cases to cluster visits with $minpts = 0$, i.e. a single interaction can be considered as a location, and $eps = 15$m, ensuring visits must be within proximity to be considered as part of the same location.

The LUI procedure is also performed on the same trajectories for comparison. In order to produce a representative comparison, parameters are selected that aim to mirror the extracted locations as well as possible. To this end, the maximum element size is constrained to be 50m across; $t_{max} = 1$hr is used for the Warwick dataset (and ignored for the MDC dataset). The same values of d_{min} as used to extract locations are used for exploring its impact on predictive accuracy. The only additional parameter required by this procedure is δ, specifying the width of the buffer to consider during the filtering stage of trajectory augmentation. Here, we set $\delta = 5$min for this task, a value selected empirically that produces representative results.

Figures 5.8, 5.9 and 5.10 show summaries of different properties of the ex-



(a) Visit/Interaction Count.



(b) Element/Location Count.

Figure 5.8: Average numbers of interactions and locations extracted for the LUI procedure and location extraction techniques.

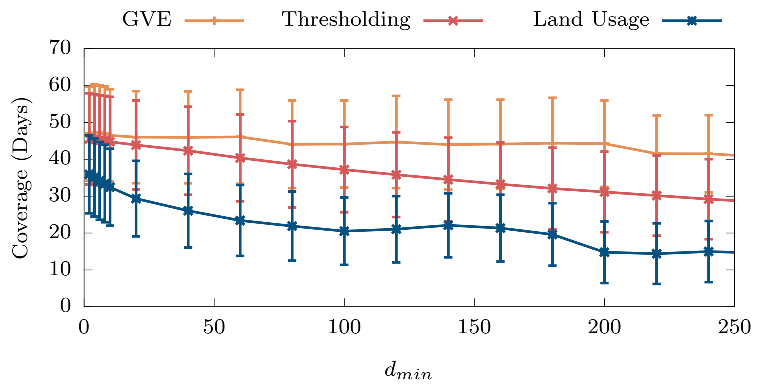


Figure 5.9: Average duration of interactions for the LUI procedure and location extraction techniques.

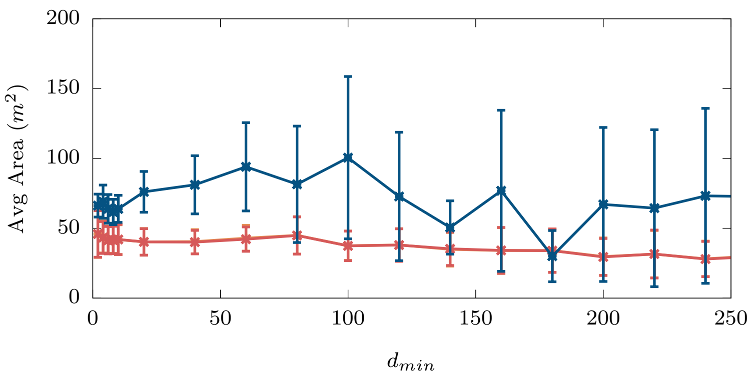


Figure 5.10: Average size of elements and locations for the LUI procedure and location extraction techniques.

tracted locations and elements for different values of d_{min}, averaged over all users of the Warwick dataset. In all cases, the trends present are the same between users of the Warwick and MDC datasets, so some of the MDC results are omitted from this chapter and instead placed in Appendix B. Figure 5.8a shows the number of interactions and visits identified by each technique, and Figure 5.8b shows the number of different extracted locations or identified land usage elements associated with these interactions. Figure 5.9 shows the total time contained within these interactions, with results also displayed in Table 5.4. As d_{min} is increased, the total coverage falls, but it is also worth noting that there is very high variance between users regardless of the technique, indicating that the coverage is heavily dependent upon properties of the user's data. This is as expected, as the users have a wide variety of trajectory durations, and therefore the variance in time covered by extracted visits and interactions is also high. Figure 5.10 shows the average area of locations and land usage elements, demonstrating that higher values of d_{min} lead to fewer interactions, locations or elements. The average area is least impacted by d_{min}, but is also the most varied among the techniques. GVE and thresholding have similar values, as parameters for GVE were selected specifically to extract locations with similar sizes to those identified through thresholding, while the land usage elements are consistently larger. Larger land usage elements are to be expected as

Table 5.4: Total visit and interaction time in days for the different techniques, averaged over the Warwick users. Results are shown for different values of d_{min}, with standard deviations shown in brackets.

d_{min}	Land Usage	Thresholding	GVE
2	35.9 (10.6)	45.6 (12.4)	47.0 (12.7)
4	35.1 (10.6)	45.4 (12.3)	47.2 (13.1)
6	34.1 (10.6)	45.1 (12.3)	47.2 (12.9)
8	33.4 (10.5)	44.9 (12.2)	46.9 (12.8)
10	32.4 (10.5)	44.8 (12.2)	46.5 (12.5)
20	29.3 (10.2)	43.9 (12.1)	46.0 (12.5)
40	26.1 (10.0)	42.3 (11.9)	45.9 (12.5)
60	23.4 (9.6)	40.4 (11.8)	46.1 (12.7)
80	21.9 (9.4)	38.7 (11.7)	44.1 (11.9)
100	20.5 (9.1)	37.2 (11.6)	44.1 (11.8)
120	21.0 (9.0)	35.8 (11.5)	44.7 (12.5)
140	22.1 (8.7)	34.5 (11.4)	44.0 (12.2)
160	21.3 (9.0)	33.2 (11.3)	44.1 (12.0)
180	19.6 (8.5)	32.1 (11.1)	44.4 (12.3)
200	14.8 (8.3)	31.1 (10.9)	44.2 (11.7)
220	14.4 (8.2)	30.2 (10.9)	41.5 (10.4)
240	15.0 (8.3)	29.2 (10.9)	41.5 (10.5)
260	14.6 (8.1)	28.4 (10.6)	40.7 (10.5)

extracted locations only consider regions where users actually spent time, while land usage elements consider entire buildings or features where a user may have only interacted with part of it. The variance of land usage element sizes is also considerably larger than locations as extracted locations are typically around the same size, while interactions with very small or large elements are possible.

In order to reduce the variance of the area of land usage elements, we also extract elements using the same procedure, but limit the size of elements by imposing a maximum area in addition to the maximum radius used previously. Figure 5.11 shows these elements, using a maximum area of 400m^2, selected empirically, and demonstrates areas much closer to those of the extracted locations, and consequently, a reduced variance. The average areas of the different techniques, along with their standard deviations, are shown in Table 5.5.

The results in Figure 5.10 show that the elements identified from augmented trajectories are often larger than the locations solely extracted from trajectories. While this comes with an increased representation of geographic features,

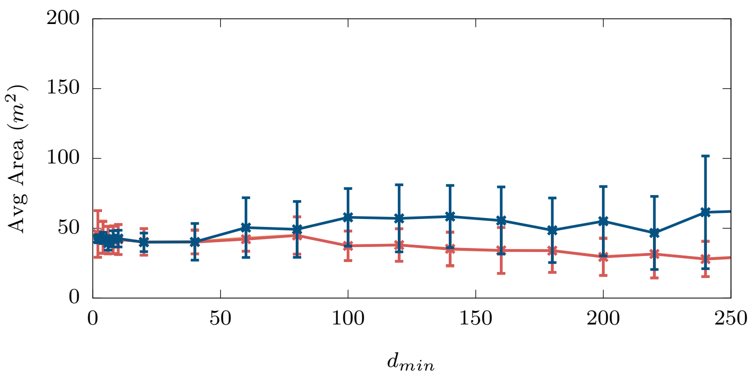


Figure 5.11: Average size of elements and locations for the LUI procedure and location extraction techniques, where land usage elements are restricted by both radius and area.

it is possible that the extracted locations may have greater meaning in certain circumstances. If, for instance, children play in part of a park which borders onto a street, locations extracted from trajectories would be able to identify that the park and street are considered one single location by the children, an *invisible space*, even though they span multiple physical features in the environment. Attempting to extract such a location from augmented trajectories would yield either the whole park or the street, but would be not be able to consider them together. This property may be desirable in some circumstances, but undesirable in others, so it is important to consider which properties are useful when selecting which approach to use for any given application.

5.4.4 Predicting Land Usage Interactions

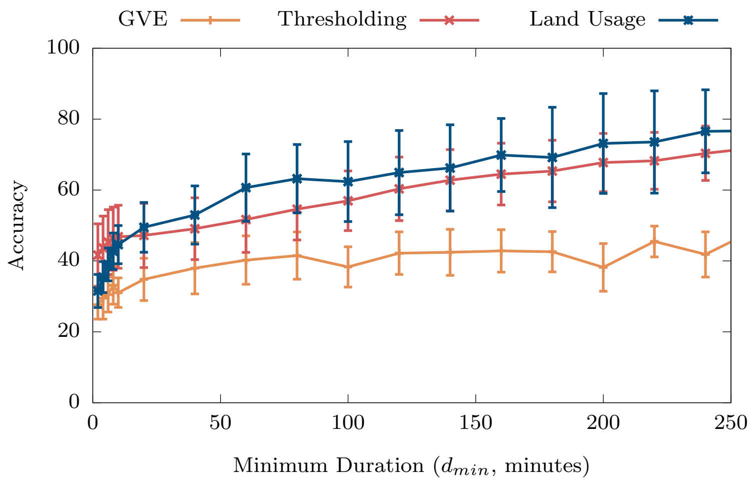
In order to understand the utility of the identified elements and their interactions, we turn our attention to a common use for the locations extracted from geospatial trajectories, that of predicting the next interaction a user will make. Prediction has been discussed previously in Chapter 2, Section 2.4, as well as Chapter 4, Section 4.5. Once locations have been extracted, and interactions identified, next location prediction is considered using established techniques: *Support Vector Machines (SVMs)* and *Hidden Markov Models (HMMs)*, both

Table 5.5: Average area of extracted locations and identified land usage elements (both unrestricted and restricted areas). Results are shown for different values of d_{min}, with standard deviations shown in brackets.

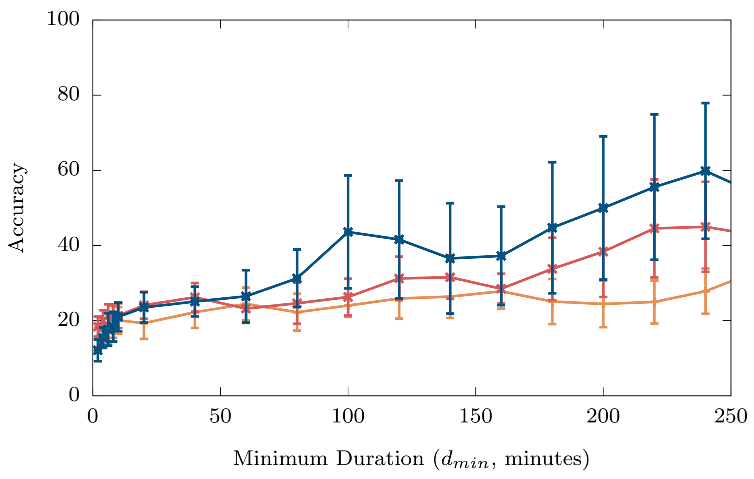
d_{min}	Land Usage	LU (Restricted)	Thresholding	GVE
2	66.2 (8.2)	42.5 (2.9)	46.0 (16.7)	45.9 (16.8)
4	69.1 (11.9)	43.1 (4.0)	43.4 (11.4)	43.8 (11.4)
6	63.9 (10.2)	39.4 (5.1)	41.5 (9.8)	41.6 (9.9)
8	61.7 (9.1)	42.5 (5.7)	41.6 (9.8)	42.0 (9.9)
10	63.9 (9.7)	42.6 (6.0)	42.0 (10.8)	41.9 (10.6)
20	76.1 (14.6)	40.0 (6.7)	40.3 (9.5)	40.3 (9.6)
40	81.2 (20.8)	40.3 (13.1)	40.1 (8.4)	40.4 (8.6)
60	94.0 (31.6)	50.5 (21.4)	42.2 (8.7)	42.9 (9.2)
80	81.5 (41.7)	49.2 (20.0)	44.8 (13.3)	44.9 (13.4)
100	100.6 (58.1)	57.8 (20.6)	37.5 (10.6)	37.5 (10.5)
120	72.8 (46.0)	57.1 (24.0)	38.0 (11.6)	38.1 (11.6)
140	50.6 (19.1)	58.4 (22.3)	35.3 (12.0)	34.9 (12.1)
160	76.8 (57.7)	55.6 (24.0)	34.1 (16.4)	34.1 (16.4)
180	30.0 (18.3)	48.6 (23.1)	34.0 (15.5)	34.1 (15.5)
200	67.1 (55.1)	55.1 (24.8)	29.5 (13.2)	29.8 (13.4)
220	64.4 (56.2)	46.7 (26.2)	31.6 (17.2)	31.5 (16.9)
240	73.2 (62.6)	61.5 (40.3)	28.0 (12.6)	28.2 (12.7)
260	72.7 (63.0)	62.6 (40.3)	29.9 (13.9)	30.3 (14.2)

of which have been demonstrated to achieve high predictive accuracies for this task [Akoush and Sameh, 2007; Bilurkar et al., 2002; Wang and Prabhala, 2012]. For both extracted locations and land usage elements, training instances must be generated. This is achieved by selecting interactions with locations or elements that last longer than d_{min} minutes. Instances are then generated by summarising interactions into a set of features: *day of year*, *day of week*, *start hour*, *start minute*, *duration*, *current identifier (element or location)*, and *class* (next identifier). The predictive models are then evaluated using 10-fold cross validation.

Figure 5.12 shows the accuracies obtained from performing prediction over extracted locations and identified land usage elements. The figure demonstrates that increasing d_{min} leads to predictions of higher accuracy, as d_{min} controls the minimum duration of an interaction to consider, with shorter interactions being ignored as noise. A larger value for d_{min} only considers locations and elements at which the user has spent significant amounts of time, thereby mak-



(a) Support Vector Machine (SVM).



(b) Hidden Markov Model (HMM).

Figure 5.12: The effect of minimum interaction duration, d_{min}, on predictive accuracy for existing location extraction and prediction techniques as well as the proposed LUI procedure for the Warwick dataset.

Table 5.6: Accuracy of different techniques for SVMs, with maximum values in bold and standard deviations in brackets.

d_{min}	**Land Usage**	**Thresholding**	**GVE**
2	31.5 (4.7)	**41.6 (8.8)**	27.6 (4.0)
4	35.4 (4.4)	**43.7 (8.9)**	29.5 (5.9)
6	39.0 (4.6)	**45.3 (9.1)**	30.5 (4.9)
8	42.7 (5.1)	**46.3 (8.9)**	33.2 (5.4)
10	44.6 (5.4)	**46.8 (8.9)**	31.1 (4.1)
20	**49.5 (7.0)**	47.2 (9.1)	34.8 (6.0)
40	**53.0 (8.2)**	49.1 (8.7)	38.0 (7.2)
60	**60.7 (9.5)**	51.7 (9.3)	40.2 (6.8)
80	**63.2 (9.6)**	54.6 (8.7)	41.5 (6.7)
100	**62.4 (11.3)**	57.0 (8.4)	38.3 (5.7)
120	**64.9 (11.9)**	60.4 (9.0)	42.2 (6.0)
140	**66.2 (12.2)**	62.8 (8.7)	42.4 (6.5)
160	**69.9 (10.3)**	64.5 (8.7)	42.8 (6.0)
180	**69.2 (14.2)**	65.4 (8.7)	42.6 (5.7)
200	**73.2 (14.1)**	67.8 (8.2)	38.2 (6.7)
220	**73.6 (14.4)**	68.2 (8.0)	45.5 (4.3)
240	**76.6 (11.7)**	70.4 (7.6)	41.8 (6.4)
260	**76.7 (12.0)**	71.9 (7.7)	48.9 (6.9)

Table 5.7: Accuracy of different techniques for HMMs, with maximum values in bold and standard deviations in brackets.

d_{min}	**Land Usage**	**Thresholding**	**GVE**
2	12.1 (2.9)	**18.4 (2.6)**	15.5 (2.8)
4	15.4 (2.7)	**20.0 (2.8)**	16.1 (3.0)
6	17.7 (4.3)	**21.3 (3.1)**	18.0 (4.2)
8	18.4 (3.9)	**20.9 (3.5)**	19.6 (4.2)
10	21.0 (3.8)	**21.3 (3.3)**	20.1 (3.5)
20	23.5 (4.0)	**24.1 (3.6)**	19.3 (4.2)
40	25.1 (3.9)	**26.2 (3.8)**	22.2 (4.2)
60	**26.5 (7.0)**	23.2 (3.7)	24.4 (4.3)
80	**31.3 (7.7)**	24.6 (5.5)	22.3 (4.9)
100	**43.6 (15.0)**	26.3 (4.9)	24.0 (3.0)
120	**41.6 (15.7)**	31.2 (5.8)	25.9 (5.3)
140	**36.6 (14.7)**	31.5 (5.7)	26.4 (5.7)
160	**37.2 (13.1)**	28.5 (4.0)	27.8 (4.6)
180	**44.7 (17.4)**	33.8 (8.3)	25.1 (6.0)
200	**50.0 (19.0)**	38.4 (12.1)	24.4 (6.2)
220	**55.5 (19.3)**	44.5 (13.0)	25.0 (5.7)
240	**59.8 (18.1)**	45.0 (12.0)	27.8 (6.0)
260	**53.7 (18.4)**	42.7 (14.3)	33.2 (5.4)

ing predictions more accurate with fewer possible locations the user will visit. This data is also shown in Tables 5.6 and 5.7, where it is clear that SVMs outperform hidden Markov models in all cases. Of most relevance, however, is the relative performance of the predictors operating over land usage elements when compared with those operating over extracted locations. For short interaction durations, extracted locations provide the foundation that affords more accurate predictions, but as d_{min} is increased beyond 20 minutes (for SVMs), extracted land usage elements provide the better foundation, as demonstrated by the higher predictive accuracies observed. Previously, in Figure 5.11, we demonstrated a reduced variance among average area of land usage elements when limiting the property during filtering to 400m^2. Using these restricted elements for prediction, Figure 5.13 demonstrates minimal impact compared to the elements identified without a maximum area limit (but with a limit to maximum radius), allowing for smaller elements without significantly impacting on predictive accuracy. Reducing the size of identified elements in this manner, to be closer to extracted locations, still yields higher predictive accuracies than predicting over the extracted locations for values of d_{min} over 20 minutes.

Additionally, the results in Figure 5.12 show that predictions over locations clustered from visits extracted from GVE perform worse than those extracted using thresholding. The reason for this is that GVE consistently extracts more locations than thresholding for the same value of d_{min}, as shown in Figure 5.8b, leading to reduced predictive accuracy as the set of possible locations is increased. Finally, Figure 5.14 and Table 5.8 show the results for performing prediction over the MDC dataset, where the same general trends are observed.

5.5 Summary and Conclusion

This chapter has extended the idea of extracting significant locations from trajectories by augmenting the trajectories with land usage elements. The Land Usage Identification (LUI) procedure presented is capable of identifying which

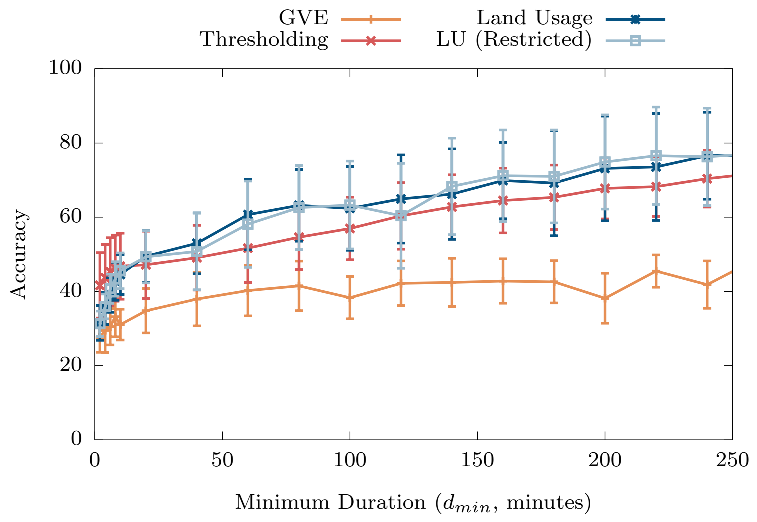


Figure 5.13: The effect of minimum interaction duration, d_{min}, on predictive accuracy for the restricted land usage elements and existing location extraction and prediction techniques, specifically the SVM, over the Warwick dataset.

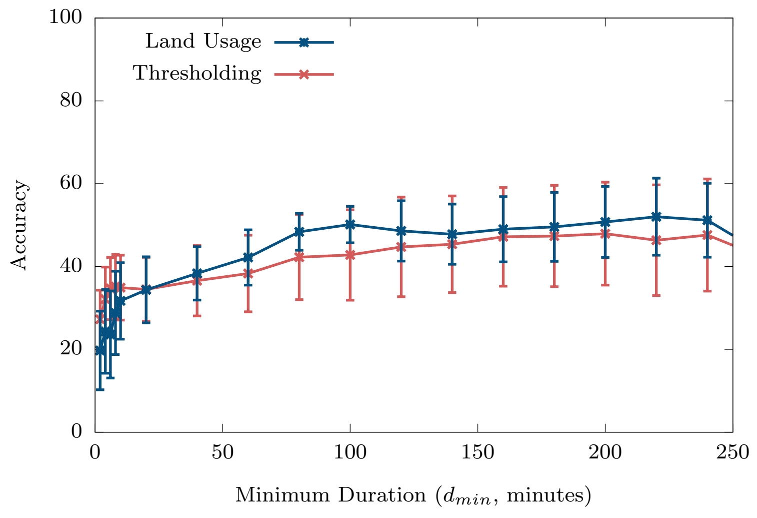


Figure 5.14: Predictive accuracies for locations extracted with thresholding, and land usage elements identified through the LUI procedure, when predicting with SVMs over the MDC dataset.

Table 5.8: Accuracy of different techniques for SVMs over the MDC dataset, with maximum values in bold and standard deviations in brackets.

d_{min}	**Land Usage**	**Thresholding**
2	19.7 (9.5)	**27.3 (7.0)**
4	24.3 (10.1)	**32.1 (7.7)**
6	23.6 (10.5)	**34.7 (7.5)**
8	28.8 (10.0)	**35.2 (7.8)**
10	31.7 (9.2)	**34.9 (7.8)**
20	34.4 (8.0)	**34.5 (7.7)**
40	**38.4 (6.5)**	36.6 (8.5)
60	**42.2 (6.7)**	38.3 (9.2)
80	**48.4 (4.5)**	42.2 (10.2)
100	**50.1 (4.4)**	42.8 (10.9)
120	**48.6 (7.3)**	44.7 (12.0)
140	**47.8 (7.3)**	45.4 (11.7)
160	**49.0 (7.9)**	47.2 (11.9)
180	**49.6 (8.3)**	47.3 (12.2)
200	**50.8 (8.6)**	47.9 (12.4)
220	**52.0 (9.3)**	46.4 (13.4)
240	**51.2 (8.9)**	47.6 (13.5)
260	**43.8 (10.9)**	42.7 (14.4)

land usage element a person was likely to have been interacting with, and summarising these interactions. The resultant output is a set of interactions and elements that are consistent with the visits and locations identified by existing approaches, but have a far greater relationship with the real world. Not only do these elements represent geographic features, but the elements also contain information that can be used as a basis for understanding what a person may have been doing.

The Land Usage Identification (LUI) procedure is evaluated through an exploration of the properties of identified elements and interactions, and through a sample application, that of predicting future interactions. Through this application, we demonstrate increased predictive accuracy when compared to predictions made over locations, using established predictive techniques. These evaluations help to demonstrate the utility of the LUI procedure and the interactions and elements it identifies. The interactions identified through the LUI procedure go on to form the basis for determining contexts in Chapter 6.

CHAPTER 6
The Context Tree

The work in Chapter 5 considers augmenting geospatial trajectories with land usage information to identify the real-world feature that an individual, or other entity, interacted with. These interactions are then summarised into a form consistent with existing work that identifies locations from solely geospatial trajectories. The utility of the procedure was demonstrated through a sample application, that of predicting the next interaction the user would make.

The identified elements and their interactions offer a wealth of information that is not present when identifying locations from trajectories alone, such as the shape and properties of the real-world feature being interacted with. This chapter focuses on leveraging the additional information to understand the context behind user actions. Specifically, the chapter presents and evaluates the *Context Tree*, a new hierarchical data structure that identifies and summarises the context behind user actions in a single model. Through an exploration of the properties of the generated trees, and the outputs of different stages of the proposed generation procedure, we demonstrate the foundation for understanding and modelling behaviour afforded by this model. Summarising user contexts into a single data structure gives easy access to information that would otherwise remain latent, providing a basis for better understanding and predicting the actions and behaviours of individuals and groups.

6.1 Introduction

Exposing the latent knowledge present in geospatial trajectories has become an increasingly important research topic in recent years, due in part to the pervasiveness of location-aware hardware and the resulting availability of trajectory data. Motivated by a desire to understand the movement patterns of users, this chapter presents a new data structure, the *Context Tree*, that uses the augmented trajectories from Chapter 5 as a basis for identifying and summarising the context behind user actions in a single hierarchical model. Using augmented trajectories places a reduced burden on users than in existing techniques for context identification, while providing vast amounts of information about what a person may have been interacting with in the physical world. Clusters are identified hierarchically and stored in a model that affords easy access to the information.

While existing work has considered identifying contexts by analysing attributes such as location, social interaction histories, and sensor readings, they are limited by the data that can be collected directly from each user. Employing augmented geospatial trajectories as a basis for context extraction allows for contexts to consider the properties of features that people interact with without collecting additional data from users directly. This chapter proposes and evaluates techniques for identifying contexts from augmented trajectories and summarising these contexts into a single model, namely the Context Tree. Towards these goals, Section 6.2 provides an overview of related work in the area of context extraction and geospatial systems. The Context Tree data structure itself, along with the generation procedure, is presented in Section 6.3, with an exploration of properties of constructed trees and a comparison to partial ground truths in Section 6.4. The chapter concludes with a summary and discussion in Section 6.5. Chapter 7 then goes on to illustrate the use of Context Trees as a basis for constructing a predictive model to forecast the future actions of individuals.

6.2 Related Work

Understanding people from data by identifying the activities they have performed, and the contexts they have been immersed within, is an area of research that has received increased focus in recent years. Chapter 2, Section 2.5, provides a discussion of the current state of research in this area, with relevant topics summarised here.

Identifying activities performed by individuals has previously been investigated using video data [Brand et al., 1997; Kim et al., 2010; Messing et al., 2009; Morris and Trivedi, 2011]. Although this provides a foundation for understanding what a person was doing, ensuring the constant availability of video footage is infeasible, and so attention has turned to data that can be collected by devices carried by the individual. Sensors such as heartrate, Global Positioning System (GPS) and accelerometers have been used to collect data as a basis for classifying the activity being performed [Choudhury et al., 2008; Lester et al., 2005; Pirttikangas et al., 2006; Subramanya et al., 2006; Van Kasteren et al., 2008], with the collected data being used to train models as a supervised learning problem. Typically, the approaches employed are not concerned with identifying when activities begin and end, but rather focus on classifying a *current* activity from a set of known labels.

To reduce the requirement for known activities to be used in training models, context identification aims to cluster periods of time in which a person has similar goals or intentions, regardless of whether the specific goal is known. This too has been considered from data generated by devices carried by the user [Bao et al., 2011; Lemlouma and Layaida, 2004], but also by categorising the contexts offered by locations known to have been visited by the user [Assam and Seidl, 2014]. These approaches have their own merits and drawbacks, as they typically focus on a single context at a time and require specific data to be collected from the users. In reality, users may be immersed within multiple contexts or have multiple goals simultaneously, where one context may be a subset of another (e.g. a specific type of task being conducted under the general context of ‘at work’). The work in this chapter aims to reduce the onus placed on individuals by requiring only the collection of geospatial trajectories, which can be collected actively or passively, and identifying such a hierarchy of contexts.

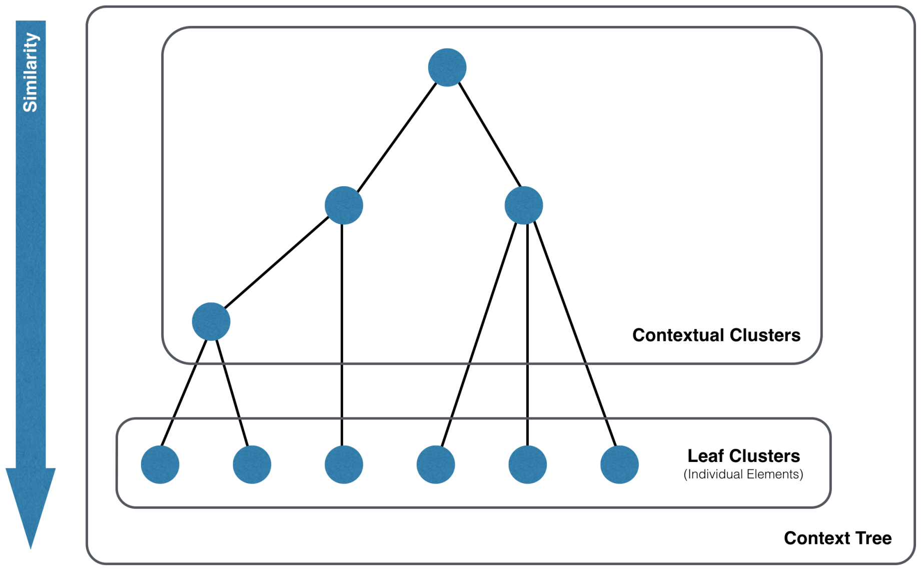


Figure 6.1: An abstract representation of a Context Tree, in which the similarity of nodes increases with depth.

6.3 Identifying Contexts Through Clustering

This chapter proposes and evaluates the *Context Tree* hierarchical data structure that summarises the contexts that a user has been immersed within at multiple scales. Each leaf node of the tree represents a real-world feature or element that the user is likely to have interacted with, be it a specific building, road, area, or item (e.g. a bench in a park). These individual elements are joined together through *context nodes* that represent a context at a specific scale, where time spent within a context means that the user is likely to have had similar aims or goals, with the root node being the highest scale, encompassing all other contexts. In a similar manner to many other clustering tasks, the act of identifying contexts is separate to that of labelling contexts. For this work, we are only interested in the identification of groups of related element interactions to form contexts, but the problem of assigning labels to contexts is left as future work, as such labels are not required for many applications, for example prediction, which is considered in Chapter 7. The contexts are identified by determining periods of time the user spends interacting with elements with similar properties, or elements that are interacted with in a similar manner. As it summarises time in this way, the Context Tree, depicted in Figure 6.1, can become the basis for understanding people from augmented geospatial data.

Generating a Context Tree uses augmented geospatial trajectories as proposed in Chapter 5, in addition to a clustering procedure that converts these trajectories into a useful structure. The overall method of construction of a Context Tree takes raw geospatial trajectories and land usage data as input, and consists of the following four stages, as depicted in Figure 6.2:

1. **Augmentation**

   Land usage elements likely to have been interacted with are identified by extracting all potential elements and filtering them to remove noise (as presented in Chapter 5). These trajectories are then summarised to detect interactions.

2. **Clustering**

   Filtered land usage elements and their interactions become the basis for contextual clustering. Clustering is achieved with a hierarchical agglomerative algorithm.

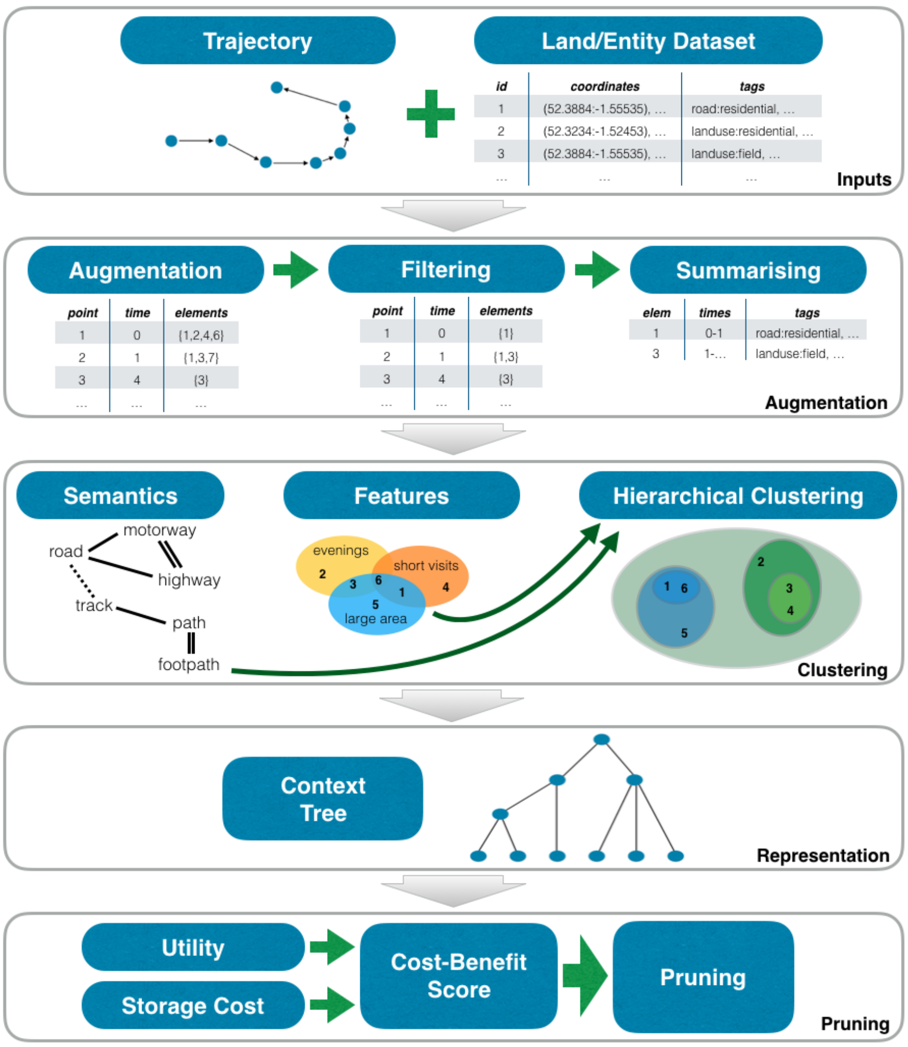
3. **Representation**

   Once clustered, the elements form a Context Tree data structure that can be used as the basis for further understanding the behaviour of individuals and groups.

4. **Pruning**

   Some applications may be limited by the amount of data they can store, or processing they can perform, so it may be necessary to prune a Context Tree to reduce its size while maintaining as much useful information as possible. Pruning is achieved through analysing the nodes of a Context Tree with respect to a defined set of metrics.

The remainder of this section presents techniques to achieve each stage in this procedure.



id	*coordinates*	*tags*
1	(52.3884:-1.55535), …	road:residential, …
2	(52.3234:-1.52453), …	landuse:residential, …
3	(52.3884:-1.55535), …	landuse:field, …
…	…	…

point	*time*	*elements*
1	0	{1,2,4,6}
2	1	{1,3,7}
3	4	{3}
…	…	…

point	*time*	*elements*
1	0	{1}
2	1	{1,3}
3	4	{3}
…	…	…

elem	*times*	*tags*
1	0-1	road:residential, …
3	1-…	landuse:field, …

Figure 6.2: Overview of the Context Tree generation framework. A trajectory is augmented with land usage data; this augmented data is then hierarchically clustered into a Context Tree. Subsequently the Context Tree can optionally be pruned.

6.3.1 Augmentation, Filtering, and Summarisation

The output of the augmentation, filtering, and summarisation procedures outlined in Chapter 5, Section 5.3, is a set of land usage elements. These elements represent geographic features and have associated information including coordinates, tags, and summarised interactions that describe the time the user spent interacting with each element. The work in this chapter uses these summarised trajectories as a basis for understanding behaviour; however, to achieve this we extend the filtering and summarisation procedures used previously. In Chapter 5, we identify the single land usage element that the user was most likely interacting with, so the filtering process selects a single element smaller than a specified size (Chapter 5, Section 5.3.2). For the process of Context Tree construction, we relax these requirements by allowing multiple elements of arbitrary size to be assigned to each trajectory point. When considering clustering contexts at multiple scales, it is reasonable to assume that a person may be interacting with a hierarchy of land usage elements, for example a shop in a shopping centre or a specific building on a university campus. By removing the maximum element size, and allowing multiple land usage elements to be associated with each point, we allow the identification of such hierarchies.

As part of the filtering procedure shown previously, each element in a buffer is assigned a score. Selecting elements can therefore be performed in several ways, depending on the desired outcome:

- Elements with a score above some threshold, t, can be assigned to the *point under consideration*.
- Soft-thresholding, where a kernel function is employed to redistribute scores, can be used to keep all elements.
- The n elements with the highest scores can be kept.

For this work, we are not concerned with element scores once filtering has been completed, so we opt to select n elements to associate with each trajectory point. Of course, with more than one element now associated with each

Algorithm 4 Modified summarisation procedure for overlapping interactions.

```
trajectory ← ($p_1, p_2, ...$) // augmented trajectory
$t_{max}$ ← 5 // maximum time between consecutive points (minutes)
$d_{min}$ ← 10 // minimum visit duration (minutes)
elements ← ElementStore // store of elements and their interactions
ongoing ← {} // stores start time of ongoing interactions until they are ended
previousTimestamp ← p1.timestamp

while currentPoint ← trajectory.shift do
    // If too much time has passed between points, end all ongoing interactions
    if (currentPoint.timestamp - previousTimestamp) > $t_{max}$ then
        toEnd ← ongoing
        toStart ← currentPoint.elements
    else
        toEnd ← (ongoing - currentPoint.elements) // End finished interactions
        toStart ← (currentPoint.elements - ongoing) // Start new interactions
    end if

    // Store interactions that are long enough
    while element ← toEnd.pop do
        if (previousTimestamp - ongoing[element]) > $d_{min}$ then
            interaction ← {start: ongoing[element], end: previousTimestamp}
            elements.addInteraction(element, interaction)
        end if
        ongoing.delete(element)
    end while

    // Mark the start time of new interactions
    while element ← toStart.pop do
        ongoing[element] = currentPoint.timestamp
    end while

    previousTimestamp ← currentPoint.timestamp
end while

return elements
```

point, the summarisation procedure must be modified to handle overlapping interactions. Algorithm 4 shows an extended version of Algorithm 3, that has the capability of dealing with overlapping land usage interactions.

6.3.2 Building Clusters

The identification of similar *contexts* is performed through clustering that considers both the properties of the elements and the properties of user interactions to determine similarity. Rather than aiming to identify a single level of clusters, which would limit the utility and applicability of the clusters to a single scale, the goal here is to build a hierarchical model, constructed by progressively merging land usage elements that represent similar contexts in a Context Tree, a depiction of which is shown earlier in Figure 6.1.

Initially, each land usage element is distinct and is treated as a singleton cluster (i.e. a cluster with exactly one element). In each round of clustering, several of these clusters are merged to represent a context and a new higher level in the hierarchy, with pointers between the levels considered as *parent* and *child* relationships. That is, if two clusters at one level become merged into another cluster at the next level, the original clusters are considered as *children* of the new cluster. This section describes how clusters are merged with respect to their properties.

As discussed in Chapter 5, Section 5.3, land usage elements are assumed to have a set of *tags* in the form of 'key:value' pairs that describe properties of the real-world entity to which the element relates, in addition to *geographical coordinate sets* that describe the geographical properties of the real-world entity. Once augmented and summarised, these elements are also associated with a set of *interactions* consisting of *times* the user interacted with each element. When clusters are merged to create a Context Tree, the following procedures are used:

Times

The times for the merged cluster are taken to be the union of the sets of times from all child clusters, where overlapping time ranges are themselves combined into one. For example, if one cluster had the set of times {10:00-10:05, 11:00-12:00} and another had {10:04-10:20, 11:10-11:15, 12:05-12:09}, then the merged times would be {10:00-10:20, 11:00-12:00, 12:05-12:09}.

Tags

Similarly, each element has associated tags. The tags of the merged cluster are defined as the union of tags from the child clusters, where if two tags share a key but not a value, both values are stored. For example, 'recreation:park, access:public' would be merged with 'recreation:pond' to form 'recreation:park,pond, access:public'.

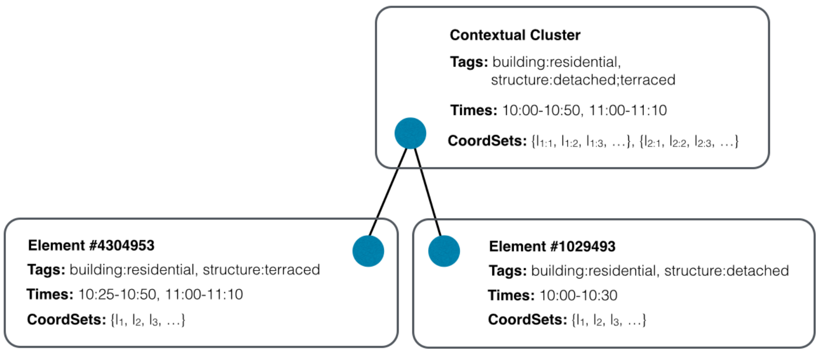

Contextual Cluster
Tags: building:residential, structure:detached;terraced
Times: 10:00-10:50, 11:00-11:10
CoordSets: $\{l_{1:1}, l_{1:2}, l_{1:3}, \ldots\}, \{l_{2:1}, l_{2:2}, l_{2:3}, \ldots\}$
Element #4304953
Tags: building:residential, structure:terraced
Times: 10:25-10:50, 11:00-11:10
CoordSets: $\{l_1, l_2, l_3, \ldots\}$
Element #1029493
Tags: building:residential, structure:detached
Times: 10:00-10:30
CoordSets: $\{l_1, l_2, l_3, \ldots\}$

Figure 6.3: An example of how clusters are merged together when generating Context Trees.

Geographical Coordinate Sets

Each element contains a set of coordinates that define the geographical shape of the entity to which they relate. Merging such elements should keep each of these sets discrete, unless they intersect, in which case the coordinates belonging to both shapes are combined and replaced with their convex hull. This is shown in Figure 6.3.

The merging of *times* assumes a periodicity of 24 hours, which while reasonable for many people (i.e. those who follow a daily routine), it may not be appropriate for everyone. As such, automatic time series learning could be utilised to improve the learning of meaningful movement patterns of the individual. While exploring such techniques is beyond the scope of this work, there are many existing approaches that may be effective for the task, as discussed in [Ahmad et al., 2004]. An example merging of two elements according to these rules is shown in Figure 6.3, where it is assumed there is no geographical overlap between the two elements (i.e. the coordinate sets cannot be merged).

6.3.3 Contextual Distance Metrics

Clustering elements together requires a distance metric to measure element similarity. While identifying contexts from certain types of data is a task considered before, and discussed in Chapter 2, no metrics currently exist that have been tailored to the identification of contexts from augmented geospatial trajectories. This section presents metrics that encapsulate the goals behind context extraction for this specific problem, with an emphasis on properties of the interactions and properties of the geographic features being interacted with. Having defined how elements are merged into clusters and, consequently, how two clusters are merged (Section 6.3.2), we can now consider the similarity between two clusters.

Semantic Similarity

Clusters have tags that describe properties of the real-world entities contained in the cluster, forming an ideal basis for understanding what the user might have been doing. Under the assumption that clusters with similar tags are likely to have properties in common, we use the semantic similarity between cluster tags as the basis for a distance metric. For this, we adopt the similarity measure proposed by Wu and Palmer [1994], and extended by Resnik [1999] for calculating distance between word taxonomies through WordNet [Miller, 1995]. The calculated scores are between 0 and 1 (inclusive), where a score of 1 means that the words are interchangeable. The semantic similarity between two sets of tags, t_1 and t_2, is therefore calculated as:

$$TagSim(t_1, t_2) = \frac{\sum_{a \in t_1} \max \forall_{b \in t_2} Sim(a, b)}{|t_1|} \tag{6.1}$$

As tag similarity is not commutative, cluster similarity is calculated as:

$$SemanticSimilarity(c_1, c_2) = \max(TagSim(c_1.tags, c_2.tags), TagSim(c_2.tags, c_1.tags)) \tag{6.2}$$

Feature Similarity

The context of an activity or period of time is dependent not only on the location in which time is spent, but on additional factors. With this in mind, we propose a second similarity measure, *FeatureSimilarity*, that compares the interaction features of two clusters, specifically:

- average interaction duration,
- most common time of day interaction begins,
- count of the number of times the element is interacted with, and
- total area covered by elements (in m^2).

The value from each feature is then discretised by placing values within bins (e.g. time of day could be recorded in 4 hour increments), and converted into a single string that describes the feature and value (e.g. 'timeofday_12' would indicate that the most common time of day that interaction begins is between 12PM–4PM). This procedure generates a set of features, f_1 and f_2, for clusters c_1 and c_2, from which a similarity score is defined using the Jaccard index [Rajaraman and Ullman, 2011]:

$$FeatureSimilarity(f_1, f_2) = \frac{|f_1 \cap f_2|}{|f_1 \cup f_2|} \tag{6.3}$$

Geographical Distance

For some applications it is possible that the similarity between clusters depends upon their geographical proximity, where two clusters that are close together may have common purposes. If this property is known to be true in the data, or given the goal of clustering, then the proximity of clusters can be considered as the minimum geographical distance between elements of a cluster, calculated using the Haversine formula [Robusto, 1957]:

$$GeographicalDistance(c_1, c_2) = \min \forall_{a,b \in c_1 \times c_2} D_s(a, b) \tag{6.4}$$

Hybrid Contextual Distance

Using one of the previously discussed metrics in isolation would not accurately capture the context of the individual, as context depends on more than just any one factor. Instead, we combine the *SemanticSimilarity* and *FeatureSimilarity* scores into *Hybrid Contextual Distance (HCD)*, a measure of the contextual similarity between two clusters:

$$\begin{aligned} HCD(c_1, c_2) = 1 - (\lambda\ SemanticSimilarity(c_1, c_2) \\ + (1 - \lambda)\ FeatureSimilarity(c_1, c_2)) \end{aligned} \tag{6.5}$$

Where λ is a user-specified weighting parameter that allows emphasis to be placed either on the semantic or feature similarity between clusters.

We choose to ignore the geographical proximity of elements, and therefore the geographical distance metric, because contexts should be separate from their location (e.g. visiting two cafes in different cities is likely to be indicative of the same context). If, however, additional domain knowledge is available that ties geographical locations together with enhanced meaning (e.g. it is known that all buildings in a given area perform a similar function), then geographical distance could be added to the HCD metric. HCD can be used as a basis for clustering elements, thus determining which elements have similar contexts, aiding in our understanding of the individual to which the data belongs.

6.3.4 Hierarchical Clustering

With a distance metric in place, clustering can be performed using standard techniques. While traditional clustering is limited in that it extracts clusters at a single scale, which may not be appropriate for a given task, hierarchical clustering identifies clusters at multiple scales. We use a greedy hierarchical agglomerative clustering algorithm, presented in Algorithm 5, that extracts clusters of increasing similarity up to a single root node, creating a *Context Tree*. While the algorithm is fairly standard in itself, its application to the generation of Context Trees is novel. The algorithm deviates slightly from existing hierarchical clustering approaches in that it is capable of extracting multiple clusters together in a single step if they have the same distance.

Algorithm 5 Agglomerative hierarchical clustering algorithm.

```
clusters ← elements // The input set of elements, each treated as its own cluster
while clusters.length > 1 do

    // Create an $n \times n$ matrix of distances between clusters
    distanceMatrix ← [ [$d_{11}$, ...], [$d_{21}$, ...], ...]

    // Find all pairs of clusters with the smallest distance between them
    // If multiple pairs overlap (i.e. share a cluster), then group them together
    closestGroups ← ClosestGroups(distanceMatrix)

    // Merge each extracted group into a single cluster
    for group ∈ closestGroups do
        newCluster ← Merge(group)

        // Set the old clusters as children of the new and remove the old clusters
        for cluster ∈ group do
            newCluster.children.append(cluster)
            clusters.delete(cluster)
        end for

        // Add the merged cluster to clusters
        clusters.append(newCluster)
    end for

end while

// By this point, clusters contains a single root cluster for the hierarchy
return clusters.first
```

6.3.5 Context Tree Pruning

Storing Context Trees in their entirety maintains the maximum amount of information; however, there are applications where reducing the size of a tree may be desirable. Memory-constrained devices, for example, may be better able to make use of a reduced size Context Tree as this would require lower storage requirements, and also enable quicker search due to the reduced number of nodes. Furthermore, reducing the size of Context Trees may have application-specific benefits, such as preventing overfitting when learning prediction models. In both of these cases, it is desirable to *prune* the tree to reduce the amount of data stored while maintaining as much information as possible. This section presents a method for such pruning, that although requiring additional processing to select nodes eligible to be removed, results in smaller Context Trees that require less memory to store and fewer operations to search. A representation of a pruned Context Tree can be seen in Figure 6.4.

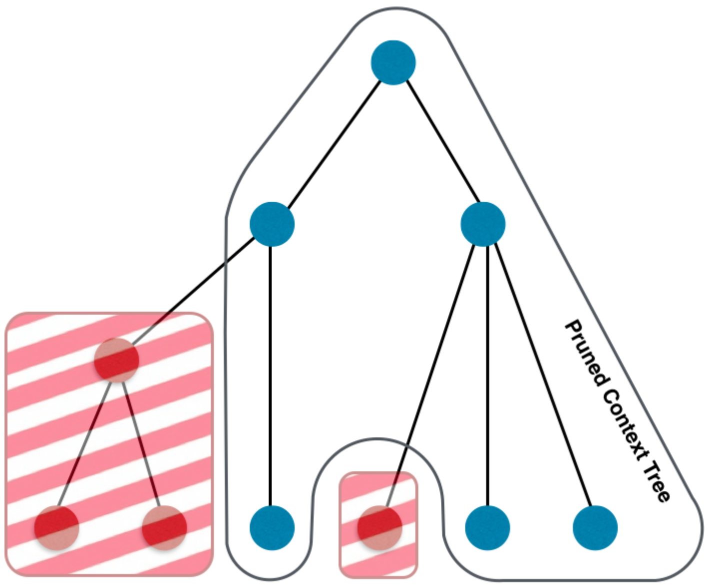


Figure 6.4: An example of a pruned Context Tree (with removed nodes crossed through).

Pruning is performed depth-first, evaluating each cluster to determine whether the additional overhead of storing the node is outweighed by the utility afforded. Clusters are considered using the *null hypothesis*, and the hypothesis rejected when the utility of storing the cluster is above a threshold. Any cluster for which we are unable to reject the hypothesis is pruned, and its parent is marked as eligible for pruning. As metrics do not already exist for this task, we adapt existing metrics used in related domains for the purpose of Context Tree pruning.

Storage Cost

Clusters are scored according to two metrics: their storage cost and their utility. To determine the cost of storing a cluster, it is important to understand how clusters are built up in a Context Tree (described in Section 6.3.2). When merging two clusters together to form a parent cluster, the aspects that belong to each cluster are considered in turn: specifically the *tags*, *times* and *coordinate sets*. Sets of tags are combined from the child clusters by taking their union, while times and coordinate sets are merged in such a way that overlapping components are combined into single elements, thus through the combining of child clusters into a parent cluster, information has been removed. The cost of storing an additional node is therefore the cost of storing the individual components (e.g. time range) that are present in a child, but not present in the same form in its parent. Assuming equal cost to store each component:

$$\begin{aligned} Cost(C|P) = \xi + |C_{times} \setminus P_{times}| \\ + |C_{coordsets} \setminus P_{coordsets}| + | \cup_{s \in C_{coordsets}} s \setminus \cup_{s \in P_{coordsets}} s| \end{aligned} \quad (6.6)$$

Where $\xi > 0$ is a small, manually selected, penalty that represents the overhead of storing each cluster, C_{times} and P_{times} are the sets of time ranges that are associated with clusters C and P, and $C_{coordsets}$ and $P_{coordsets}$ are the sets of coordinate sets associated with clusters C and P. Remembering that the coordinate sets belonging to a cluster themselves contain sets of points (i.e. $C_{coordsets} = \{\{p_{1:1}, p_{1:2}, p_{1:3}, ...\}, \{p_{2:1}, p_{2:2}, p_{2:3}, ...\}, ...\}$), $\cup_{s \in C_{coordsets}} s$ is taken to be the set of all points associated with any coordinate set that belongs to cluster C. Having ξ as non-zero represents that there is always a small cost associated with each cluster. Equation 6.6 must be tuned for the specific application to better represent the true cost of storing a node, but it provides a basic foundation.

Cluster Utility

Determining the utility of a cluster is difficult and is dependent on the specific use of the Context Tree. For this reason, any application of the approach will need to consider the goal of pruning and use this to inform the measurement of the utility afforded by a specific cluster. We adopt a general approach that can be tailored to specific needs by providing a measure of the information lost if the parent of a cluster were used to represent the child, similar in idea to the *Kullback-Leibler divergence* used to measure the difference between probability distributions. As parents contain a superset of the children, we consider the utility of a child cluster (C) given its parent (P) to be the proportion of data present in the parent that is not covered by the child, where the measure of data must consider the attributes (i.e. tags, times, and coordinate sets) present in each cluster:

$$Data(C) = \textstyle\sum_{t \in C_{times}} duration(t) + \sum_{s \in C_{coordsets}} area(s) + |C_{tags}| \tag{6.7}$$

Providing even weighting to the different elements for the measure of utility:

$$Utility(C|P) =$$
$$1 - \left(\frac{1}{3} \frac{\sum_{t \in C_{times}} duration(t)}{\sum_{t \in P_{times}} duration(t)} + \frac{1}{3} \frac{\sum_{s \in C_{coordsets}} area(s)}{\sum_{s \in P_{coordsets}} area(s)} + \frac{1}{3} \frac{|C_{tags}|}{|P_{tags}|} \right) \tag{6.8}$$

Specifically, this metric considers the proportion of time, area and tags covered by the child with respect to the parent, and holds true to the aims of such a metric to produce a score of 0 if the parent and child contain identical information and a score approaching 1 if the child only represents a fraction of the parent.

Cost-Benefit Score

The *cost-benefit score* of a child cluster given its parent is taken to be the utility of the cluster divided by the storage cost:

$$CostBenefitScore(C|P) = \frac{Utility(C|P)}{Cost(C|P)} \tag{6.9}$$

While utility is normalised between 0 and 1 as it represents the proportion of the parent that is not covered by the child, cost only has a minimum bound of ξ, where $\xi > 0$. Depending upon the application, it may be desirable to also normalise cost relative to the current Context Tree. Using this metric on nodes in a depth-first manner, pruning should occur for any cluster C with parent P and $CostBenefitScore(C|P) < \theta$, where θ is the *pruning threshold* and C has no unpruned children.

6.4 Case Study

In this work it is not practical to obtain a concrete ground truth to act as a point of comparison for evaluating complete Context Trees because the *correctness* of an extracted set of clusters depends on the task for which the clusters will be used. In light of this, our evaluation of the proposed techniques follows a similar approach to those used in existing literature where a single ground truth does not exist, as discussed in Chapter 2, Section 2.1.4. This is achieved by exploring the properties of the generated Context Trees and comparing them against expected results while providing small, representative, examples that demonstrate the utility afforded by these procedures.

This section provides a case study of the proposed Context Tree data structure, along with the modified filtering procedure (Section 6.3.1) and the clustering and pruning procedures (Sections 6.3.2 and 6.3.5). Although there are many use cases for Context Trees, including as a basis for anomaly detection, location prediction, and city planning, we focus on understanding the high-level behaviour of an individual throughout a 24 hour period as a representative example. The evaluation here is continued in the following chapter where Context Trees are converted into a hierarchical classification model capable of predicting both future contexts and future interactions of individuals, thereby further demonstrating the utility afforded by the structure evaluated here.

6.4.1 Methodology

Exploring the Context Tree uses augmented geospatial trajectories as a basis for understanding past actions. For this purpose, we use the same trajectories as in the previous chapter, detailed in Section 5.4.1. These trajectories are augmented, filtered, and summarised using the procedure outlined in this and the previous chapter. From the data generated through this process, Context Trees can be constructed. A portion of a Context Tree generated from the same data as in Figure 5.2 (Chapter 5) can be seen in Figure 6.5, where the example

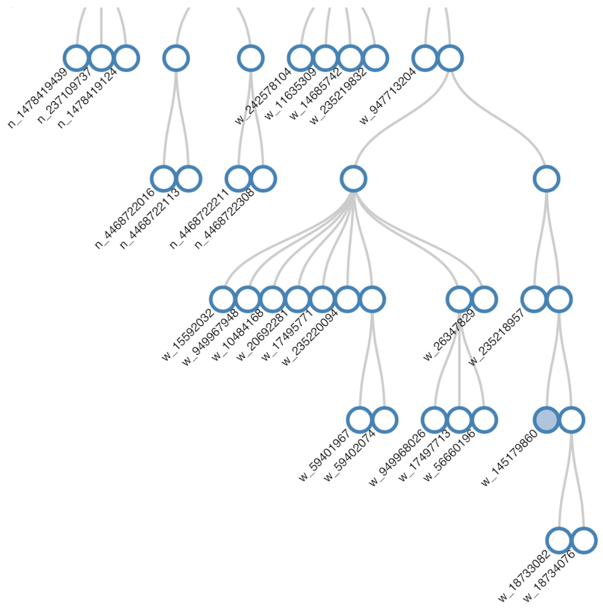


Figure 6.5: An extract of a Context Tree generated using real data.

element ('w_145179860') from the sample data is highlighted with a solid blue circle.

Data

The data used for evaluating Context Trees is the same as used for evaluating the Land Usage Identification (LUI) procedure in Chapter 5; specifically 10 users from each of the Nokia Mobile Data Challenge (MDC) and Warwick datasets are used in addition to land usage data from OpenStreetMap (OSM). In Chapter 5, we limit the size of OSM elements selected for inclusion in the augmentation procedure through the parameter *maxradius*. For this chapter, however, we do not impose such a limit, to allow the identification of a hierarchy of features (e.g. a building and the campus it resides on). However, OSM includes several designations that go beyond our definition of a land usage element, namely the boundaries of towns, counties, countries, etc. In the previous chapter, these would have been ignored due to their size, but for this work instead we remove these elements from the dataset during the augmentation process[1].

In addition to information about each element, OSM contains several *meta* tags that relate to the dataset, but not specifically the real-world feature. Generating Context Trees uses these tags heavily to determine semantics of the element, so we remove such metadata when generating augmented trajectories[2].

6.4.2 Filtering and Summarising

The expanded filtering procedure presented in this chapter enables multiple land usage elements to be associated with a single trajectory point. To understand the effectiveness of this procedure, we explore properties of the filtered data, specifically focusing on how the elements and their semantics change. The aim of filtering is to remove noise and focus the data on elements that the user was likely to have been interacting with at a given time. It is reasonable therefore to assume that the elements post-filtering should have more similarity than those before, with less variation caused by the inclusion of *random* elements. To explore this hypothesis, Figure 6.6 shows the average tag key similarity (i.e. only the key part of the 'key:value' pair that makes up an element's tags, which corresponds to broad type, e.g. 'building') both pre- and post-filtering for a sample user over 1,000 points of their data. Semantic similarity here is calculated using the same method as discussed in Section 6.3.3. The figure demonstrates that in the majority of cases, tag key similarity is increased after filtering has occurred, specifically going from mean 0.089 (standard deviation: 0.036) to mean 0.187 (standard deviation: 0.156). These results indicate that the elements present post-filtering are more similar and that unrelated noise elements have been removed correctly.

Additionally, Figure 6.7 shows the relationship between n, the maximum number of elements associated with each trajectory point and the number of

[1]Specifically, we remove OSM *relations* that have either 'boundary:administrative' or 'boundary:ceremonial' as one of their tags.

[2]Tags that begin with the following strings are removed: *source*, *wikipedia*, *note*, *name*, *alt*, *created*, *fixme*, *todo*, *website*, *phone*, *layer*, *url*.

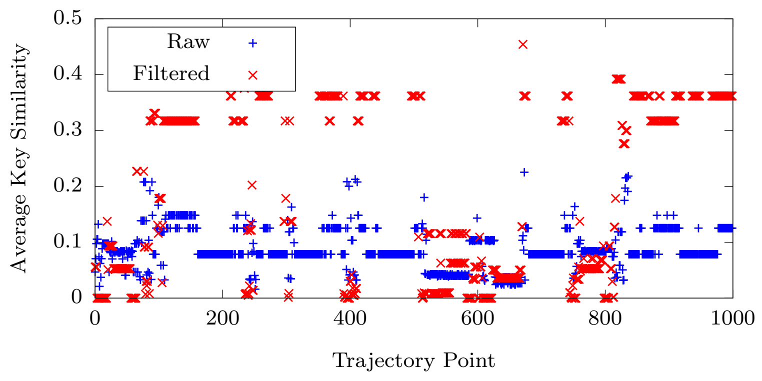


Figure 6.6: The effect of element filtering on tag key similarity from a single user's data, both pre- and post-filtering.

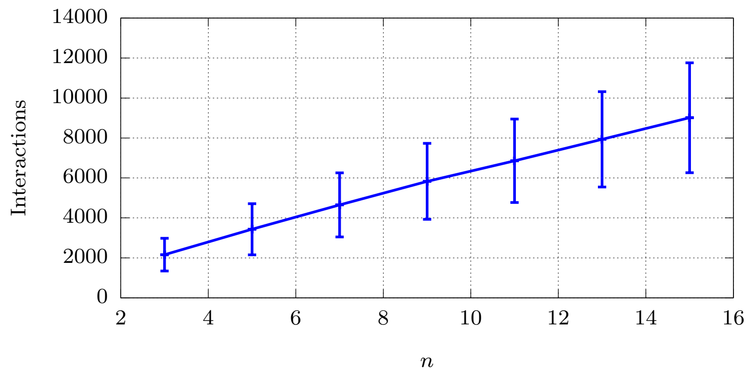


Figure 6.7: The effect of n on number of land usage interactions identified when constructing Context Trees.

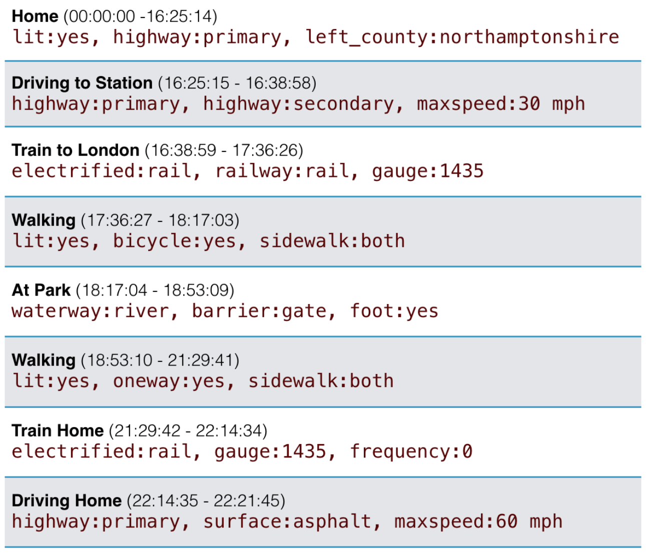
individual interactions identified, averaged over all users of both datasets. The trend present is as would be expected, with an increase in n resulting in more interactions being identified as each additional element extracted has additional interactions. For the remainder of this section, unless specified otherwise, we set $n = 5$ as it produces representative results.

User-informed Evaluation

To understand the applicability of using identified land usage elements as a basis for understanding the actions of people, we turn our attention to exploring how well the identified elements represent what users were actually doing. While there is no ground truth available for this type of problem, we can evaluate the procedure by considering desirable properties of the output for a specific application and manually compare the expected and actual results for small subsets of data. For this, we are unable to use the MDC data because contacting the users who provided data is not possible, but we can use the Warwick dataset, where we have the ability to converse with participants. By focusing on identified land usage elements instead of extracted locations, we can consider time that the user spent moving as part of a user's context.

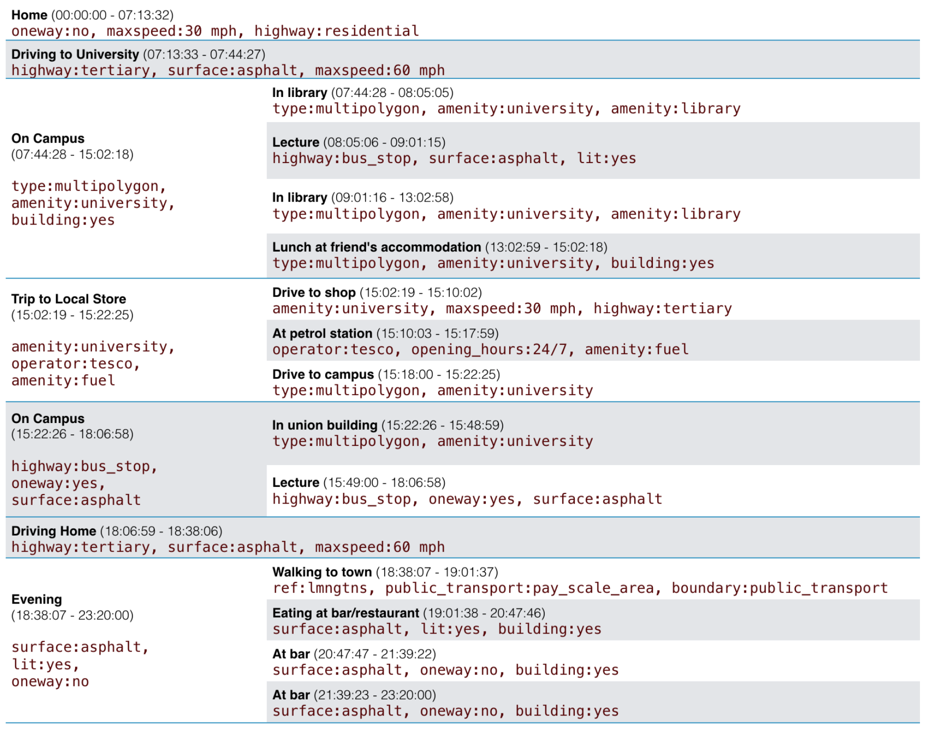
This section presents analyses on small amounts of manually labelled real-world data with the aim of understanding the types of feature a person interacts with. The data analysed spans 24 hours from 3 of the 10 users of the Warwick dataset, where annotations were added manually as accurately as possible, and in consultation with the users. The augmentation and filtering procedures were run over this data with parameters $\delta = 10$, $n = 5$, and, for each labelled time period, the 3 most common element tags were identified. This is shown in Figures 6.8-6.10. The aim here is not to label the time periods with the exact activity being performed, but rather to demonstrate that a meaningful relationship exists between the tags extracted and the true activity.

In Figure 6.8, general labels are applied to the activities being performed, and a meaningful correlation between the tags extracted by the procedure and these labels is evident. Specific examples include the action of driving being labelled with the 'highway' key, and taking the train with 'railway'. Although the tags are not always perfect, they are indicative. For instance, when the individual was at home no residential building was identified, but an indication of the type of location was given by the tags 'lit:yes' and 'highway'. In the region where this data was collected, roads with street lighting typically signify residential areas. A similar relationship is shown in Figures 6.9 and 6.10, with labels applied hierarchically and at lower granularities. While not every item is labelled exactly, this is likely to be a result of the data collection method. We used a data collection rate of one point per minute, meaning that several labelled



Home (00:00:00 -16:25:14) `lit:yes, highway:primary, left_county:northamptonshire`
Driving to Station (16:25:15 - 16:38:58) `highway:primary, highway:secondary, maxspeed:30 mph`
Train to London (16:38:59 - 17:36:26) `electrified:rail, railway:rail, gauge:1435`
Walking (17:36:27 - 18:17:03) `lit:yes, bicycle:yes, sidewalk:both`
At Park (18:17:04 - 18:53:09) `waterway:river, barrier:gate, foot:yes`
Walking (18:53:10 - 21:29:41) `lit:yes, oneway:yes, sidewalk:both`
Train Home (21:29:42 - 22:14:34) `electrified:rail, gauge:1435, frequency:0`
Driving Home (22:14:35 - 22:21:45) `highway:primary, surface:asphalt, maxspeed:60 mph`

Figure 6.8: Partial ground truth: Manually labelled data (in bold) compared against extracted element labels.



Home (00:00:00 - 07:13:32) `oneway:no, maxspeed:30 mph, highway:residential`	
Driving to University (07:13:33 - 07:44:27) `highway:tertiary, surface:asphalt, maxspeed:60 mph`	

On Campus
(07:44:28 - 15:02:18)
`type:multipolygon,`
`amenity:university,`
`building:yes`
	In library (07:44:28 - 08:05:05)
	`type:multipolygon, amenity:university, amenity:library`

	Lecture (08:05:06 - 09:01:15) `highway:bus_stop, surface:asphalt, lit:yes`
	In library (09:01:16 - 13:02:58) `type:multipolygon, amenity:university, amenity:library`
	Lunch at friend's accommodation (13:02:59 - 15:02:18) `type:multipolygon, amenity:university, building:yes`

Trip to Local Store
(15:02:19 - 15:22:25)
`amenity:university,`
`operator:tesco,`
`amenity:fuel`
	Drive to shop (15:02:19 - 15:10:02)
	`amenity:university, maxspeed:30 mph, highway:tertiary`

	At petrol station (15:10:03 - 15:17:59) `operator:tesco, opening_hours:24/7, amenity:fuel`
	Drive to campus (15:18:00 - 15:22:25) `type:multipolygon, amenity:university`

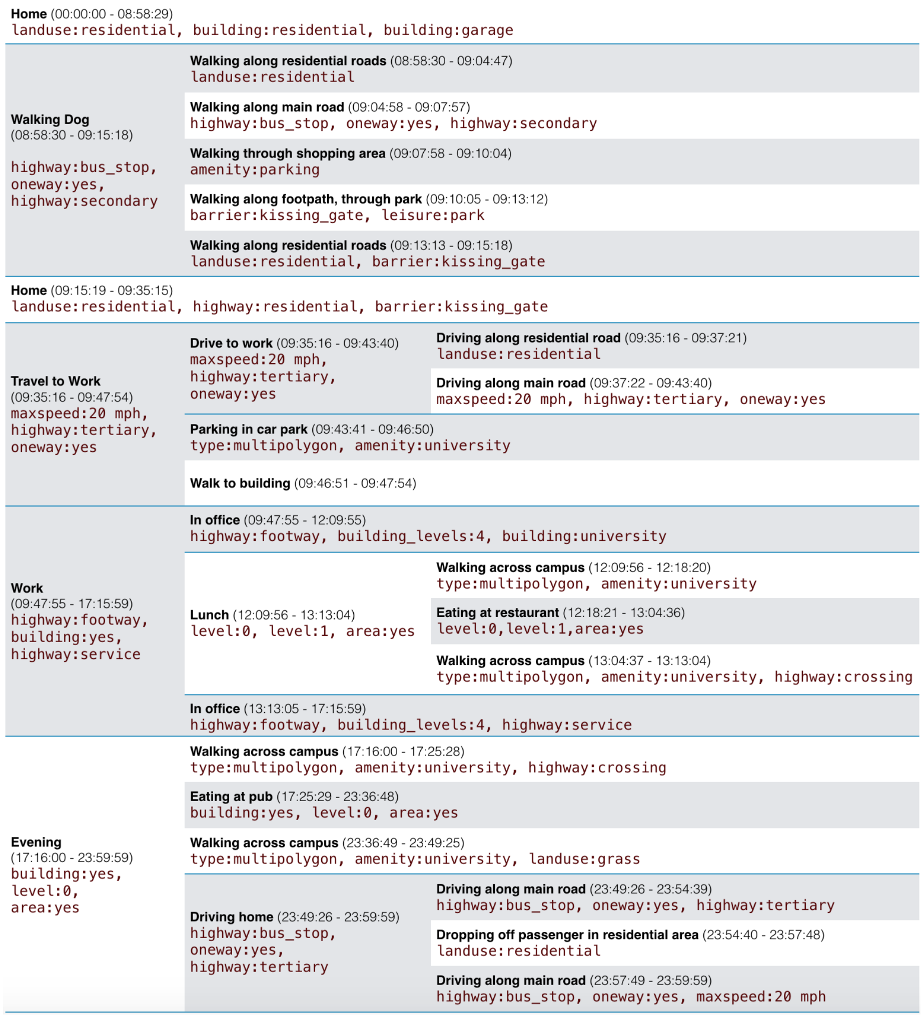
On Campus
(15:22:26 - 18:06:58)
`highway:bus_stop,`
`oneway:yes,`
`surface:asphalt`
	In union building (15:22:26 - 15:48:59)
	`type:multipolygon, amenity:university`

	Lecture (15:49:00 - 18:06:58) `highway:bus_stop, oneway:yes, surface:asphalt`
Driving Home (18:06:59 - 18:38:06) `highway:tertiary, surface:asphalt, maxspeed:60 mph`	

Evening
(18:38:07 - 23:20:00)
`surface:asphalt,`
`lit:yes,`
`oneway:no`
	Walking to town (18:38:07 - 19:01:37)
	`ref:lmngtns, public_transport:pay_scale_area, boundary:public_transport`

	Eating at bar/restaurant (19:01:38 - 20:47:46) `surface:asphalt, lit:yes, building:yes`
	At bar (20:47:47 - 21:39:22) `surface:asphalt, oneway:no, building:yes`
	At bar (21:39:23 - 23:20:00) `surface:asphalt, oneway:no, building:yes`

Figure 6.9: Partial ground truth: Manually labelled data (in bold) compared against extracted element labels, for a different user and with increased granularity over Figure 6.8.



Home (00:00:00 - 08:58:29) `landuse:residential, building:residential, building:garage`		
Walking Dog (08:58:30 - 09:15:18) `highway:bus_stop, oneway:yes, highway:secondary`	**Walking along residential roads** (08:58:30 - 09:04:47) `landuse:residential`	
	Walking along main road (09:04:58 - 09:07:57) `highway:bus_stop, oneway:yes, highway:secondary`	
	Walking through shopping area (09:07:58 - 09:10:04) `amenity:parking`	
	Walking along footpath, through park (09:10:05 - 09:13:12) `barrier:kissing_gate, leisure:park`	
	Walking along residential roads (09:13:13 - 09:15:18) `landuse:residential, barrier:kissing_gate`	
Home (09:15:19 - 09:35:15) `landuse:residential, highway:residential, barrier:kissing_gate`		
Travel to Work (09:35:16 - 09:47:54) `maxspeed:20 mph, highway:tertiary, oneway:yes`	**Drive to work** (09:35:16 - 09:43:40) `maxspeed:20 mph, highway:tertiary, oneway:yes`	**Driving along residential road** (09:35:16 - 09:37:21) `landuse:residential`
		Driving along main road (09:37:22 - 09:43:40) `maxspeed:20 mph, highway:tertiary, oneway:yes`
	Parking in car park (09:43:41 - 09:46:50) `type:multipolygon, amenity:university`	
	Walk to building (09:46:51 - 09:47:54)	
Work (09:47:55 - 17:15:59) `highway:footway, building:yes, highway:service`	**In office** (09:47:55 - 12:09:55) `highway:footway, building_levels:4, building:university`	
	Lunch (12:09:56 - 13:13:04) `level:0, level:1, area:yes`	**Walking across campus** (12:09:56 - 12:18:20) `type:multipolygon, amenity:university`
		Eating at restaurant (12:18:21 - 13:04:36) `level:0,level:1,area:yes`
		Walking across campus (13:04:37 - 13:13:04) `type:multipolygon, amenity:university, highway:crossing`
	In office (13:13:05 - 17:15:59) `highway:footway, building_levels:4, highway:service`	
Evening (17:16:00 - 23:59:59) `building:yes, level:0, area:yes`	**Walking across campus** (17:16:00 - 17:25:28) `type:multipolygon, amenity:university, highway:crossing`	
	Eating at pub (17:25:29 - 23:36:48) `building:yes, level:0, area:yes`	
	Walking across campus (23:36:49 - 23:49:25) `type:multipolygon, amenity:university, landuse:grass`	
	Driving home (23:49:26 - 23:59:59) `highway:bus_stop, oneway:yes, highway:tertiary`	**Driving along main road** (23:49:26 - 23:54:39) `highway:bus_stop, oneway:yes, highway:tertiary`
		Dropping off passenger in residential area (23:54:40 - 23:57:48) `landuse:residential`
		Driving along main road (23:57:49 - 23:59:59) `highway:bus_stop, oneway:yes, maxspeed:20 mph`

Figure 6.10: Partial ground truth: Manually labelled data (in bold) compared against extracted element labels, for a different user to Figure 6.9.

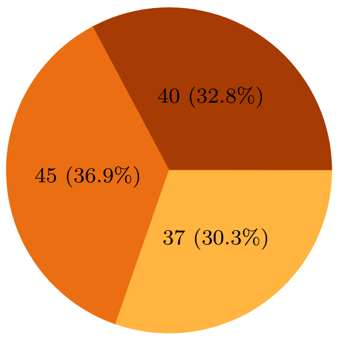
activities consist of only 1 or 2 trajectory points, leaving little information for the procedure to utilise. Similarly, the land usage dataset contains a vast amount of information, but can be limited in parts. An example of this is that the pub which was visited at 17:25 (Figure 6.10) is inside a larger building. The procedure is only capable of identifying that the building was occupied by the user, but there is no information pertaining to which element inside the building was being interacted with, so the only available information is 'building:yes'.

To explore quantitatively how well the procedure worked over these examples, each tag extracted is scored based on the relevancy to the label using three classifications: *high*, *medium*, *low/none*. These scores are manually assigned and shown in Table 6.1, where a *high* tag indicates that the label is very well correlated to the activity (e.g. 'building:residential' to the activity 'Home'), *medium* indicates that there is some link (e.g. 'surface:asphalt' to 'Driving on a main road'), and *low/none* being given to tags with little or no relationship to the activity (e.g. 'highway:bus_stop' to 'Attending lecture'). Figure 6.11a shows the proportion of tags assigned to each of these weightings, demonstrating that the procedure identified tags with *high* or *medium* relevancy 69.7% of the time. We also consider the highest-ranked tag assigned to each labelled time period; the proportion of time periods represented by each tag score is shown in Figure 6.11b. From these results, it is clear that while in the data from the three users only 32.8% of tags were awarded a *high* relevancy score, 60.0% of labels have at least one tag with such a score, and 88.9% contain at least one tag with a score of *high* or *medium*. This indicates that while not all of the 3 tags per label were useful, in nearly all cases at least one of them was.

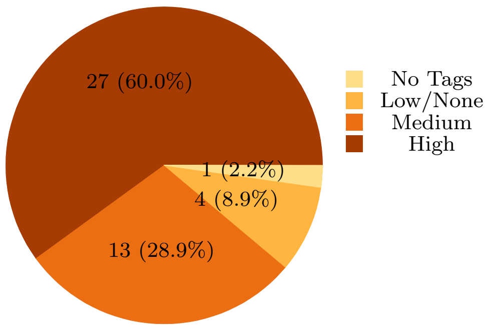
While this evaluation is limited in that it only considers 24 hours' worth of data from 3 different users, it provides an indication that the techniques discussed previously are extracting useful and correct elements. This is demonstrated by showing that there is a strong relationship between the tags identified by the system and the labels manually assigned to data as a partial ground truth. A complete ground truth is not possible in this domain, since the desir-

Table 6.1: Summary of tags and frequency count for each type of ground truth interaction, scored based on the relevancy of each tag (High, Medium and Low/None).

Label	**Tag**	**S**	**#**	**Tag**	**S**	**#**
Home	landuse:residential	**H**	2	barrier:kissing_gate	L	1
	highway:residential	**H**	2	oneway:no	L	1
	building:residential	**H**	1	maxspeed:30	L	1
	building:garage	**M**	1	highway:primary	L	1
	lit:yes	**M**	1	left_county:nor...	L	1
Walking (res.)	landuse:residential	**H**	1			
Walking (shops)	amenity:parking	**M**	1			
Walking (road)	sidewalk:both	**H**	2	highway:bus_stop	**M**	1
	highway:secondary	**H**	1	bicycle:yes	**M**	1
	oneway:yes	**M**	2	ref:lmngtns	L	1
	lit:yes	**M**	2	public_transport:pay...	L	1
	boundary:public...	L	1			
Walking (park)	leisure:park	**H**	1	waterway:river	**M**	1
	foot:yes	**H**	1	barrier:gate	**M**	1
	barrier:kissing_gate	**M**	1			
Driving (res.)	landuse:residential	**H**	2			
Driving (road)	highway:tertiary	**H**	6	maxspeed:60	**M**	3
	highway:primary	**H**	2	maxspeed:30	**M**	2
	highway:secondary	**H**	1	maxspeed:20	**M**	2
	oneway:yes	**M**	4	amenity:university	L	2
	highway:bus_stop	**M**	3	type:multipolygon	L	1
	surface:asphalt	**M**	3			
Parking (uni)	amenity:university	**M**	1	type:multipolygon	L	1
Work (office)	building:university	**H**	2	highway:footway	L	1
	building_levels:4	**M**	2	highway:service	L	1
Walking (uni)	amenity:university	**H**	4	landuse:grass	**M**	1
	highway:crossing	**H**	2	type:multipolygon	L	4
Eating (rest.)	level:0	**M**	1	area:yes	L	1
	level:1	**M**	1	lit:yes	L	1
	building:yes	**M**	1	surface:asphalt	L	1
Eating (pub)	building:yes	**M**	1	area:yes	L	1
	level:0	**M**	1			
Work (library)	amenity:library	**H**	2	type:multipolygon	L	2
	amenity:university	**M**	2			
Work (lecture)	surface:asphalt	L	2	highway:bus_stop	L	1
	type:multipolygon	L	1	oneway:yes	L	1
	lit:yes	L	1			
Visiting friend	amenity:university	**M**	1	type:multipolygon	L	1
	building:yes	**M**	1			
Petrol station	operator:tesco	**H**	1	amenity:fuel	**H**	1
	opening_hours:24/7	**H**	1			
Union (uni)	amenity:university	**M**	1	type:multipolygon	L	1
Bar	building:yes	**M**	2	oneway:yes	L	2
	surface:asphalt	L	2			
Train	electrified:rail	**H**	2	railway:rail	**H**	1
	gauge:1435	**H**	2	frequency:0	L	1



(a) Overall tag relevancy.



(b) Best tag relevancy per time period.

Figure 6.11: The proportion of tags identified as relevant through the ground truth comparison.

able properties of Context Trees will vary significantly based on their intended use. However, we believe that this exploration goes some way to demonstrating the accuracy of the technique.

6.4.3 Clustering

With the applicability of the land usage elements to the task demonstrated, we now explore properties of the Context Trees generated by clustering land usage elements together based on their semantics and properties of the user's interactions with each element. When constructing Context Trees from summarised data, the only required parameter is λ, which specifies the weighting to be given to *semantic similarity* as part of the *HCD* distance metric (Equation 6.5). A weighting of 1 will construct a tree based only on the semantic similarity between node tags, and a weighting of 0 will construct a tree based only on the similarity of features, with any value in between using a combination of the two. The relationship between λ and the number of nodes in a Context Tree is shown in Figure 6.12 (constructed from augmented trajectories from both datasets, filtered with parameters $\delta = 10$, $n = 5$, $d_{min} = 10$, and $t_{max} = 60$). While low values of λ produce fewer context nodes, the meaning behind the identified clusters will be most influenced by the parameter.

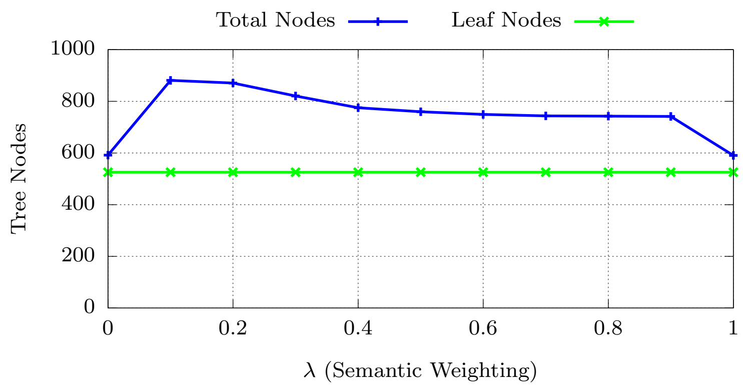


Figure 6.12: The effect of λ on the number of tree nodes in a Context Tree.

Since our work on understanding context from trajectories augmented with land usage information is novel, there are no existing baseline methods or ground truth datasets with which to compare. Instead, we take the closest method to a baseline that exists and compare the results against this. Figures 6.13 and 6.14 show the results of clustering Context Trees using naive distance metrics that consider only geographic distance between elements (Figure 6.13) and temporal distance between interactions (Figure 6.14). While these figures show only one small example, the results are representative of using such metrics in that the elements clustered together have no clear contextual relationship. This is in contrast to the Context Trees generated from the same data using the Hybrid Contextual Distance metric, along with different values of λ, as shown in Figures 6.15–6.17.

In all of these examples, the element identifier has been replaced manually with a descriptive keyword to represent the element. Semantic clustering (shown in Figure 6.15) creates distinctive groups for buildings, footpaths and public amenities, as the elements in these groups are similar, while feature-based clustering (Figure 6.16) creates groups that are less easily identifiable and relate to properties of the elements (e.g. the footpaths are not grouped because they were not encountered in the same journey, but rather were used at different times of the day). Finally, hybrid clustering (Figure 6.17) shows properties of both semantic and feature-based clustering where both the description of the

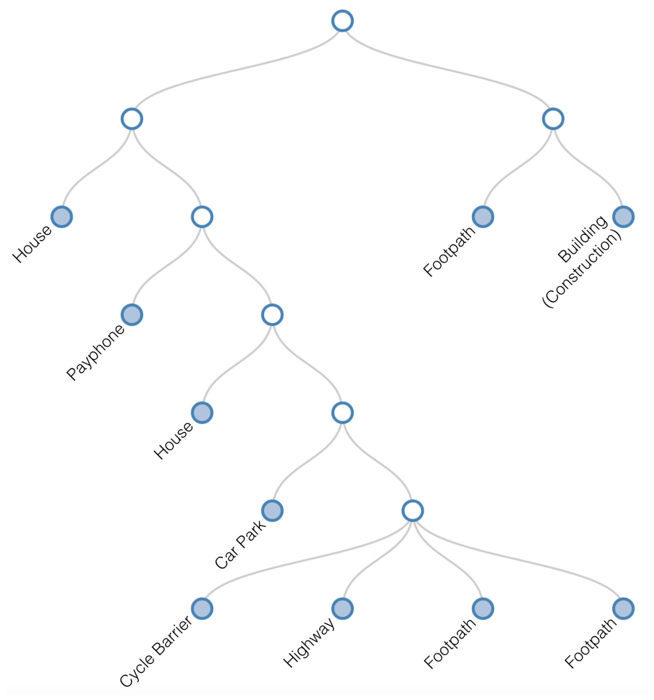


Figure 6.13: Example Context Tree: Geographic clustering.

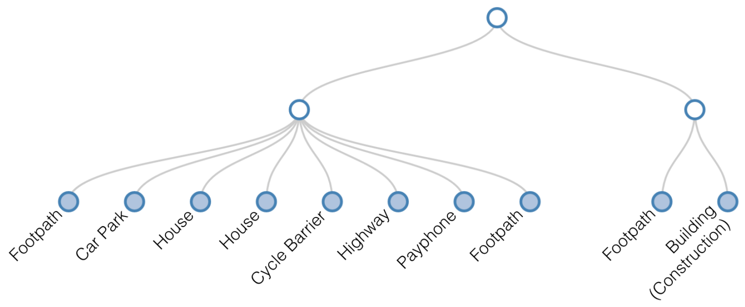


Figure 6.14: Example Context Tree: Temporal clustering.

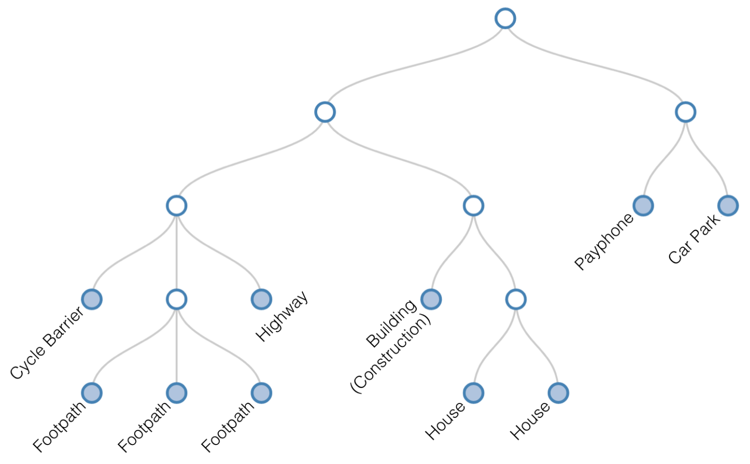


Figure 6.15: Example Context Tree: Semantic clustering ($\lambda = 1$).

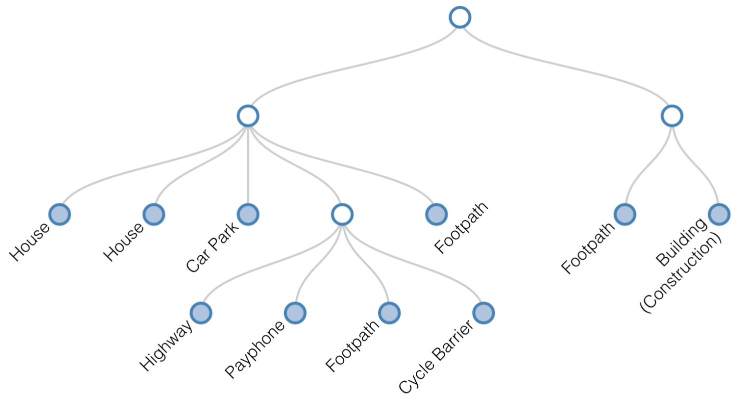


Figure 6.16: Example Context Tree: Feature-based clustering ($\lambda = 0$).

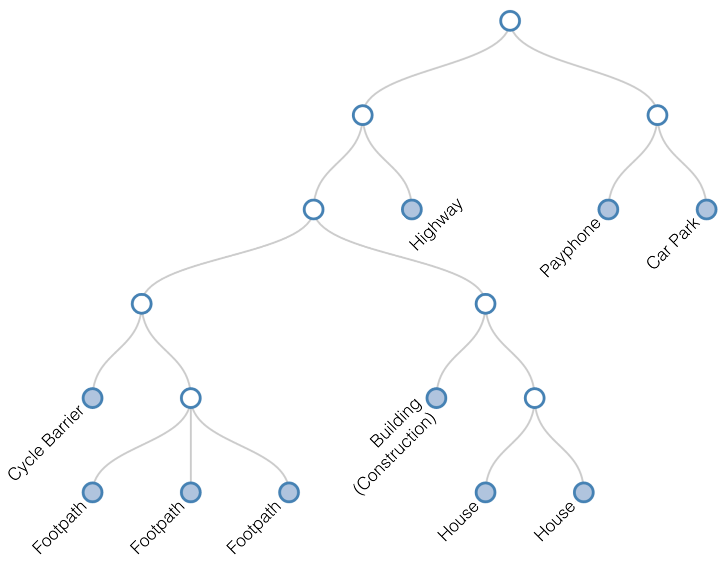


Figure 6.17: Example Context Tree: Hybrid clustering ($\lambda = 0.6$).

element and properties of the interaction with the element are considered to create clusters. Selecting an appropriate value of λ is application-specific.

These Context Trees provide only small examples of the differences between trees generated with naive distance metrics (Figures 6.13 and 6.14) and those generated with the HCD metric (Figures 6.15–6.17). In order to quantify such differences, and given knowledge of the data and how it was collected, we opt to make several assumptions of expected properties of generated Context Trees and explore the extent to which these expectations are violated with each distance metric. While this is of course a subjective evaluation, and the utility will vary based on the specific application to which the Context Tree is put, it goes some way to providing an indicator of the utility of this approach in lieu of a ground truth. The assumptions made are:

1. Buildings should be grouped together unless they have very different uses (e.g. residential buildings should not be in the same group as office buildings).
2. Roads should be grouped together, with elements relating to roads grouped at a higher level (e.g. junctions).

3. Public amenities should be grouped together unless the interactions have very different properties.

These assumptions focus on the semantics of elements, but the features also need to be considered when exploring possible reasons for clusters being split. For instance, if a person visited many houses as part of their job, it would be reasonable to assume that these houses should be semantically close to the residence of the individual in the Context Tree, but not at exactly the same level. The usefulness of such assumptions will depend on the application, but it is possible to see that when aiming to characterise how a person has spent their time, it is beneficial to identify the times spent at residential buildings separately to those spent at work. On the small example Context Trees shown in this section, geographic and temporal clustering (Figures 6.13 and 6.14) violate all 3 assumptions. Semantic clustering (Figure 6.15) adheres best to these assumptions, with the houses grouped at the same level and the building under construction close by in the next level up. Similarly, the footpaths have been grouped together with the cycle barrier, a related element, and the highway grouped one level up. Feature-based clustering (Figure 6.16) has fewer valid assumptions than semantic clustering, as it only considers the interactions with the elements and not the elements themselves. Although the houses are together in a single cluster, they are also joined with the car park and footpath. Finally, hybrid clustering (Figure 6.17) is very similar to semantic clustering with the exception that the highway is no longer situated close to the footpaths, but is further up the Context Tree by itself. This still leaves 2 of the assumptions strictly adhered to, with 1 very close: a change that can be explained by the consideration of interaction features, where the highway has a different profile of interaction than the footpath and cycle barrier elements. Again, these are small examples; however the trends present have been observed to be consistent across larger Context Trees.

With a better understanding of filtering, summarising, and clustering, we turn our attention to exploring how data influences the properties of the gener-

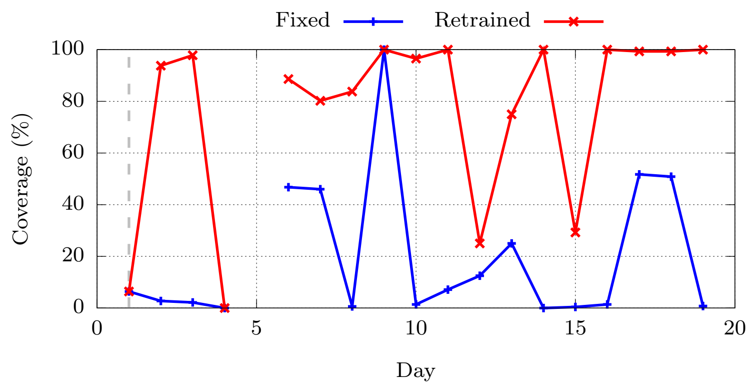
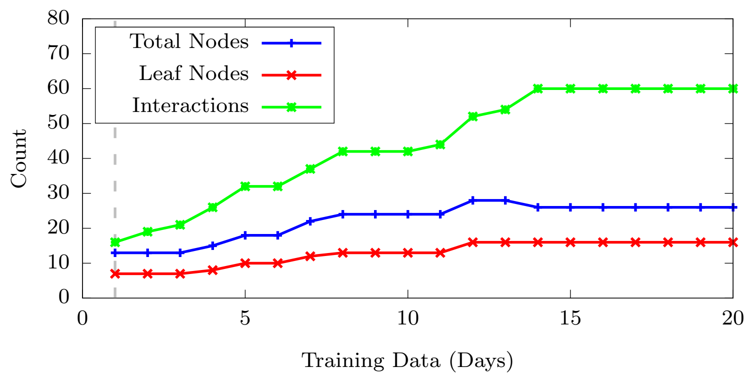


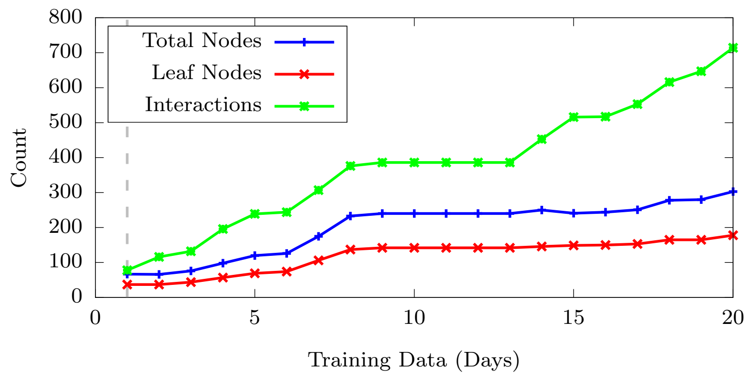
Figure 6.18: Example data from a single user showing the proportion of new land usage elements encountered each day for a sequence of 20 days. Note that no data was recorded during day 5 for this user.

ated Context Tree. Focusing on 21 days of data from a single user, Figure 6.18 shows repetition in data by using the first day as a set of *training* data and calculating the coverage (i.e. the proportion of elements encountered in the *test* day present in the *training* day's data) for each following day, shown by the blue *Fixed* line. Additionally, the red *Retrained* line shows the coverage when using all previous days (i.e. 0 to $d - 1$, where d is the current day) as the *training* set. The total number of nodes, leaf nodes, and interactions for a Context Tree generated using the same data (where day d shows a summary for a tree built using all data from days 0 to d) are shown in Figure 6.19a. Please note that no data was recorded during day 5 for this sample user in the MDC dataset.

Figure 6.18 begins with a low coverage for both *Fixed* and *Retrained* lines, indicating that few elements encountered in day 1 were present in the training set (day 0). However, while the *Fixed* score remains low for days 2–4, the *Retrained* score approaches 100%. In this instance, this is indicative of the user visiting elements they did not encounter in the initial training day (day 0), but that they did encounter during subsequent days, as the *Retrained* line includes all previous days as training data. The figure shows similar results for the remaining test days, where during day 9 the user visited only locations visited during day 0; during days 9, 11 and 16–20 the user encountered no



(a) MDC user.



(b) Warwick user.

Figure 6.19: The relationship between amount of training data and number of tree nodes in a Context Tree.

new elements as the score for *Retrained* is at 100%. Figure 6.19a shows how these properties relate to the size of Context Trees generated. The number of *leaf nodes* is the number of unique elements and the number of *interactions* is a count of the total number of interactions extracted. That is, if the user encountered the same element 3 times, or 3 different elements, both would count as 3 interactions. At day 1, the number of interactions is roughly the same as the number of leaf nodes, indicating that all elements were encountered approximately once. As time goes by, more elements are encountered, but a large number of existing elements are revisited, demonstrated by the disproportionate rise in the number of interactions. This indicates that over a short period, where the user is likely to have remained within a single region, the size of the tree does not increase significantly as additional data is added. However, considering trees over larger time periods will not have the same property as the user is likely to visit new regions with entirely new leaf nodes. Figure 6.19b shows a similar graph as Figure 6.19a, however it was generated using data from a user of the Warwick dataset instead of the MDC dataset. As is evidenced by the figures, the procedure extracts similar trends in users from each dataset.

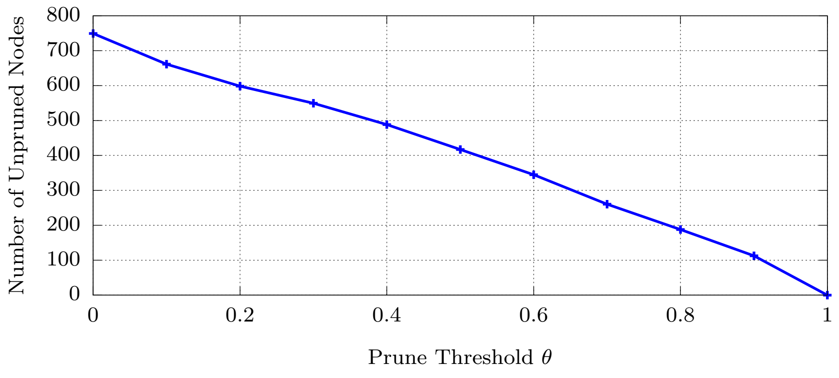
This section has characterised the outputs and properties of the Context Tree generation procedure presented in Section 6.3. While the concept of a ground truth for this work is not applicable, and existing approaches for comparison are lacking, through the provision of multiple small examples and a discussion of general trends we have demonstrated the applicability of the approach to the task of identifying similar contexts and storing such information in a hierarchical data structure.

6.4.4 Pruning Evaluation

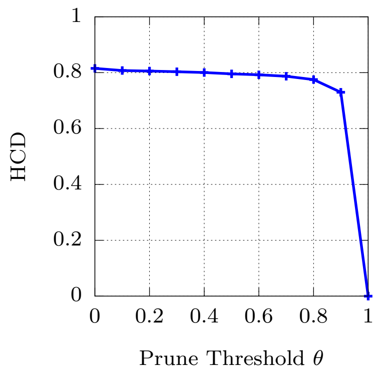
Pruning requires a pre-built Context Tree and two parameters, namely θ and ξ, where θ provides a threshold for pruning and ξ is a penalty associated with every node when calculating its *storage cost*.

Figure 6.20 shows the effect of varying θ when pruning Context Trees generated from the same data and parameters as those used in Figure 6.12, with $\lambda = 0.5$ and $\xi = 1$ and trajectories from both datasets. From this figure it is possible to see that the number of nodes in a Context Tree can be reduced drastically while maintaining the majority of the information. Selecting $\theta = 0.8$, the resultant pruned Context Tree contains approximately 25% of the nodes present in the unpruned tree, but maintains almost 75% of the useful information. While the process to prune the Context Tree adds additional complexity, the resultant tree is considerably more compact and thus applications that require storing or searching the tree will have significantly lower overheads.

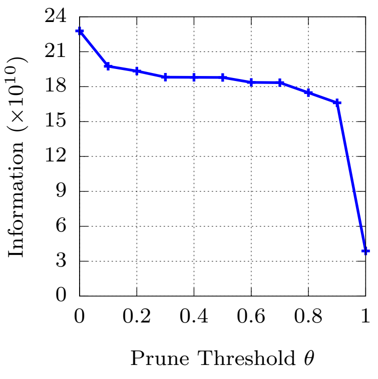
Using the same data again, but this time holding $\theta = 0.2$, Figure 6.21 shows the effect of changing ξ on the number of unpruned nodes, average HCD and



(a) Node Count.



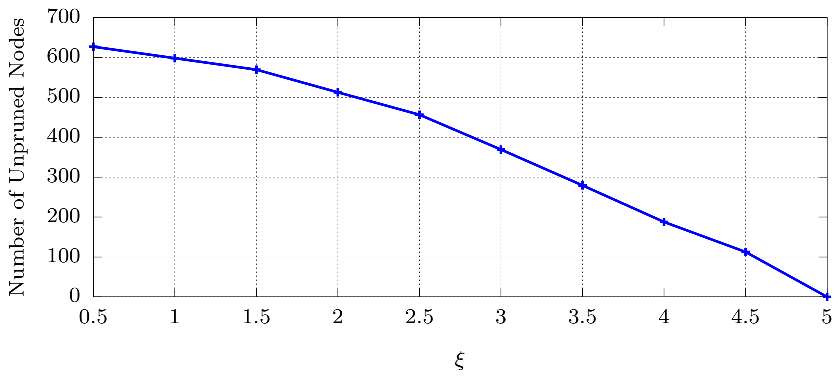
(b) Average HCD.



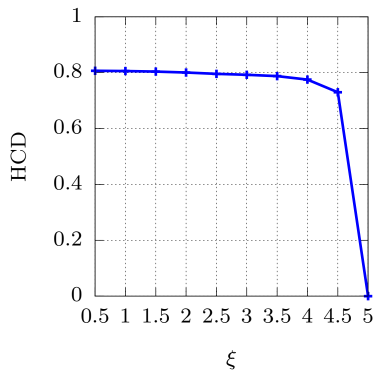
(c) Total Information.

Figure 6.20: The effect of θ on number of nodes in a sample Context Tree ($\lambda = 0.5, \xi = 1$).

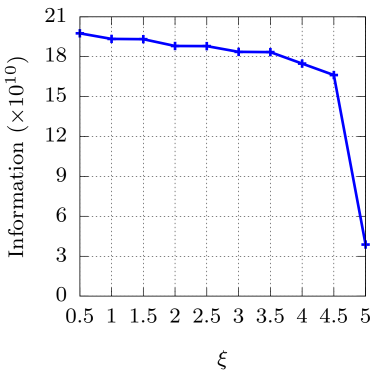
information contained within the tree. Increasing either θ or ξ reduces the number of nodes remaining after pruning (Figures 6.20a and 6.21a), as increasing θ specifies a higher threshold required to maintain a node, and increasing ξ assigns a higher cost to each node, making it less likely to exceed the threshold. The results also demonstrate that as more nodes are pruned from the Context Tree, the average distance of the remaining nodes becomes smaller (i.e. they become more similar, Figures 6.20b and 6.21b). Finally, Figures 6.20c and 6.21c demonstrate that although pruning reduces the total information in the tree, it does so gradually until the number of unpruned nodes approaches 0, under the definition of data, i.e. useful information, presented in Equation 6.7 (Section 6.3.5). This helps to demonstrate the effectiveness of pruning as the number of nodes



(a) Node Count.



(b) Average HCD.

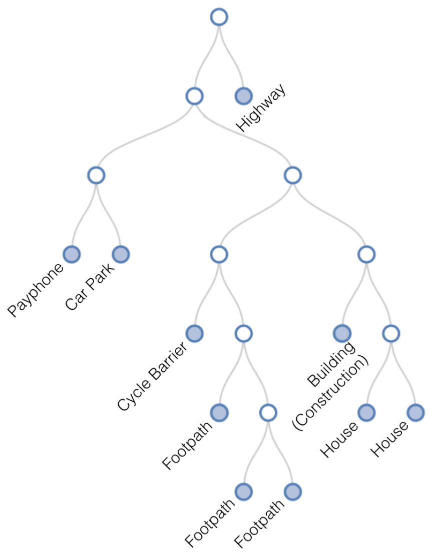


(c) Total Information.

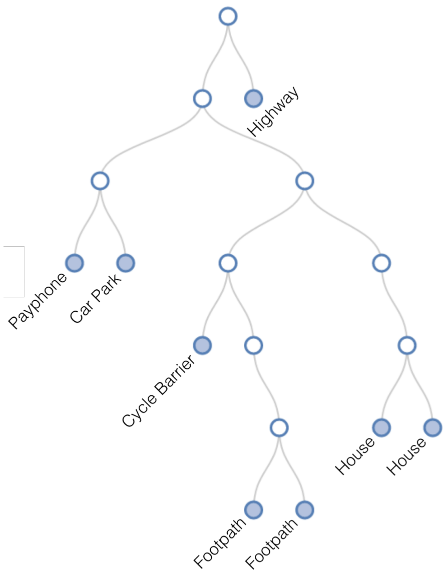
Figure 6.21: The effect of ξ on number of nodes in a sample Context Tree ($\lambda = 0.5, \theta = 0.2$).

in the tree can be reduced drastically, but the amount of information remains high.

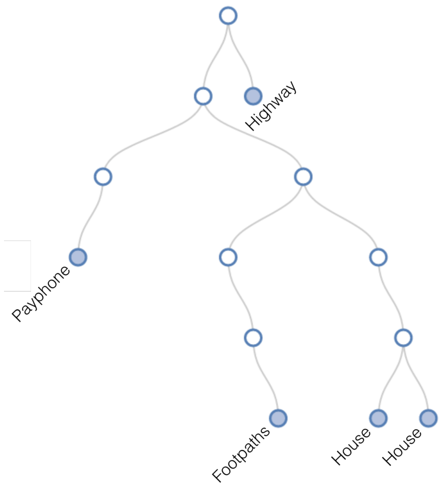
Figure 6.22 shows how pruning affects trees generated from real-world data (using the same data and clustering as in Figure 6.17). With the lowest value of θ ($\theta = 0.25$ shown in Figure 6.22b), only two leaf nodes have been pruned: one of the footpaths and one of the buildings. Increasing θ ($\theta = 0.35$ shown in Figure 6.22c) causes more leaf nodes to be pruned, and a further increase ($\theta = 0.45$ shown in Figure 6.22d) has the effect of pruning entire sub-trees, resulting in a much smaller and more compact tree. Although containing less information, such pruned trees provide benefits in resource-constrained applications where storing and processing an entire tree may be infeasible.



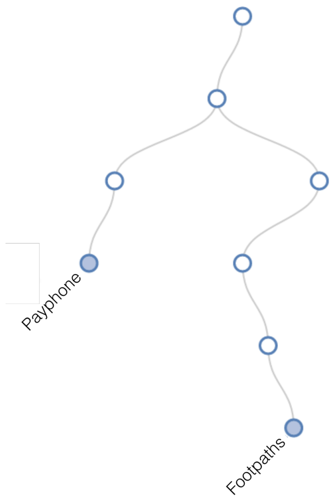
(a) Unpruned tree.



(b) $\theta = 0.25$.



(c) $\theta = 0.35$.



(d) $\theta = 0.45$.

Figure 6.22: Example Context Trees pruned with different values of θ, with $\xi = 1.5$.

6.5 Summary and Conclusion

This chapter presented and evaluated the *Context Tree* data structure, along with techniques for generating Context Trees from augmented geospatial trajectories. The Context Tree is a novel hierarchical data structure that identifies and summarises user contexts at multiple scales, allowing rapid access to summary information about a user's interactions with their environment. From this, the structure provides a foundation for further analysis, understanding, and modelling of the behaviour of individuals and groups. The Context Tree and its generation and pruning procedures are evaluated over real-world trajectories through a comparison to expected properties and a partial ground truth.

The process for generating a Context Tree begins with augmented geospatial trajectories, and clusters the land usage elements present in such trajectories using a new distance metric, the Hybrid Contextual Distance (HCD) metric that considers the semantics of elements and properties of the users' interactions with these elements. This distance metric then becomes the foundation for hierarchical clustering and construction of the Context Tree summary model. By summarising contexts into a single data structure, it becomes easier to detect changes in routine through anomaly identification, identify similarities and differences between users, and predict users' future actions. These areas are proving to be increasingly important to the provision of tailored and useful services both on individual and societal scales. The following Chapter builds upon this foundation by converting the Context Tree into a predictive model capable of predicting both the future context and interactions of individuals, further demonstrating its utility.

CHAPTER 7
Applying Context Trees: The Predictive Context Tree

Summarising the past actions of individuals goes a long way to understanding how people spend their time, but it is the prediction of future actions that allows us to provide real utility in the form of tailored and optimised services. Using the Context Tree, presented in Chapter 6, as a basis, this chapter presents the *Predictive Context Tree (PCT)*, a hierarchical classification model that both summarises and predicts the future contexts and interactions of individuals. The PCT is evaluated over real-world trajectories, with results demonstrating that the PCT achieves higher predictive accuracies than existing approaches predicting over extracted locations (Chapter 4), and matches the performance achieved by predicting over land usage interactions (Chapter 5), while adding utility in the form of context predictions. Such a prediction system is capable of understanding not only where a user will visit, but also future context in terms of what they are likely to be doing.

7.1 Introduction

Prediction of the future actions of individuals in existing literature primarily focuses on predicting locations the individual will visit. While location can be used as a source of context, knowing only the geographic region that a person is likely to visit provides little information. In order to improve upon this, Chapter 5 proposed a methodology for extracting real-world land usage elements as a basis for prediction. Using real-world elements for prediction provides applications with more information, such as the type and properties of elements. Such information can be leveraged to understand what action a person may wish to perform, and thus form a basis for offering services tailored to this action.

Combining the idea of prediction with the Context Tree summary model presented in Chapter 6, this chapter focuses on predicting both future element interactions and the future contexts of individuals. Specifically, the proposed technique, the Predictive Context Tree (PCT), is an extension of the Context Tree that converts the data structure into a classification model that makes use of the existing hierarchical structure of the Context Tree. This chapter presents the PCT and evaluates its performance over real-world trajectories and compares it with existing approaches that predict locations and land usage elements. The results demonstrate increased accuracies when compared with predictions made over extracted locations, and commensurate accuracies with the technique proposed in Chapter 5, as well as increased utility over both approaches when allowing for context predictions.

Section 7.2 provides a summary of related work in the area of prediction and classification. The PCT is presented in Section 7.3, and evaluated in Section 7.4. The chapter is concluded with a discussion in Section 7.5.

7.2 Related Work

The work in this chapter builds upon the other contributions of this thesis, specifically using the Land Usage Identification (LUI) procedure presented in Chapter 5, where background information can be found in Section 5.2. Additionally, the Context Tree data structure, presented in Chapter 6, is extended for this work, with relevant related work being summarised in Section 6.2. The extension to the Context Tree takes the form of converting it into a hierarchical classification model.

Hierarchical classifiers operate in a similar way to traditional classifiers, by taking a set of labelled *training* data, constructing a model and returning classifications from this model when presented with unlabelled *instances*. They differ, however, in that the model is capable of learning from, or making use of, hierarchical relationships between nodes or concepts. Existing work has considered hierarchical classifiers as a basis for text classification, where a hierarchy of labels may exist [Dumais and Chen, 2000; Rousu et al., 2005], as well as in a general form for multiple domains [Cesa-Bianchi et al., 2006; Gopal et al., 2012]. Achieving hierarchical classification is typically achieved in one of two ways: *big-bang* classifiers are those where a single classifier is constructed for the entire hierarchy, and are usually application-dependent. *Top-down* classifiers, on the other hand, make use of an existing hierarchy and classification technique by training a classifier at either each node or each level in the hierarchy [Silla Jr and Freitas, 2011].

Many existing techniques can be used as the classifiers at each level or node in a top-down hierarchical classification system, including decision trees [Jin et al., 2009; Quinlan, 1996], Support Vector Machines (SVMs) [Cristianini and Shawe-Taylor, 2000], and Artificial Neural Networks (ANNs) [Mitchell, 1997]. The top-down approach is the one we select for the PCT, as it is capable of learning the hierarchical structure of the Context Tree without requiring an entirely new model to be devised.

7.3 The Predictive Context Tree (PCT)

The Predictive Context Tree (PCT) (Figure 7.1) is a hierarchical predictive model, built upon the Context Tree (Chapter 6), that is capable of both summarising a user's historical contexts as well as predicting their future element interactions and contexts as a classification model. This section details the procedure to convert a Context Tree into a PCT through the training of classification nodes in the Context Tree. As the Context Tree is already a hierarchical data structure, it contains a vast amount of information pertaining to the relationships between nodes, so it is desirable to conserve this information in a predictive model. For this work, we maintain this hierarchical structure by training the Context Tree as a *top-down* predictor.

This conversion is achieved by augmenting each node in a constructed Con-

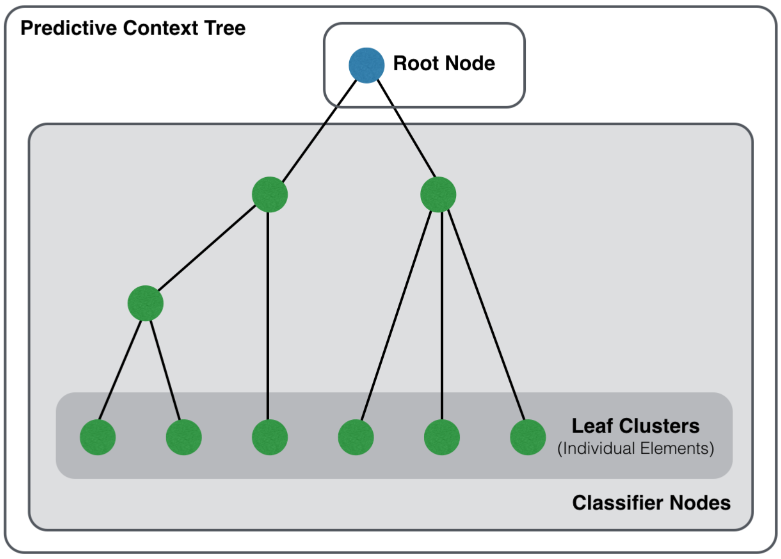


Figure 7.1: Abstract representation of a PCT.

text Tree, except the root node, with a binary classifier that is tasked with answering the question "does this instance belong in the subtree rooted at this node?" when presented with an instance for classification. With each non-root node capable of answering this question, the overall classification of an instance occurs by starting at the root node and requesting a classification from each of the root's children; children will be selected to follow based on a criteria until a final classification is reached, again determined by a criteria. The goals of prediction will determine the criteria:

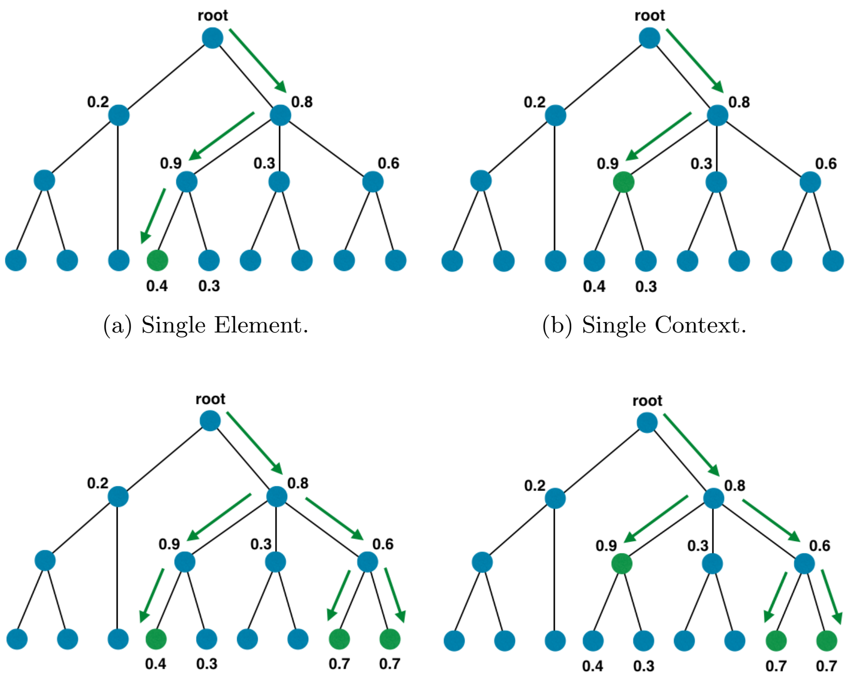
- **Single element**: When predicting specific land usage elements, the predictor must return a leaf node, achieved by following the child with the highest confidence at each stage. This process is shown in Figure 7.2a.
- **Single context**: When predicting contexts instead of land usage elements, there is no requirement for a prediction to reach a leaf node. Figure 7.2b shows the procedure for single context prediction, where at each node the child with the highest confidence is selected providing that the confidence is at least T_s, the *selection threshold*. If no child has confidence of at least T_s, the current node is returned as the class label.

- **Multiple element**: When trained on augmented trajectories that allow more than one land usage element to be associated with each trajectory point, PCTs can be used to predict the multiple land usage elements that will be interacted with next, unlike *single element* which is limited to one leaf. This is achieved at each stage by following all classifiers that return confidence $> T_s$ if such exist, otherwise taking the one with the highest confidence, as shown in Figure 7.2c. Once a node has returned a classification with confidence $> T_s$, the procedure continues even if the children of this node have confidence $< T_s$. The child with the highest confidence would be followed regardless.
- **Multiple context**: Again, this technique aims to predict multiple elements, but when confidence is low in specific elements, contexts or combinations of elements and contexts can be predicted instead. Predictions are made by taking all children of a node whose confidence is above T_s, and returning the node when no child fulfils this criteria, as shown in Figure 7.2d.

Single element and *multiple element* are examples of *mandatory leaf node* prediction, in that the predictive model is required to return only leaf nodes from the tree. Similarly, *multiple element* and *multiple context* are examples of *hierarchical multi-label* classifiers in that they can return more than one class label to a given test instance [Silla Jr and Freitas, 2011], although they can still return leaf nodes (i.e. elements) if confidence is high.

7.3.1 Training a PCT

Training a PCT takes a set of *instances* as the *training set*, with known class labels. The PCT can be used for *next location* or *next context* prediction where the class label is an identifier pointing to the next location or context of the user after finishing with their *current* context or location. In contrast, *future location* or *future context* prediction may generate training instances for each



(a) Single Element. (b) Single Context.

(c) Multiple Element. (d) Multiple Context.

Figure 7.2: Classification methods for PCTs. Classification begins at the root node and selects which children to follow based on the output of their binary classifiers, with different selection schemes and $T_s = 0.5$.

time step and the class label is simply which location or context the user was in during that time window. These instances are fed into each classifier in turn, with the class variable modified to become binary in the following ways:

- If the instance's class represents this node, it is a `positive` example.
- If the class represents a node in the subtree rooted at this node, it is a `positive` example.
- If the class represents a sibling of this node, or a descendant of one, it is a `negative` example.
- If the class represents an ancestor of this node, it is a `negative` example.
- If the class represents any other node, it is ignored and not used for training this classifier.

Figure 7.3 shows how each node treats a particular instance. Training

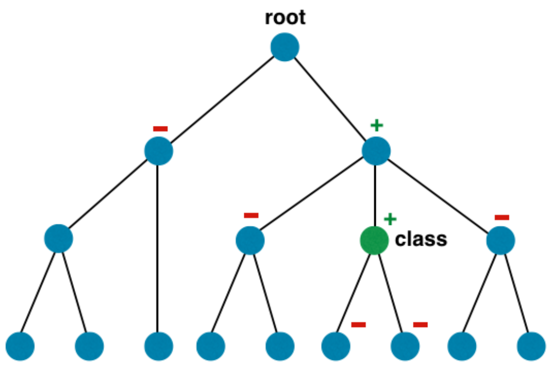


Figure 7.3: Example of how a training instance is treated by each classifier when the class label is associated with the node labelled 'class'. All nodes labelled with '+' treat this instance as a positive example, nodes labelled '-' treat it as negative, while nodes without a label ignore this instance for training.

through this process ensures that the hierarchical links between elements and contexts are learnt by the PCT, as each node's classifier is trained to return `yes` if the instance belongs to itself or one of its descendants, or `no` if the instance belongs to a sibling or one of their descendants (i.e. following this particular child would be a mistake). Other nodes are ignored because they do not relate to the current problem.

Each node can now be trained as a binary classifier using standard techniques, such as decision trees or k-nearest neighbour approaches. However, using SVMs, which are tailored to the task of binary classification, and have been shown to be applicable to the specific area of location prediction, are likely to be good candidates here [Frohlich and Zell, 2005; Wang and Prabhala, 2012].

SVMs are not probabilistic classifiers, so they are traditionally trained to return only a class value (in our case, `yes` or `no`). However, the PCT requires a *confidence* value for each classification to determine which child or children to follow at any given stage in the process. To calculate these probabilities, we use a variant of the SVM classifier which uses logistic regression to calculate class probabilities when returning classifications, and we evaluate other, probabilistic, models in Section 7.4.

7.4 Evaluation

Evaluating the PCT takes the form of exploring the predictive accuracies obtained from the model, in comparison to existing approaches and with respect to parameters used to train the underlying Context Trees. Constructing PCTs for evaluation is performed using the methodology presented in Section 7.3 and the same data as used in Chapters 5 and 6. For the first stage of evaluation, we focus on *single element* and *single context* predictions, so we limit the number of land usage elements that can be associated with each trajectory point to one (i.e. $n = 1$, the same as used in Chapter 5). As we are imposing this limit, we also make use of the *maxradius* parameter introduced in Chapter 5, Section 5.3.1, and set it to 50m. Although this parameter was removed for Context Tree construction in Chapter 6, it is required here to ensure comparable predictions to those made in Chapter 5. Throughout this evaluation, we refer back to the predictive accuracies presented in Section 5.4, for both predictions made over extracted locations and identified land usage elements using the technique that produced the highest accuracies, namely SVMs.

In addition to this, we also consider constructing PCTs from multi-element land usage datasets (i.e. those that allow more than one element to be associated with a single trajectory point; $n > 1$). Multi-element datasets may be useful if, for example, a person is interacting with a building that is contained within a larger building (e.g. a shop in a shopping centre), so we also utilise the land usage extraction procedure with different values of n. Generating training instances for these datasets uses the same features as in previous predictions we have evaluated in this thesis, namely: *day of year*, *day of week*, *start hour*, *start minute*, *duration*, *current identifier (element or location)*, and *class (next identifier)*. However, the class label becomes the set of elements that the user interacts with next. This is defined as taking the next interaction in the dataset, selecting all other interactions that overlap, and combining the identifiers of all such elements into a single string value that represents the set of elements.

7.4.1 Constructing Predictive Context Trees

Context Trees are constructed from the land usage datasets, both single and multi, and using the Hybrid Contextual Distance (HCD) metric with $\lambda = 0.5$, an empirically selected value that produces representative results. The HCD metric is a similarity measure that balances semantic and feature similarities into a single score used for clustering Context Trees, where λ specifies the weighting towards semantic similarity. The task now becomes that of converting the generated Context Trees into Predictive Context Trees and evaluating the predictive ability of such a hierarchical model. Each non-root node in the Context Tree is trained as a binary classifier using an SVM with the modification of instances as described in Section 7.3.1.

7.4.2 Evaluating Predictions

For all location and element prediction approaches (extracted location, land usage, and single element PCT), the training data's class label represents the next extracted location or land usage element that the user interacted with. Evaluating the correctness of a prediction can simply be performed by comparing the output of the predictor against the known class, referred to as an *element correct* prediction.

Definition 7.1 *A prediction is* ***element correct*** *if the predicted and actual nodes are the same.*

For *context* prediction, in some cases the PCT will return a leaf node which can then be compared to see if it is *element correct*. In other cases, a non-leaf node will be returned which requires the introduction of the notion of *context correctness*.

Definition 7.2 *A prediction is* ***context correct*** *if the node represented by the predicted class label is an ancestor of the actual class node.*

For trees constructed over multi-element land usage datasets, predictions take the form of a set of elements or contexts, so we require evaluative methods to differentiate between these predictions. For this, we define several tests, applied in order such that the first test that passes is used as the classification, with examples illustrated in Figure 7.4.

Full element correct: The set of predicted land usage elements matches the set of actual elements exactly.

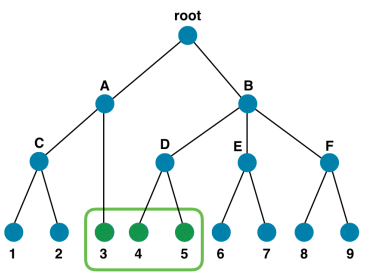
Full context correct: Every member in the set of actual elements is represented in the predicted set either by itself or an ancestor in the tree. Additionally, every element in the predicted set is either contained within, or an ancestor of at least one element in, the actual set (and the predictions are not full element correct).

Partial element correct: Some elements were correctly predicted: the union of the predicted and actual sets is nonempty (and neither of the previous classifications are applicable).

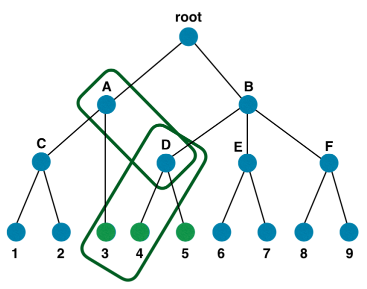
Partial context correct: Some contexts were correctly predicted: the union of the predicted set and the set of all ancestors of members of the actual set is nonempty (and none of the previous classifications are applicable).

Incorrect: There is no overlap between the predicted and actual sets, or the predicted set and ancestors of members of the actual set.

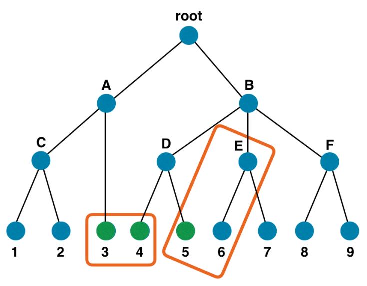
These tests ensure that we can evaluate the predictions made by a multi-element context tree. If the tree predicts the exact set of elements to be interacted with, the prediction is *full element correct.* If some or all elements are represented by their contexts because element prediction confidence was not high enough, it is *full context correct.* In times when some elements were predicted, but some missed (and not represented by a context) or some erroneous elements or contexts were included, it is *partial element correct. Partial context correct* is similar, but for times when no elements were correctly predicted, only some contexts were. Finally, classifications which have no correct elements or



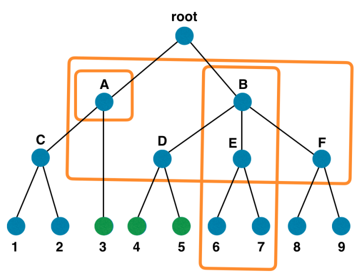
(a) Full element correct: the set of elements is predicted exactly.



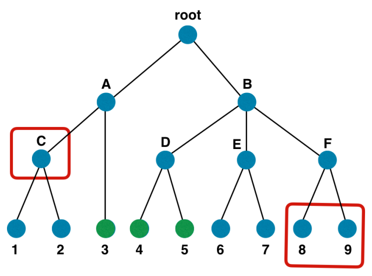
(b) Full context correct: the element or a corresponding context for each element is predicted, and no erroneous elements or contexts are predicted.



(c) Partial element correct: some elements are predicted correctly, but either elements are missing or additional elements and/or contexts are predicted.



(d) Partial context correct: no elements are correctly predicted, but some predicted contexts are ancestors of correct elements. Either some correct elements are not covered by a predicted context or additional contexts are predicted that do not relate to a correct element.



(e) Incorrect: the predicted set does not contain any correct elements or ancestors of correct elements.

Figure 7.4: Example classification labels for different predicted sets through the PCT, where the correct prediction is the set $\{3, 4, 5\}$. Each outlined box represents a classification that will be given the corresponding classification label.

contexts are *incorrect*. These metrics are used as the basis for evaluating individual predictions, with the overall PCT model evaluated using 10-fold cross validation.

7.4.3 Results

Figure 7.5 reproduces Figure 5.12a from Chapter 5, showing the predictive accuracies achieved for location prediction and land usage element prediction for the SVM classifier from the Warwick dataset, with results for the Nokia Mobile Data Challenge (MDC) dataset shown later in Figure 7.10. With these baselines in place, the task becomes that of understanding the relative performance of the PCT. Figure 7.6 shows the predictive accuracies achieved when using the PCT to predict which land usage element a user will interact with, which was the same task performed by the SVM in Figure 7.5 (with the performance of the SVM predictor shown for comparison). As with Chapter 5, the results for both the Warwick and MDC datasets are consistent, so we show only the results for the Warwick dataset here, and summarise results for the MDC data later in Figure 7.10. The figure demonstrates that the PCT produces comparable predictive accuracies to SVM when predicting the next element a person will interact with. The PCT is also designed to predict contexts as well as elements; the accuracies achieved for context prediction are shown in Figure 7.7 and Table 7.1. A comparison of the performance of all techniques is shown in Figure 7.8, and tabulated in Table 7.2, for d_{min} = 10min and 40min, where location extraction was performed by *thresholding*, since it gives the highest predictive accuracies. For d_{min} = 10min, predicting over extracted locations using SVMs provides the highest element correct accuracy (i.e. it is best able to predict the exact location or element to be interacted with). However, when allowing for contextual prediction, the PCT outperforms this existing approach. With larger values of d_{min}, such as 40min, shown in Figure 7.8b, predicting over identified land usage elements provides higher accuracies than predicting over extracted locations for both the SVM based approach from Chapter 5, and the

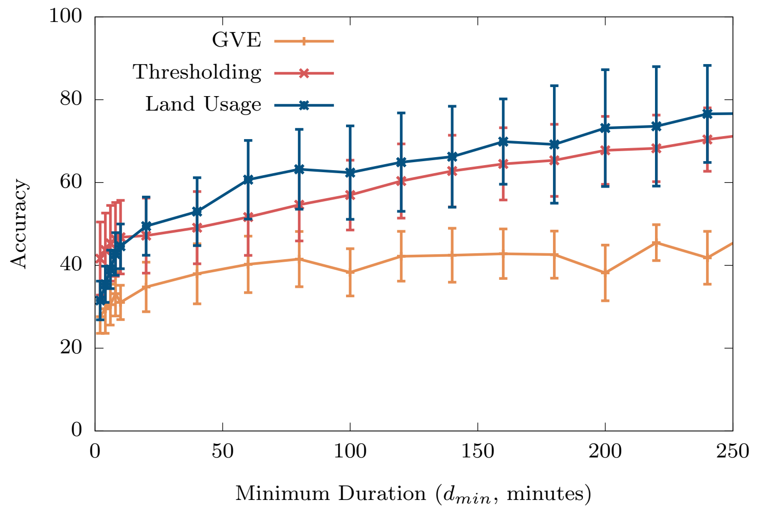


Figure 7.5: The effect of minimum interaction duration, d_{min}, on predictive accuracy for existing location extraction and prediction techniques and the LUI procedure for the Warwick dataset (reproduced from Chapter 5).

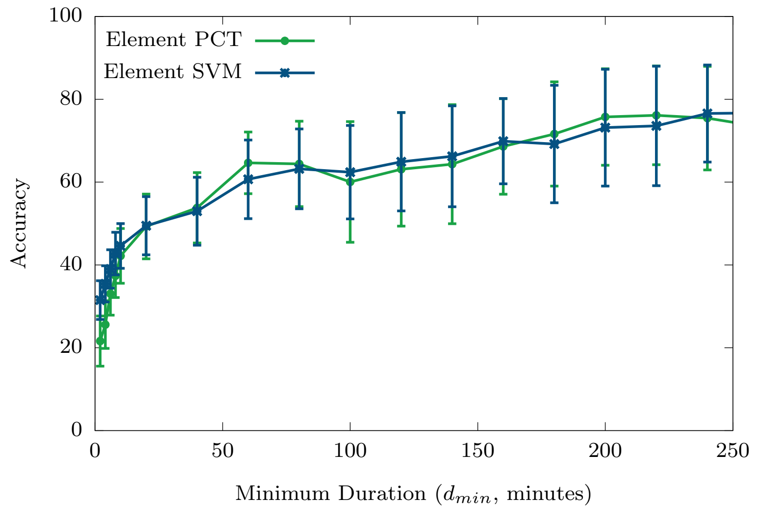


Figure 7.6: Predictive accuracy of the PCT, considering element prediction ($\delta = 5,\ t_{max} = 60,\ \lambda = 0.5,\ T_s = 0.6,\ n = 1,\ maxradius = 50$).

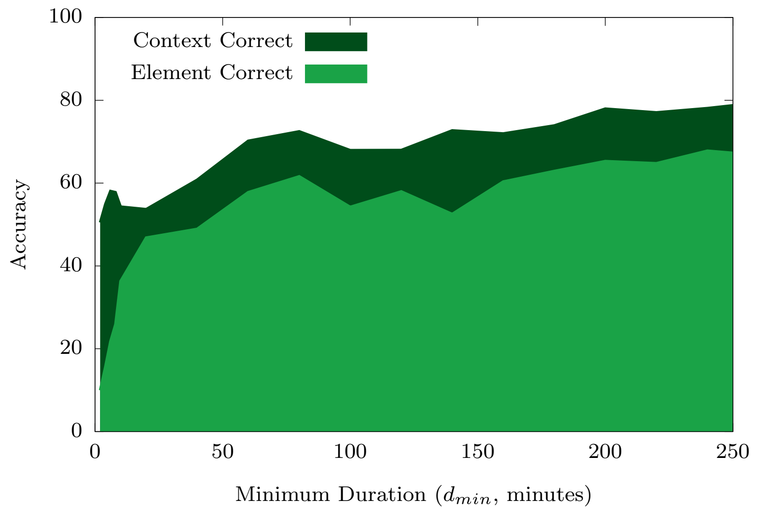
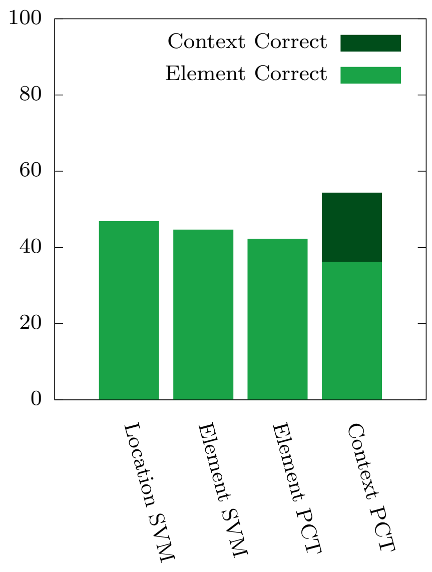


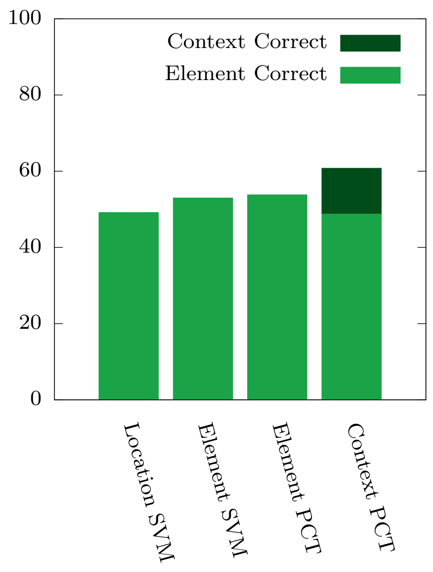
Figure 7.7: Predictive accuracy of the PCT, considering context prediction ($\delta = 5,\ t_{max} = 60,\ \lambda = 0.5,\ T_s = 0.6,\ n = 1,\ maxradius = 50$).

Table 7.1: Accuracy of the PCT when predicting contexts for the Warwick dataset, with standard deviation shown in brackets.

d_{min}	**Element Correct**	**Context Correct**
2	40.5 (8.2)	50.5 (3.5)
4	39.2 (9.9)	54.9 (6.0)
6	36.3 (12.4)	58.1 (8.2)
8	31.9 (9.3)	57.8 (7.2)
10	18.0 (6.3)	54.3 (7.6)
20	6.9 (4.7)	53.7 (8.7)
40	11.8 (10.7)	60.8 (12.2)
60	12.4 (14.4)	70.2 (13.2)
80	10.9 (14.1)	72.5 (12.9)
100	13.7 (10.9)	67.9 (17.7)
120	10.0 (9.8)	68.0 (16.8)
140	20.2 (14.3)	72.7 (18.4)
160	11.6 (9.5)	72.0 (16.9)
180	11.0 (10.8)	73.9 (19.2)
200	12.7 (10.6)	78.0 (19.4)
220	12.3 (10.2)	77.1 (20.1)
240	10.3 (9.8)	78.1 (18.3)
260	12.7 (11.4)	79.5 (19.4)



(a) $d_{min} = 10$min.



(b) $d_{min} = 40$min.

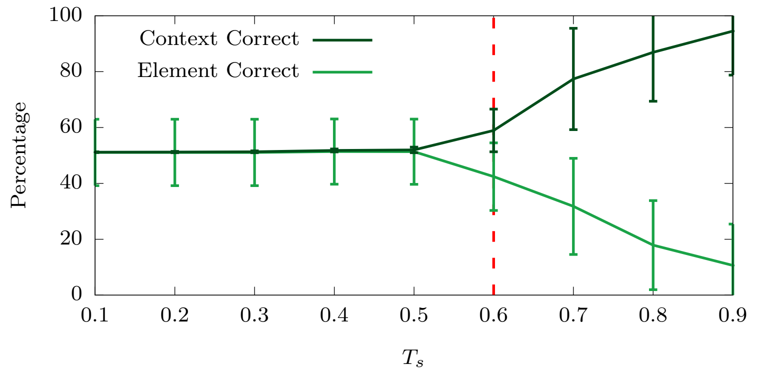
Figure 7.8: Comparison of different predictive techniques, considering extracted locations, land usage elements, and contexts. Locations are extracted using the *thresholding* technique.

Table 7.2: Comparison of different predictive techniques as plotted in Figure 7.8, with standard deviations in brackets.

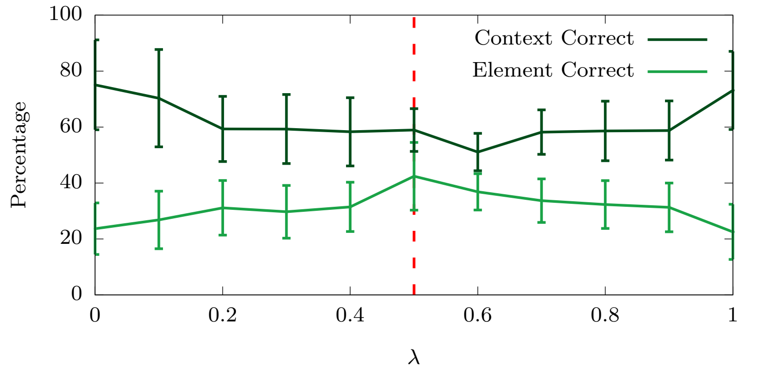
d_{min}	**10**	**40**
Location SVM	46.8 (8.89)	49.1 (8.71)
Element SVM	44.6 (5.39)	53.0 (8.20)
Element PCT	42.2 (6.59)	53.8 (8.48)
Context PCT (Element)	36.3 (7.58)	48.9 (12.2)
Context PCT (Context)	54.3 (6.29)	60.8 (10.74)

PCT. Again, allowing for contextual prediction, the PCT outperforms the other approaches.

The impact of two of the remaining parameters, T_s and λ, for context prediction using the PCT are shown in Figure 7.9 for the Warwick dataset, and in Figure A.5 (Appendix C) for MDC. The *Selection Threshold*, T_s, specifies the predictive confidence required to follow a node through to its child when traversing the Context Tree. A higher value for T_s will mean that children are less likely to be followed, instead returning contextual predictions, leading to fewer



(a) Selection Threshold (T_s).



(b) Semantic Weighting (λ).

Figure 7.9: The effect of parameters T_s and λ on context prediction, where $d_{min} = 10$min. The dashed red lines show the values used for previous figures.

element correct predictions, but more context correct ones. In Figure 7.9a, the variance of context correct predictions is 0 until the context and element correct lines diverge as no context-only predictions are made until this point. The PCT is only returning element predictions. The *Semantic Weighting* parameter, λ, is used when clustering Context Trees. A value of 0 uses only the feature similarity for determining contextual clusters, while a value of 1 uses only the semantic similarity. The highest proportion of element correct predictions can be seen when $\lambda = 0.5$; moving away from this in either direction lowers the accuracy of the predictor. Although this is made up for in context correct predictions, the indication is that as λ is moved towards 0 or 1, the contextual relationships are less useful for determining which element a user will interact with, and consequently are likely to be less representative of meaningful contexts, meaning that the context correct predictions are likely to be less useful.

As discussed previously, the results presented so far have all come from the Warwick dataset as this has a high level of coverage. In order to demonstrate the applicability of the PCT to other data, we also use the 10 users of the MDC dataset for comparison; this is shown in Figure 7.10 and Table 7.3. Although the MDC data contains periods of missing data, where truncated latitude and longitude values are removed (as discussed in Section 3.1.4), the trends are consistent with previous results. Contextual PCT prediction outperforms the other approaches, with SVM and the PCT in element prediction mode performing similarly.

Throughout this thesis, and as described in Chapter 3, we have been using a form of the MDC dataset where data points that have had their latitude and longitude values truncated to preserve privacy have been removed from the dataset. In Section 3.1.4 we posited that this would be more representative of real-world applications as missing data would be expected in most geospatial datasets, but that artificially limiting the accuracy of data may have unexpected consequences. In order to ensure that this assumption is correct, we have performed the same experiments using the MDC dataset without having removed the truncated periods. A selection of these results are shown in Appendix D, demonstrating similar trends between the data both with and without these truncated periods. However, Figure 7.11 shows a direct comparison of the predictive accuracies attained by using the MDC dataset both with and without these truncated periods as a basis for prediction through the LUI procedure and the PCT. The figure demonstrates that both predicting land usage elements (extracted through the LUI procedure and predicted with SVMs) and predicting elements through the PCT has significantly higher accuracy when including the truncated periods than when removing such periods. The reason for this is that the truncated periods cover vast amounts of time (a total of 3,552 hours across all considered users), and with no variance between latitude

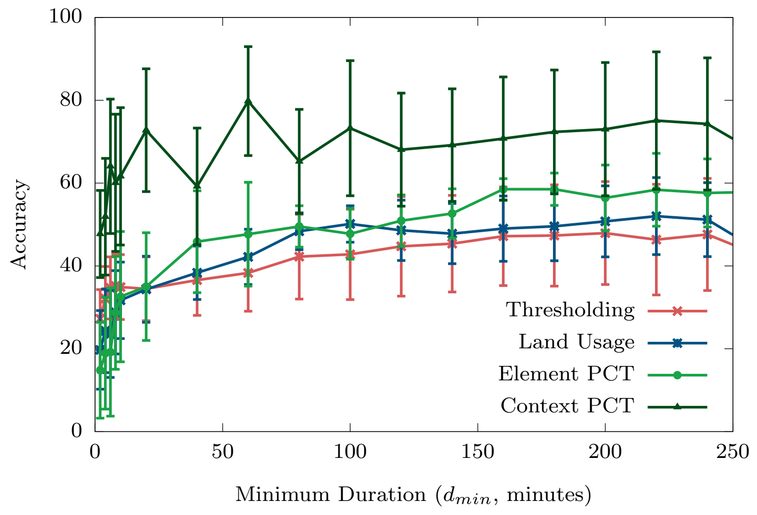
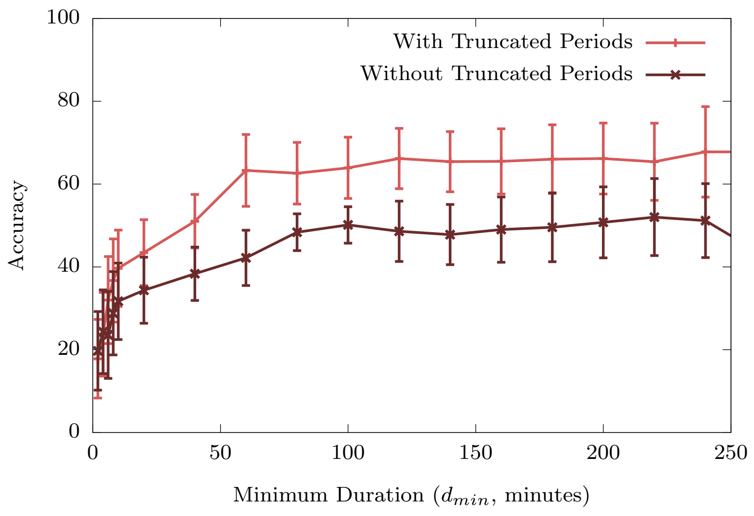


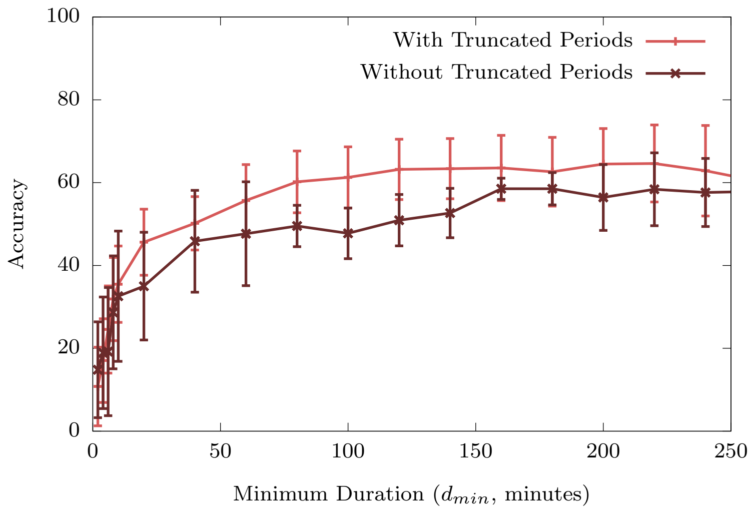
Figure 7.10: Predictive accuracies for the different prediction techniques for the MDC dataset.

Table 7.3: Comparison of different predictive techniques for the MDC dataset, with standard deviation shown in brackets.

d_{min}	**Thresholding**	**Land Usage**	**Element PCT**	**Context PCT**
2	27.3 (7.0)	19.7 (9.5)	14.8 (11.6)	**47.7 (10.5)**
4	32.1 (7.7)	24.3 (10.1)	18.9 (13.5)	**51.9 (14.1)**
6	34.7 (7.5)	23.6 (10.5)	19.2 (15.5)	**64.1 (16.2)**
8	35.2 (7.8)	28.8 (10.0)	28.7 (13.6)	**60.1 (16.6)**
10	34.9 (7.8)	31.7 (9.2)	25.6 (15.7)	**61.6 (16.6)**
20	34.5 (7.7)	34.4 (8.0)	30.0 (13.0)	**72.8 (14.8)**
40	36.6 (8.5)	38.4 (6.5)	36.8 (12.3)	**59.2 (14.1)**
60	38.3 (9.2)	42.2 (6.7)	38.7 (12.5)	**79.8 (13.2)**
80	42.2 (10.2)	48.4 (4.5)	49.5 (5.0)	**65.2 (12.6)**
100	42.8 (10.9)	50.1 (4.4)	47.7 (6.1)	**73.3 (16.3)**
120	44.7 (12.0)	48.6 (7.3)	50.9 (6.2)	**68.1 (13.7)**
140	45.4 (11.7)	47.8 (7.3)	52.7 (6.0)	**69.2 (13.6)**
160	47.2 (11.9)	49.0 (7.9)	58.5 (2.6)	**70.7 (14.9)**
180	47.3 (12.2)	49.6 (8.3)	58.5 (3.9)	**72.3 (15.0)**
200	47.9 (12.4)	50.8 (8.6)	56.4 (7.9)	**73.0 (16.1)**
220	46.4 (13.4)	52.0 (9.3)	58.4 (8.8)	**75.1 (16.6)**
240	47.6 (13.5)	51.2 (8.9)	57.6 (8.2)	**74.3 (15.9)**
260	42.7 (14.4)	43.8 (10.9)	43.9 (16.0)	**67.1 (19.5)**



(a) Land Usage Prediction.



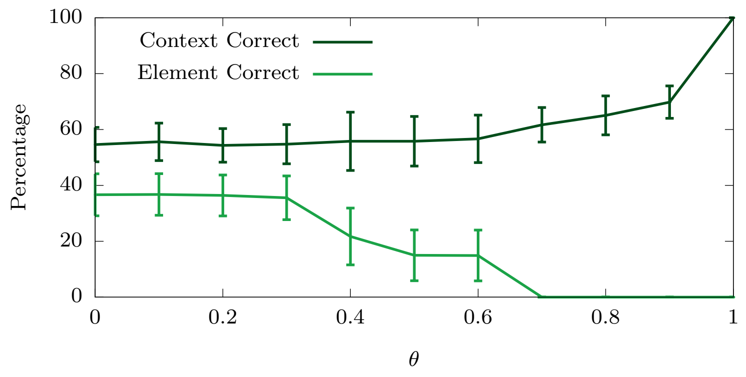
(b) Element PCT.

Figure 7.11: Comparisons of predictive accuracies for the MDC data with and without truncated periods for land usage and element PCT prediction.

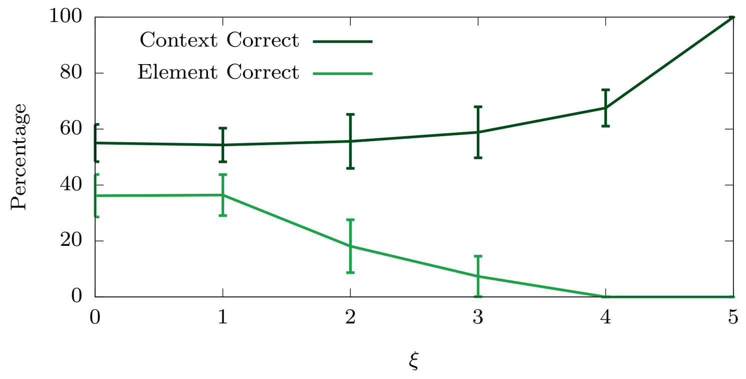
and longitude values, each such region covered by truncated data maps directly to a single land usage element. These elements represent significant amounts of time where the person was likely to have been at home or work, locations that are visited frequently, and are thus easier to predict than other elements, increasing the average predictive accuracy by a significant amount. While it is not possible to say exactly what the average predictive accuracy would be for the MDC data if the longitude and latitude values had not been truncated, it is likely to be higher than the accuracies reported throughout this thesis which used the MDC data discarding such periods, perhaps bringing the predictions in line with those made over the Warwick dataset. These results demonstrate that our decision to remove such periods was justified in order to prevent artificially inflating the accuracy of the presented techniques, and therefore the MDC results in this thesis present a baseline for the accuracy that can be expected from the presented techniques. Further results obtained from the MDC dataset while retaining the truncated periods can be found in Appendix D.

Pruning

The Context Tree generation procedure (Chapter 6), includes an approach for pruning nodes to reduce storage and processing requirements. This process takes two parameters, θ and ξ, where θ specifies a threshold between 0 and 1 for a node to be pruned, and ξ assigns a storage overhead to each node in the tree. Figure 7.12 shows the impact of these parameters on predictive accuracy for the Warwick dataset, with MDC results shown in Figure A.6 (Appendix C). A larger value of either parameter leads to more nodes being removed from the tree, resulting in fewer element correct predictions (as the leaf nodes corresponding to the elements are removed), but an increased number of context correct predictions. If using the Context Tree for predictive applications, through the PCT, these results indicate that a reduced size tree comes at a trade-off of reduced element correct predictive accuracy.



(a) Pruning Threshold (θ).

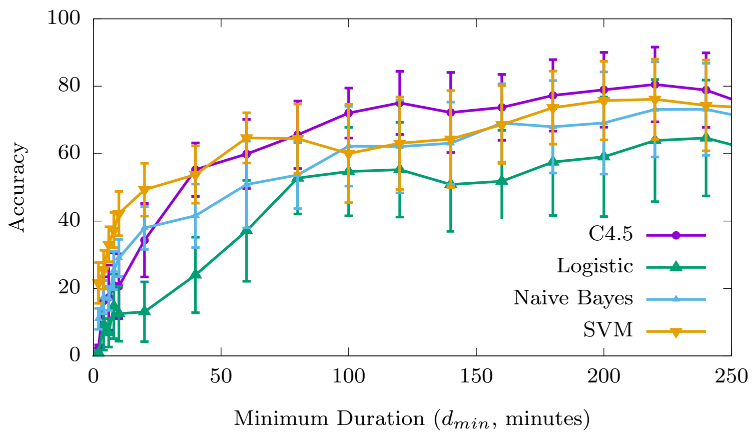


(b) Storage Overhead (ξ).

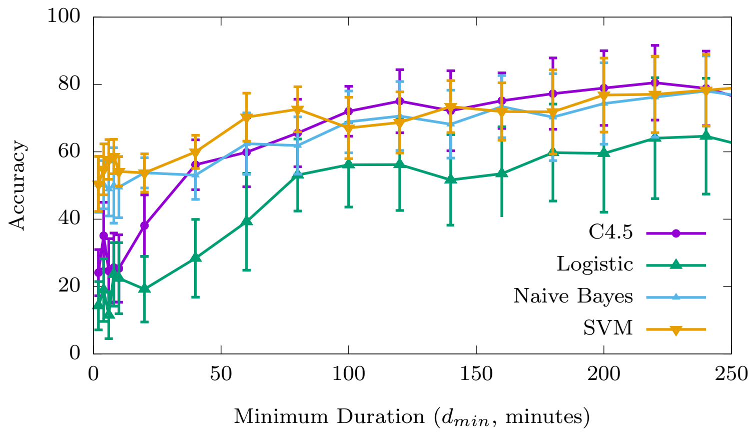
Figure 7.12: Predictive accuracies observed when using pruned Context Trees.

Classification Models

As discussed in Section 7.3.1, the results presented so far have been determined from PCTs trained using an SVM classifier in each node. While SVMs are capable of determining confidence values for each classification, through logistic regression, Figure 7.13 shows the predictive accuracy observed when using other probabilistic classification techniques as the classifiers in each node for the Warwick dataset (with similar MDC results in Figure A.7, Appendix C). Specifically, we test *C4.5*, a decision tree learning algorithm, *Logistic Regression* and *Naive Bayes* in addition to the *SVM*. These models were selected as representative examples of widely-used types of classification models. The same data as in Figure 7.13 is shown in Tables 7.4 and 7.5, where the best performing



(a) Single element PCT, element correct.



(b) Single context PCT, context correct.

Figure 7.13: Predictive accuracies obtained when using different probabilistic models as classifiers in the PCT.

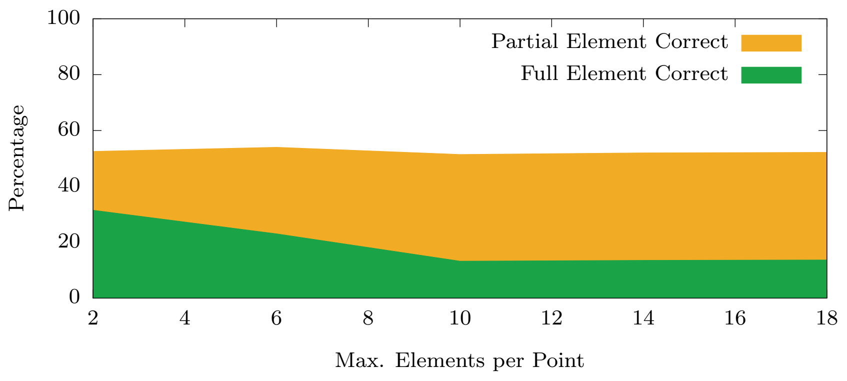
technique is shown in bold for each value of d_{min}. From these results, it is clear that there is no single algorithm that consistently outperforms all others, however, *Logistic Regression* is consistently beaten, and *Naive Bayes* has only a single case when it performs best, and only then it is within the margin of error of another technique, indicating that it too can likely be discarded. This leaves SVMs and *C4.5*, where for lower values of d_{min}, SVMs always perform best, but can be beaten by *C4.5* for higher values.

Table 7.4: Accuracies achieved using different classification models in the PCT when predicting elements, with the highest score in bold, and standard deviations in brackets, for each value of d_{min}.

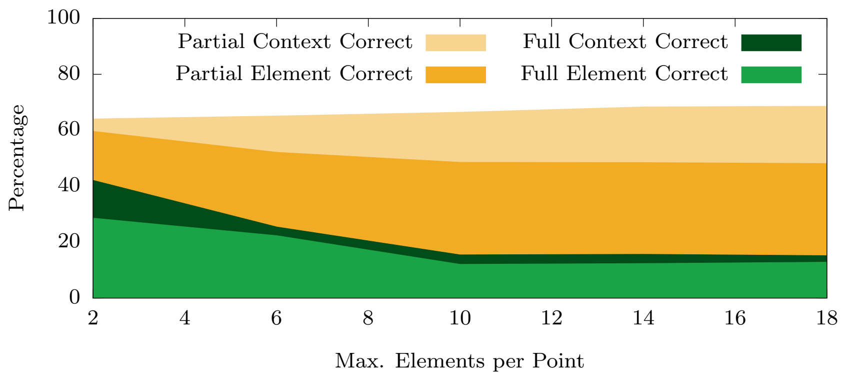
d_{min}	**C4.5**	**Logistic**	**Naive Bayes**	**SVM**
2	2.3 (0.9)	0.9 (0.6)	11.0 (3.1)	**21.7 (6.1)**
4	16.3 (7.1)	9.3 (7.6)	17.4 (4.9)	**25.6 (5.8)**
6	17.3 (9.6)	6.9 (4.2)	17.0 (3.5)	**33.1 (5.2)**
8	20.9 (9.7)	14.7 (9.6)	25.3 (5.7)	**37.3 (5.2)**
10	20.6 (9.6)	12.5 (8.1)	29.0 (5.5)	**42.2 (6.6)**
20	34.3 (10.9)	13.1 (8.9)	38.0 (6.4)	**49.3 (7.8)**
40	55.2 (7.9)	24.0 (11.2)	41.6 (9.4)	**53.8 (8.5)**
60	59.9 (10.2)	37.1 (15.0)	50.8 (12.9)	**64.7 (7.4)**
80	**65.6 (10.0)**	52.7 (10.6)	53.7 (10.0)	64.4 (10.3)
100	**72.0 (7.4)**	54.7 (13.1)	62.2 (11.8)	60.0 (14.6)
120	**75.0 (9.4)**	55.2 (14.1)	62.1 (13.8)	63.1 (13.7)
140	**72.2 (11.9)**	50.9 (13.9)	63.1 (12.2)	64.3 (14.4)
160	**73.7 (9.8)**	51.8 (15.1)	69.1 (11.6)	68.6 (11.5)
180	**77.3 (10.6)**	57.5 (15.8)	68.0 (13.7)	73.6 (10.8)
200	**78.9 (11.1)**	59.1 (17.7)	69.1 (15.1)	75.7 (11.6)
220	**80.5 (11.1)**	63.9 (18.1)	73.1 (14.1)	76.1 (11.9)
240	**78.9 (11.0)**	64.6 (17.2)	73.2 (13.7)	74.3 (13.5)
260	**73.5 (14.4)**	60.6 (19.2)	70.0 (16.4)	73.4 (14.8)

Table 7.5: Accuracies achieved using different classification models in the PCT when predicting contexts, with the highest score in bold, and standard deviations in brackets, for each value of d_{min}.

d_{min}	**C4.5**	**Logistic**	**Naive Bayes**	**SVM**
2	24.1 (6.8)	14.3 (7.2)	50.3 (8.3)	**50.5 (8.2)**
4	35.1 (9.9)	18.9 (9.3)	50.2 (7.1)	**54.8 (7.6)**
6	24.6 (9.6)	11.5 (7.0)	49.0 (8.0)	**57.7 (5.9)**
8	25.6 (10.3)	23.6 (9.4)	50.0 (11.2)	**58.6 (5.1)**
10	25.3 (10.0)	22.5 (10.6)	49.5 (9.1)	**54.2 (4.4)**
20	38.1 (9.1)	19.2 (9.7)	**53.8 (4.5)**	53.7 (5.7)
40	56.2 (7.4)	28.4 (11.5)	53.1 (7.2)	**60.0 (4.9)**
60	59.9 (10.2)	39.2 (14.4)	62.4 (9.1)	**70.3 (7.1)**
80	65.6 (10.0)	53.1 (10.7)	61.9 (8.5)	**72.6 (6.7)**
100	**72.0 (7.4)**	56.2 (12.6)	68.9 (9.2)	67.1 (9.1)
120	**75.0 (9.4)**	56.2 (13.6)	70.6 (10.2)	68.7 (9.1)
140	72.2 (11.9)	51.7 (13.4)	68.2 (10.1)	**73.4 (7.7)**
160	**75.2 (8.2)**	53.5 (13.9)	73.4 (9.2)	72.0 (8.5)
180	**77.3 (10.6)**	59.8 (14.4)	70.3 (12.9)	71.9 (12.5)
200	**78.9 (11.1)**	59.5 (17.5)	74.3 (12.1)	76.9 (11.0)
220	**80.5 (11.1)**	64.1 (18.0)	76.3 (11.9)	77.1 (11.3)
240	**78.9 (11.0)**	64.6 (17.2)	78.1 (10.4)	78.3 (10.6)
260	74.6 (13.8)	60.9 (19.0)	75.9 (12.4)	**79.5 (10.4)**



(a) Multi-element.



(b) Multi-context.

Figure 7.14: The effect of n on predictive accuracy for the multi-element Context Tree.

Multi-element Prediction

The PCT is capable of predicting multiple elements and contexts at the same time. This may be useful in instances where a user is within, for instance, a building that is contained within another building (e.g. a shop within a shopping centre). Evaluation of predictions made by PCTs that allow multiple such elements to be associated with a single time is conducted in accordance with the metrics defined in Section 7.4.2. Multi-element prediction is shown in Figure 7.14a (and Table 7.6) for different maximum numbers of elements per point. Increasing the number of elements decreases the ability for the PCT to identify exactly which elements are being interacted with; however, the *partial*

Table 7.6: Accuracy of the PCT when predicting multiple elements over the Warwick dataset, with standard deviation shown in brackets.

Maximum Elements per Point	**Full Element Correct**	**Partial Element Correct**
2	31.4 (12.4)	52.5 (7.4)
6	23.0 (13.4)	54.0 (11.7)
10	13.2 (9.8)	51.4 (11.0)
14	13.5 (10.7)	52.0 (11.0)
18	13.7 (12.0)	52.2 (11.8)

Table 7.7: Accuracy of the PCT when predicting multiple contexts over the Warwick dataset, with standard deviation shown in brackets.

Maximum Elements per Point	**Full Element Correct**	**Full Context Correct**	**Partial Element Correct**	**Partial Context Correct**
2	28.6 (13.2)	42.1 (10.6)	59.6 (7.3)	64.1 (3.8)
6	22.3 (13.5)	25.4 (3.1)	52.1 (12.1)	65.1 (8.9)
10	12.1 (9.7)	15.5 (3.9)	48.6 (12.1)	66.4 (10.9)
14	12.3 (10.6)	15.7 (4.1)	48.4 (12.0)	68.3 (11.5)
18	12.9 (11.9)	15.2 (3.0)	48.1 (13.1)	68.6 (12.3)

value remains fairly consistent, demonstrating that the PCT typically gets some of the predictions correct on occasions when it cannot predict all elements correctly. Figure 7.14b (and Table 7.7) shows the same graph, but for multi-context prediction where *full element correct* indicates that the set of elements being interacted with was predicted correctly, and *full context correct* represents times when a prediction covers all correct elements through a parent context. *Partial element correct* indicates that some elements were correctly predicted, but either additional elements were included in the prediction or some elements were overlooked. *Partial context correct* means that some contexts that were predicted were correct, but again, not all elements are covered by a context or additional contexts are predicted. The graph shows similar results to Figure 7.14a, where an increase in n reduces the number of *full element correct* predictions, but these are made up for with *partial element* and *context* predictions.

7.5 Summary and Conclusion

This work has explored the potential for extending the Context Tree data structure to perform hierarchical classification of land usage elements and contexts from trajectories collected about an individual. The Predictive Context Tree (PCT) is a hierarchical classification model that trains and classifies instances in a top-down approach, starting at the root node and following children until an overall classification is reached. The entire model can be built using only geospatial trajectories and a land usage dataset.

Through a comparison with predictions made over locations extracted from existing techniques, and land usage elements identified through the LUI procedure proposed in Chapter 5, the evaluation in this chapter demonstrates the applicability of the PCT. Specifically, the predictive accuracies achieved exceed those made over extracted locations and are on a par with the accuracies achieved in predicting land usage elements, presented previously. The PCT's primary benefit over these techniques is its additional ability to predict contexts, offering utility to applications when a prediction for a specific element has low confidence or when a contextual prediction would be useful. This is true of many applications where, for example, predicting that a user will be immersed within a 'shopping' or 'work' context is of more importance than knowing exactly which building a user will be in. The additional utility afforded by the PCT can provide the basis for constructing smart applications and services that understand the future contexts of individuals, while requiring only the collection of geospatial trajectories from the users.

CHAPTER 8
Discussion and Conclusion

This thesis has explored the potential that geospatial trajectories can offer to the understanding and prediction of human behaviour through machine learning techniques. Towards this aim, we have presented several novel approaches and techniques, including the Gradient-based Visit Extractor (GVE) algorithm and Land Usage Identification (LUI) procedure, alongside the Context Tree data structure and Predictive Context Tree (PCT) classification model. Using these techniques, we have an improved ability to understand past actions through identifying regions where individuals spend time, and the physical features they interact with; we also have an improved ability to predict the future movements and actions of individuals through locations, land usage elements and contexts. Combined, we are better able to understand and predict the movement of individuals, based on inference from geospatial trajectories. This chapter summarises and reviews the contributions made throughout this thesis, along with their limitations, and presents a discussion of possible avenues for further research in this area.

Throughout this work we have made several contributions to the field of geospatial machine learning, focusing on the discovery of knowledge from trajectories. These include the extraction and prediction of contexts, locations and interactions from geographic features. Specifically, the contributions are:

1. **The Gradient-based Visit Extractor (GVE) Algorithm:** An algorithm for the extraction of periods of low mobility, referred to as *visits*, from geospatial trajectories that can handle noisy data while overcoming many drawbacks of existing approaches.
2. **The Land Usage Identification (LUI) Procedure:** A procedure for

the identification of land usage elements, that represent geographic features, with which a user has interacted, to replace extracted locations.

3. **The Context Tree:** A data structure and associated generation technique that identifies and stores contexts from augmented geospatial trajectories.

4. **The Predictive Context Tree (PCT):** An application of the Context Tree as a hierarchical classifier that predicts both contexts and interactions with a demonstrated increase in utility over existing approaches.

8.1 Contribution Summary and Future Work

This section revisits the problem statement and contributions listed in Chapter 1 by discussing how well each contribution meets its aims. In addition, the limitations of each of the techniques are discussed and avenues for further work are presented and explored.

1. **Improving on current algorithms for identifying low mobility in geospatial trajectories for the purpose of identifying locations meaningful to the individual.**

   The **Gradient-based Visit Extractor (GVE)**, introduced in Chapter 4, was developed to achieve this task. Overcoming several drawbacks of existing approaches, including improved handling of noise in data, and not assuming evenly timesliced data, the GVE algorithm is capable of extracting *visits*, i.e. periods of low mobility, from geospatial trajectories. The algorithm is presented in detail and a thorough evaluation of the properties of the extracted visits is conducted over real-world trajectories, with specific comparisons to existing approaches, namely *thresholding* and the *Spatio-Temporal Activity (STA) extraction* algorithm. This evaluation not only establishes the applicability of the GVE algorithm to the task of identifying periods of low mobility from geospatial trajectories, but it also demonstrates the foundation afforded for identifying locations that are meaningful to users, with results indicating increased representativeness of locations over existing techniques.

The GVE algorithm's primary weakness is the range of parameters required for it to operate: while these parameters allow for tuning the algorithm, not all parameters map neatly to real-world properties (e.g. radius of visits), as with some existing approaches. The parameter d_{min} was introduced to allow specifying a minimum duration of a visit to consider, and t_{max} specifies the maximum duration between two consecutive points for them to be included in the same visit. However, the parameters α and β, which scale a function used for selecting a threshold, above which a visit is marked as having ended, and n_{points}, specifying the maximum number of points in the buffer being considered, have a less well-defined relationship to properties of the visits extracted. To mitigate this limitation somewhat, Chapter 4 also proposed the use of mathematical optimisation to automatically select and tune parameters for a given application. Focusing on location prediction as a sample application, a metric is presented that assigns a cost to each set of parameters based on the size of extracted locations and accuracy of predictions made over these locations. This metric enables the optimisation procedure to determine parameters consistent with the goals of location prediction. While this reduces the burden placed on application developers to select parameters, it is a computationally intensive task. Future work on the GVE algorithm could therefore aim to reduce the domain knowledge needed to select parameters, or to reduce the complexity of automatic parameter selection techniques, while maintaining the utility of the algorithm itself.

2. **Developing a technique for the identification of geographic features with which an entity interacts (e.g. specific buildings).**

   In Chapter 5 we proposed the **Land Usage Identification (LUI)** procedure, a method of augmenting geospatial trajectories with land usage elements extracted from a dataset. These elements in the augmented trajectories are scored and filtered, and then summarised, to identify a sequence of *interactions* made by the user that mirror the visits to locations of previous work. As with the GVE algorithm, the LUI procedure is presented in depth and evaluated over real-world trajectories along with a sample application, that of predicting future interactions. Identifying elements that represent physical features in the world, instead of using extracted locations, provides applications with additional utility in the form of known properties of these elements, often including their shape, name and purpose. This additional information can become the basis for understanding what the user may have been doing, and is indeed used in this way in Chapters 6 and 7 to identify contexts in which individuals are immersed.

The results presented in Chapter 5 demonstrate the applicability of the LUI procedure and reveal that predicting over land usage elements can obtain higher predictive accuracies than extracted locations, for minimum visit durations of approximately 20 minutes or longer. Despite the advantages of using land usage elements as a basis for identifying where people spend time, there are some drawbacks: in particular, for an element to be identified as being interacted with by the user, it must exist in the dataset as a single feature. In reality, there are times when a person may consider something a location, but it is identified by multiple land usage elements. An example of this would be if children were to play in a park that borders a street; they may consider both the park and street as the area in which they play and would expect it to be treated as a single entity. The LUI procedure is capable of extracting either the whole park or street, but is not capable of considering the two together. For this reason, the decision of whether to use extracted locations or identified elements as a method to understand how an individual has spent their time will need to be carefully considered based on the goals of the application.

Additionally, the LUI procedure requires geospatial trajectories with associated *accuracy* values, as this value is used as a radius to consider when selecting appropriate land usage elements. Some geospatial datasets, such as GeoLife (as discussed in Chapter 3), do not include accuracy values, and others may have inaccurate or overly-pessimistic ones. In order to ensure that the LUI procedure can be used on a wide variety of datasets, it would be desirable to understand the implications of using either a fixed value, or determining an approximate value based on other factors (e.g. the amount of movement between trajectory points); further work in this area is required to explore this possibility. One final issue with the procedure as outlined in Chapter 5 is that land usage datasets are typically extremely large, especially if they cover whole cities or countries. Augmenting trajectories, therefore, can require significant processing power to locate appropriate elements within the dataset. While it would be possible to store a small dataset on a mobile device, in order to maintain up-to-date information and lessen the burden placed on the device, it is likely that the dataset would need to be hosted as an external service. In this case, any application of augmented trajectories must have an active data connection for the LUI procedure to operate. Requiring access to a land usage dataset is a fundamental requirement of the LUI procedure and is the source of many of its advantages over existing techniques, so removing this requirement is not possible, but methods to pre-filter datasets to reduce their size may need to be considered, depending upon the application.

3. **Establishing a data structure for summarising identified contexts from augmented geospatial trajectories to identify periods of time with similar goals, desires and intentions.**

   Identifying land usage elements that a person was likely to have been interacting with (Chapter 5) provides the foundation for understanding not only where people spend their time, but also the type of activity they may be performing. By having elements associated with properties, such as the name and function of buildings, or the type of roads, these properties provide the foundation for identifying contexts. To this end, Chapter 6 presented the **Context Tree**, a new hierarchical summary model paired with a procedure for contextually clustering land usage elements based on their semantics and the properties of their interactions. The resultant tree models the contexts that a user has been immersed in at multiple scales, with leaf nodes representing geographic features, and non-leaf nodes representing contexts. Time spent within any such context is likely to be indicative of time spent with similar goals. Such contexts are identified using the proposed Hybrid Contextual Distance (HCD) metric, that aims to find periods of time a user spent interacting with elements with similar properties and function, or elements interacted with in a similar manner.

The generation procedure, along with the constructed Context Trees, are evaluated both with respect to partial ground truths and naive clustering approaches, but also with respect to the expected properties of data at each stage in the Context Tree generation procedure. Both types of evaluation are conducted over real-world trajectories. In addition to this, a method of pruning Context Trees is proposed that aims to maximise information while reducing the storage and processing required to search a constructed Context Tree, designed for applications where resources may be limited.

The Context Tree's main drawback relates to the complexity of hierarchical clustering, where the distance between each pair of elements must be calculated to find the closest pair or pairs, $O(n^2)$. The technique does, however, place a lower burden on the users as only geospatial trajectories are required to be collected from them, in contrast to other context identification techniques which require data from multiple low-level sensors (e.g. heart-rate) to identify contexts. Improving on the Context Tree should therefore focus on reducing the computational complexity of the hierarchical clustering algorithm, possibly by using heuristics to guide the selection of pairs of elements that have low distances, instead of requiring a comparison of all possible pairs.

In addition to reducing the complexity of clustering Context Trees, retraining the trees with new data is a task which has not been considered. At present, if additional trajectories were to be collected from the user, a new Context Tree must be built from scratch to incorporate it. With real-time collection approaches, this would require a great deal of processing power to train trees continuously. More likely, the trees would be trained periodically and thus would be based on out-of-date data for the majority of the time. Exploring techniques to train Context Trees continuously as new data arrives should therefore also gain focus when considering future avenues of research for the Context Tree summary model.

Finally, while we consider an application of context trees, that of prediction in Chapter 7, there are many other possible applications. For example, the identification of similarities between trees from the same user could be used to identify repeating patterns, or similarities between trees from different users could be used to identify users with behaviours in common. Calculating the similarity, or distance, between two trees has been explored for other domains (e.g. using the tree edit-distance metric [Klein, 1998]), but further work in this area could develop more specific metrics for this task.

4. **Evaluating the Context Tree through a predictive model for forecasting the future contexts and location interactions of individuals.**

   Building upon the work of the two previous chapters, Chapter 7 presented the **Predictive Context Tree (PCT)**. The PCT is an extension to the Context Tree data structure (Chapter 6) that leverages the hierarchical nature of the Context Tree to construct a classification model capable of predicting both future contexts and element interactions of individuals. This extension takes the form of training each non-root node of the Context Tree as a binary classifier that determines if a specific instance belongs in the subtree rooted at the node being trained; this is a *top-down* predictor. By building the PCT in this manner, the hierarchical relationship between contexts, as determined through contextual clustering as part of the Context Tree generation procedure, is learned by the model.

The utility of this application of Context Trees is demonstrated by a comparison to existing predictive approaches. Specifically, we have compared the PCT to the results of predicting over extracted locations and land usage elements using established techniques (presented in Chapter 5). In these comparisons, the PCT is shown to exceed the accuracy of predicting over extracted locations and match the accuracy of predicting over land usage elements using Support Vector Machines (SVMs). While the PCT is more complex, and therefore has a higher computational cost to construct than the SVM, the PCT is able to offer additional utility in the form of context predictions lacking from existing approaches. In times when predictive confidence in a specific land usage element is low, or times when the application is interested in what a user will do, but not where, a context prediction can be returned instead of a specific element. Factoring in such context predictions, the *PCT* outperforms the other techniques tested.

Although the PCT performs well when compared to other techniques, there are times when existing approaches, or even PCTs trained using alternate predictive models, can achieve higher accuracies. Better predictions can be made, for example, using only SVMs when considering small minimum interaction durations (low values of d_{min}). It would be desirable therefore to harness the higher accuracies achieved by such techniques by constructing a *hybrid* predictor that is capable of using multiple approaches to achieve the highest possible number of *element correct* predictions, while maintaining the ability to produce *context correct* predictions when no technique has enough confidence to predict a specific element. Additionally, the problem of retraining Context Trees was previously mentioned, and the same issue exists with the PCT, where if additional trajectories were to be brought into the system, the tree would need to be constructed from scratch. Extending the Context Tree and PCT to leverage additional data and training instances without requiring a new tree to be constructed is, therefore, very desirable.

In addition to avenues of future work that aim to remove or reduce the specific limitations of the approaches discussed above, expanding the focus of the techniques would present utility for additional domains. An example of this would be the consideration of additional sources of data for each of the techniques presented in this thesis. Currently, only trajectories collected from devices carried by individuals have been used for evaluating the proposed techniques, but trajectories of different sources exist (as discussed in Chapter 3). For example, tailoring the procedures for trajectories collected from vehicles would require additional work to consider the different properties present in the data, and would be likely to have slightly different goals. The GVE algorithm could be tailored towards identifying parking areas, where vehicles are left, and the LUI procedure could be used to determine the roads taken by the car, in addition to the type of feature visited by the driver. The PCT could then be used to predict what type of element the car is being driven towards, and this information leveraged to offer intelligent traffic avoidance, or perhaps to suggest an alternative location with the same function if the drive would be shorter.

8.2 Final Remarks

In this thesis, we have presented several new techniques, algorithms and procedures in the domain of geospatial machine learning. Specifically, we have improved upon the state-of-the-art for extracting visits from geospatial trajectories, by presenting an algorithm with improved ability to handle noise and with fewer limitations than existing works. We have extended the idea of visit extraction by considering land usage elements that represent geographic features a user was likely to have been interacting with, rather than focusing on identifying arbitrary shapes. These features have then gone on to form the basis of contextual clustering to determine periods of time when the user is likely to have had similar goals and intentions. Finally, these contextual clusters have formed the basis of a new prediction approach that considers not only where a person will be, but the likely context within which they will be immersed.

All of these processes have been thoroughly evaluated over two real-world datasets with results demonstrating the applicability of each stage, and increased accuracy and utility over existing techniques. With the increased availability of geospatial trajectories, in part due to the now-pervasive nature of location-aware devices, we have an unprecedented platform on which to build smarter and more tailored services to improve the lives of individuals and groups. The work in this thesis has provided additional tools and techniques on which such services can be built.

References

Abe, Naoki; Zadrozny, Bianca; and Langford, John. 2006. Outlier Detection by Active Learning. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 504–509, Philadelphia. doi: 10.1145/1150402.1150459.

Ackerman, Mark; Darrell, Trevor; and Weitzner, Daniel. 2001. Privacy in Context. *Human-Computer Interaction*, 16(2-4):167–176. doi: 10.1207/S15327051HCI16234_03.

Ahmad, Saif; Taskaya-Temizel, Tugba; and Ahmad, Khurshid. 2004. Summarizing Time Series: Learning Patterns in 'Volatile' Series. In *Proceedings of the 5th International Conference on Intelligent Data Engineering and Automated Learning*, pages 523–532, Exeter. doi: 10.1007/978-3-540-28651-6_77.

Akoush, Sherif and Sameh, Ahmed. 2007. Bayesian Learning of Neural Networks for Mobile User Position Prediction. In *Proceedings of the 12th International Euro-Par Conference on Parallel Processing*, pages 1234–1239, Honolulu. doi: 10.1109/ICCCN.2007.4317989.

Alhindi, Azhar; Kruschwitz, Udo; Fox, Chris; and Albakour, M-Dyaa. 2015. Profile-Based Summarisation for Web Site Navigation. *ACM Transactions on Information Systems*, 33 (1):4:1–4:39. doi: 10.1145/2699661.

Alvarez-Garcia, Juan Antonio; Ortega, Juan Antonio; Gonzalez-Abril, Luis; and Velasco, Francisco. 2010. Trip Destination Prediction Based on Past GPS Log Using a Hidden Markov Model. *Expert Systems With Applications*, 37(12):8166–8171. doi: 10.1016/j.eswa.2010.05.070.

Anagnostopoulos, Christos; Tsounis, Athanasios; and Hadjiefthymiades, Stathes. 2006. Context Awareness in Mobile Computing Environments. *Wireless Personal Communications*, 42(3):445–464. doi: 10.1007/s11277-006-9187-6.

Andrienko, Gennady; Andrienko, Natalia; Hurter, Christophe; Rinzivillo, Salvatore; and Wrobel, Stefan. 2011. From Movement Tracks Through Events to Places: Extracting and Characterizing Significant Places from Mobility Data. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*, pages 161–170, Providence. doi: 10.1109/VAST.2011.6102454.

Andrienko, Gennady; Andrienko, Natalia; Fuchs, Georg; Olteanu Raimond, Ana-Maria; Symanzik, Juergen; and Ziemlicki, Cezary. 2013. Extracting Semantics of Individual Places from Movement Data by Analyzing Temporal Patterns of Visits. In *Proceedings of the ACM*

SIGSPATIAL International Conference on Advances in Geographic Information Systems, pages 9–16, Orlando. doi: 10.1145/2534848.2534851.

Angiulli, Fabrizio and Fassetti, Fabio. 2009. DOLPHIN: An Efficient Algorithm for Mining Distance-Based Outliers in Very Large Datasets. *ACM Transactions on Knowledge Discovery from Data*, 3(1):4:1–4:57. doi: 10.1145/1497577.1497581.

Ashbrook, Daniel and Starner, Thad. 2002. Learning Significant Locations and Predicting User Movement with GPS. In *Proceedings of the 6th International Symposium on Wearable Computers*, pages 101–108, Seattle. doi: 10.1109/ISWC.2002.1167224.

Ashbrook, Daniel and Starner, Thad. 2003. Using GPS to Learn Significant Locations and Predict Movement Across Multiple Users. *Personal and Ubiquitous Computing*, 7(5):275–286. doi: 10.1007/s00779-003-0240-0.

Ashman, Helen; Brailsford, Tim; and Brusilovsky, Peter. 2009. Personal Services: Debating the Wisdom of Personalisation. In *Proceedings of the 8th International Conference on Advances in Web Based Learning*, pages 1–11, Aachen. doi: 10.1007/978-3-642-03426-8_1.

Assam, Roland and Seidl, Thomas. 2013. BodyGuards: A Clairvoyant Location Predictor Using Frequent Neighbors and Markov Model. In *Proceedings of the 10th IEEE International Conference on Ubiquitous Intelligence and Computing*, pages 25–32, Vietri sul Mere. doi: 10.1109/UIC-ATC.2013.18.

Assam, Roland and Seidl, Thomas. 2014. Context-based Location Clustering and Prediction using Conditional Random Fields. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*, pages 1–10, Melbourne. doi: 10.1145/2677972.2677989.

Bamis, Athanasios and Savvides, Andreas. 2010. Lightweight Extraction of Frequent Spatio-Temporal Activities from GPS Traces. In *Proceedings of the 31st Real-Time Systems Symposium*, pages 281–291, San Diego. doi: 10.1109/RTSS.2010.33.

Bamis, Athanasios and Savvides, Andreas. 2011. Exploiting Human State Information to Improve GPS Sampling. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 32–37, Seattle. doi: 10.1109/PERCOMW.2011.5766898.

Bao, Jie; Zheng, Yu; Wilkie, David; and Mokbel, Mohamed. 2015. Recommendations in Location-based Social Networks: A Survey. *GeoInformatica*, 19(3):525–565. doi: 10.1007/s10707-014-0220-8.

Bao, Tengfei; Cao, Huanhuan; Chen, Enhong; Tian, Jilei; and Xiong, Hui. 2011. An Unsupervised Approach to Modelling Personalized Contexts of Mobile Users. *Knowledge and Information Systems*, 31(2):345–370. doi: 10.1007/s10115-011-0417-1.

Bashir, Faisal I; Khokhar, Ashfaq A; and Schonfeld, Dan. 2006. Real-Time Motion Trajectory-Based Indexing and Retrieval of Video Sequences. *IEEE Transactions on Multimedia*, 9 (1):58–65. doi: 10.1109/TMM.2006.886346.

Bashir, Faisal I; Khokhar, Ashfaq A; and Schonfeld, Dan. 2007. Object Trajectory-Based Activity Classification and Recognition Using Hidden Markov Models. *IEEE Transactions on Image Processing*, 16(7):1912–1919. doi: 10.1109/TIP.2007.898960.

Bayir, Murat Ali; Demirbas, Murat; and Eagle, Nathan. 2009. Discovering Spatiotemporal Mobility Profiles of Cellphone Users. In *Proceedings of the International Symposium on World of Wireless, Mobile and Multimedia Networks & Workshops*, pages 1–9, Kos. doi: 10.1109/WOWMOM.2009.5282489.

Beresford, Alastair and Stajano, Frank. 2003. Location Privacy in Pervasive Computing. *IEEE Pervasive Computing*, 2(1):46–55. doi: 10.1109/MPRV.2003.1186725.

Beresford, Alastair and Stajano, Frank. 2004. Mix Zones: User Privacy in Location-aware Services. In *Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pages 127–131, Orlando. doi: 10.1109/PERCOMW.2004.1276918.

Bertsimas, Dimitris and Tsitsiklis, John. 1993. Simulated Annealing. *Statistical Science*, 8 (1):10–15. doi: 10.1214/ss/1177011077.

Bhyri, Nitin; Kidiyoor, Gautham V; Varun, Subramaniam K; Kalambur, Subramaniam; Sitaram, Dinkar; and Kollengode, Chid. 2015. Predicting the Next Move: Determining Mobile User Location Using Semantic Information. In *Proceedings of the 2015 International Conference on Advances in Computing, Communications and Informatics*, pages 2359–2365, Kochi. doi: 10.1109/ICACCI.2015.7275971.

Bilurkar, Pradeep; Rao, Narasimha; Krishna, Gowri; and Jain, Ravi. 2002. Application of Neural Network Techniques for Location Predication in Mobile Networking. In *Proceedings of the 9th International Conference on Neural Information Processing*, pages 2157–2161, Whistler. doi: 10.1007/978-3-642-40988-2.

Birant, Derya and Kut, Alp. 2007. ST-DBSCAN: An Algorithm for Clustering Spatial–Temporal Data. *Data & Knowledge Engineering*, 60(1):208–221. doi: 10.1016/j.datak.2006.01.013.

Boutsis, Ioannis and Kalogeraki, Vana. 2016. Location Privacy for Crowdsourcing Applications. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 694–705, Heidelberg. doi: 10.1145/2971648.2971741.

Brand, Matthew; Oliver, Nuria; and Pentland, Alex. 1997. Coupled Hidden Markov Models for Complex Action Recognition. In *Proceedings of the 5th IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, San Juan. doi: 10.1109/CVPR.1997.609450.

Brax, Christoffer. 2008. Finding Behavioural Anomalies in Public Areas Using Video Surveillance Data. In *Proceedings of the 11th International Conference on Information Fusion*, pages 1–8, Cologne. doi: 10.1117/12.800095.

Breunig, Markus M.; Kriegel, Hans-Peter; Ng, Raymond T.; and Sander, Jörg. 2000. LOF: Identifying Density-Based Local Outliers. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 93–104, Dallas. doi: 10.1145/342009.335388.

Burbey, Ingrid and Martin, Thomas L. 2008. Predicting Future Locations Using Prediction-by-Partial-Match. In *Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, pages 1–6, San Francisco. doi: 10.1145/1410012.1410014.

Cao, Huiping; Mamoulis, Nikos; and Cheung, David. 2005. Mining Frequent Spatio-temporal Sequential Patterns. In *Proceedings of the 5th IEEE International Conference on Data Mining*, pages 82–89, New Orleans. doi: 10.1109/ICDM.2005.95.

Cao, Huiping; Mamoulis, Nikos; and Cheung, David. 2007. Discovery of Periodic Patterns in Spatiotemporal Sequences. *IEEE Transactions on Knowledge and Data Engineering*, 19 (4):453–467. doi: 10.1109/TKDE.2007.1002.

CEH. Land Cover Map, 2007. URL `http://www.ceh.ac.uk/services/land-cover-map-2007`.

Cesa-Bianchi, Nicolò; Gentile, Claudio; and Zaniboni, Luca. 2006. Hierarchical Classification: Combining Bayes with SVM. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 177–184, Pittsburgh. doi: 10.1145/1143844.1143867.

Chandola, Varun; Banerjee, Arindam; and Kumar, Vipin. 2009. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3):15–58. doi: 10.1145/1541880.1541882.

Chen, Chao; Zhang, Daqing; Castro, Pablo Samuel; Li, Nan; Sun, Lin; and Li, Shijian. 2011a. Real-time Detection of Anomalous Taxi Trajectories from GPS Traces. In *Proceedings of the 8th International Conference on Mobile and Ubiquitous Systems*, pages 63–74, Copenhagen. doi: 10.1007/978-3-642-30973-1_6.

Chen, Ling; Lv, Mingqi; and Chen, Gencai. 2010. A System for Destination and Future Route Prediction Based on Trajectory Mining. *Pervasive and Mobile Computing*, 6(6):657–676. doi: 10.1016/j.pmcj.2010.08.004.

Chen, Ling; Lv, Mingqi; Ye, Qian; Chen, Gencai; and Woodward, John. 2011b. A Personal Route Prediction System Based on Trajectory Data Mining. *Information Sciences*, 181(7): 1264–1284. doi: 10.1016/j.ins.2010.11.035.

Chen, Peng; Lu, Zhao; and Gu, Junzhong. 2009. Vehicle Travel Time Prediction Algorithm Based on Historical Data and Shared Location. In *Proceedings of the 5th International Joint Conference on Networked Computing, Information Management and Service and Digital Content, Multimedia Technology and its Applications*, pages 1632–1637, Seoul. doi: 10.1109/NCM.2009.138.

Chiang, Meng-Fen; Zhu, Wen-Yuan; and Yu, Philip S. 2013. Distant-Time Location Prediction in Low-Sampling-Rate Trajectories. In *Proceedings of the 14th IEEE International Conference on Mobile Data Management*, pages 117–126, Milan. doi: 10.1109/MDM.2013.22.

Cho, Eunjoon; Myers, Seth A; and Leskovec, Jure. 2011. Friendship and Mobility: User Movement in Location-based Social Networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1082–1090, San Diego. doi: 10.1145/2020408.2020579.

Cho, Sung-Bae. 2016. Exploiting Machine Learning Techniques for Location Recognition and Prediction with Smartphone Logs. *Neurocomputing*, 176:98–106. doi: 10.1016/j.neucom.2015.02.079.

Chon, Yohan and Cha, Hojung. 2011. Lifemap: A Smartphone-based Context Provider for Location-based Services. *Pervasive Computing*, 10(2):58–67. doi: 10.1109/MPRV.2011.13.

Chon, Yohan; Talipov, Elmurod; Shin, Hyojeong; and Cha, Hojung. 2011. Mobility Prediction-based Smartphone Energy Optimization for Everyday Location Monitoring. In *Proceedings of the 17th International Conference on World Wide Web*, pages 82–85, Seattle. doi: 10.1145/2070942.2070952.

Chon, Yohan; Shin, Hyojeong; Talipov, Elmurod; and Cha, Hojung. 2012. Evaluating Mobility Models for Temporal Prediction with High-Granularity Mobility Data. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 206–212, Lugano. doi: 10.1109/PerCom.2012.6199868.

Choudhury, Tanzeem; Consolvo, Sunny; Harrison, Beverly; Hightower, Jeffrey; LeGrand, Louis; Rahimi, Ali; Rea, Adam; Borriello, Gaetano; Hemingway, Bruce; Klasnja, Predrag; Koscher, Karl; Landay, James A; Lester, Jonathan; Wyatt, Danny; and Haehnel, Dirk. 2008. The Mobile Sensing Platform: An Embedded Activity Recognition System. *Pervasive Computing*, 7(2):32–41. doi: 10.1109/MPRV.2008.39.

Comito, Carmela; Falcone, Deborah; and Talia, Domenico. 2016. Mining Human Mobility Patterns from Social Geo-tagged Data. *Pervasive and Mobile Computing (In Press)*. doi: 10.1016/j.pmcj.2016.06.005.

Cook, Deborah F; Ragsdale, Cliff T; and Major, Raymond L. 2000. Combining a Neural Network with a Genetic Algorithm for Process Parameter Optimization. *Engineering Applications of Artificial Intelligence*, 13(4):391–396. doi: 10.1016/S0952-1976(00)00021-X.

Cooper, Simon and Durrant-Whyte, Hugh. 1994. A Kalman Filter Model for GPS Navigation of Land Vehicles. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, pages 157–163, Munich. doi: 10.1109/IROS.1994.407396.

Cristianini, Nello and Shawe-Taylor, John. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press. ISBN 978-0521780193.

Dash, Manoranjan; Koo, Kee Kiat; Gomes, Joao Bartolo; Krishnaswamy, Shonali Priyadarsini; Rugeles, Daniel; and Shi-Nash, Amy. 2015. Next Place Prediction by Understanding Mobility Patterns. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communication Workshops*, pages 469–474, St. Louis. doi: 10.1109/PERCOMW.2015.7134083.

De Domenico, Manlio; Lima, Antonio; and Musolesi, Mirco. 2013. Interdependence and Predictability of Human Mobility and Social Interactions. *Pervasive and Mobile Computing*, 9(6):798–807. doi: 10.1016/j.pmcj.2013.07.008.

Do, Trinh Minh Tri and Gatica-Perez, Daniel. 2013. The Places of Our Lives: Visiting Patterns and Automatic Labeling from Longitudinal Smartphone Data. *IEEE Transactions on Mobile Computing*, 13(3):638–648. doi: 10.1109/TMC.2013.19.

Dumais, Susan and Chen, Hao. 2000. Hierarchical Classification of Web Content. In *Proceedings of the 23rd Annual International ACM Conference on Research and Development in Information Retrieval*, pages 256–263, Athens. doi: 10.1145/345508.345593.

Eagle, Nathan and Pentland, Alex Sandy. 2005. Reality Mining: Sensing Complex Social Systems. *Personal and Ubiquitous Computing*, 10(4):255–268. doi: 10.1007/s00779-005-0046-3.

Eagle, Nathan and Pentland, Alex Sandy. 2009. Eigenbehaviors: Identifying Structure in Routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066. doi: 10.1007/s00265-009-0739-0.

Endsley, Mica R. 1995. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64. doi: 10.1518/001872095779049543.

Endsley, Mica R. 2000. Theoretical Underpinnings of Situation Awareness: A Critical Review. In *M. R. Endsley & D. J. Garland (Eds.) Situation Awareness Analysis and Measurement*, pages 3–28. Lawrence Erlbaum Associates.

Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; and Xu, Xiaowei. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland.

ETSI. 1996. Digital Cellular Telecommunications System (Phase 2+); Physical Layer on the Radio Path; General Description. Technical report, European Telecommunications Standards Institute. URL `http://www.etsi.org/deliver/etsi_gts/05/0501/05.00.00_60/gsmts_0501v050000p.pdf`.

Farrahi, Katayoun and Gatica-Perez, Daniel. 2008a. Daily Routine Classification from Mobile Phone Data. In *Proceedings of the 5th International Workshop on Machine Learning for Multimodal Interaction*, pages 173–184, Utrecht. doi: 10.1007/978-3-540-85853-9_16.

Farrahi, Katayoun and Gatica-Perez, Daniel. 2008b. What Did You Do Today?: Discovering Daily Routines from Large-Scale Mobile Data. In *Proceeding of the 16th ACM International Conference on Multimedia*, pages 849–852, Vancouver. doi: 10.1145/1459359.1459503.

Farrahi, Katayoun and Gatica-Perez, Daniel. 2009. Learning and Predicting Multimodal Daily Life Patterns from Cell Phones. In *Proceedings of the International Conference on Multimodal Interfaces*, pages 277–280, Cambridge. doi: 10.1145/1647314.1647373.

Farrahi, Katayoun and Gatica-Perez, Daniel. 2010. Probabilistic Mining of Socio-Geographic Routines from Mobile Phone Data. *IEEE Journal of Selected Topics in Signal Processing*, 4(4):746–755. doi: 10.1109/JSTSP.2010.2049513.

Frohlich, Holger and Zell, Andreas. 2005. Efficient Parameter Selection for Support Vector Machines in Classification and Regression via Model-based Global Optimization. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1431–1436, Montreal. doi: 10.1109/IJCNN.2005.1556085.

Fukano, Jun; Mashita, Tomohiro; Hara, Takahiro; and Kiyokawa, Kiyoshi. 2013. A Next Location Prediction Method for Smartphones Using Blockmodels. In *Proceedings of the IEEE Conference on Virtual Reality*, pages 1–4, Orlando. doi: 10.1109/VR.2013.6549434.

Gao, Huiji; Tang, Jiliang; and Liu, Huan. 2012. Mobile Location Prediction in Spatio-Temporal Context. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, Newcastle.

Ge, Linlin; Han, Shaowei; and Rizos, Chris. 2000. Multipath Mitigation of Continuous GPS Measurements Using an Adaptive Filter. *GPS Solutions*, 4(2):19–30. doi: 10.1007/PL00012838.

Gerber, Matthew S. 2014. Predicting Crime Using Twitter and Kernel Density Estimation. *Decision Support Systems*, 61(C):115–125. doi: 10.1016/j.dss.2014.02.003.

Giannotti, Fosca; Nanni, Mirco; Pinelli, Fabio; and Pedreschi, Dino. 2007. Trajectory Pattern Mining. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 330–339, San Jose. doi: 10.1145/1281192.1281230.

Giremus, Audrey; Doucet, Arnaud; Calmettes, Vincent; and Jean-Yves, Tourneret. 2004. A Rao-Blackwellized Particle Filter for INS/GPS Integration. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 964–977, Montreal. doi: 10.1109/ICASSP.2004.1326707.

GMV. 2011. GLONASS Performances. Technical report, EESA Navipedia. URL `http://navipedia.net/index.php/GLONASS_Performances`.

Gogoi, Prasanta; Bhattacharyya, Dhruba K; Borah, Bhogeswar; and Kalita, Jugal K. 2011. A Survey of Outlier Detection Methods in Network Anomaly Identification. *The Computer Journal*, 54(4):570–588. doi: 10.1093/comjnl/bxr026.

Goh, Chong Yang; Dauwels, Justin; Mitrovic, Nikola; and Asif, Muhammad Tayyab. 2012. Online Map-matching Based on Hidden Markov Model for Real-time Traffic Sensing Applications. In *Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems*, pages 776–781, Anchorage. doi: 10.1109/ITSC.2012.6338627.

Gong, Yu; Li, Yong; Jin, Depeng; Su, Li; and Zeng, Lieguang. 2011. A Location Prediction Scheme Based on Social Correlation. In *Proceedings of the IEEE Vehicular Technology Conference*, pages 1–5, Yokohama. doi: 10.1109/VETECS.2011.5956736.

Gopal, Siddharth; Yang, Yiming; Bai, Bing; and Niculescu-Mizil, Alexandru. 2012. Bayesian Models for Large-scale Hierarchical Classification. In *Proceedings of 26th Conference on Neural Information Processing Systems*, pages 2411–2419, Lake Tahoe.

Gossen, Tatiana; Nitsche, Marcus; and Nürnberger, Andreas. 2013. Evolving Search User Interfaces. In *Proceedings of the 3rd European Workshop on Human-Computer Interaction and Information Retrieval*, pages 31–34, Dublin.

Grimes, John. 2008. Global Positioning System Standard Positioning Service Performance Standard. Technical report, United States Department of Defense. URL `http://www.gps.gov/technical/ps/2008-SPS-performance-standard.pdf`.

Gudmundsson, Joachim; van Kreveld, Marc; and Speckmann, Bettina. 2004. Efficient Detection of Motion Patterns in Spatio-temporal Data Sets. In *Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems*, pages 250–257, Washington DC. doi: 10.1145/1032222.1032259.

Guidotti, Riccardo; Trasarti, Roberto; and Nanni, Mirco. 2015. TOSCA: TwO-Steps Clustering Algorithm for Personal Locations Detection. In *Proceedings of the 23rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 38:1–38:10, Bellevue. doi: 10.1145/2820783.2820818.

Han, Jiawei. 2002. CLARANS: a Method for Clustering Objects for Spatial Data Mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016. doi: 10.1109/TKDE.2002.1033770.

Hariharan, Ramaswamy and Toyama, Kentaro. 2004. Project Lachesis: Parsing and Modeling Location Histories. In *Proceedings of the 3rd International Conference on Geographic Information Science*, pages 106–124, Adelphi. doi: 10.1007/978-3-540-30231-5_8.

Hazas, Mike; Scott, James; and Krumm, John. 2004. Location-Aware Computing Comes of Age. *IEEE Computer*, 37(2):95–97. doi: 10.1109/MC.2004.1266301.

He, Zengyou; Xu, Xiaofei; and Deng, Shengchun. 2003. Discovering Cluster-based Local Outliers. *Pattern Recognition Letters*, 24(9):1641–1650. doi: 10.1016/S0167-8655(03)00003-5.

Helal, Sumi; Winkler, Bryon; Lee, Choonhwa; Kaddoura, Youssef; Ran, Lisa; Giraldo, Carlos; Kuchibhotla, Sree; and Mann, William. 2003. Enabling Location-aware Pervasive Computing Applications for the Elderly. In *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications*, pages 531–536, Fort Worth. doi: 10.1109/PERCOM.2003.1192785.

Hoh, Baik; Gruteser, Marco; Xiong, Hui; and Alrabady, Ansaf. 2010. Achieving Guaranteed Anonymity in GPS Traces via Uncertainty-aware Path Cloaking. *IEEE Transactions on Mobile Computing*, 9(8):1089–1107. doi: 10.1109/TMC.2010.62.

Howard, Newton. 2002. *Theory of Intention Awareness in Tactical Military Intelligence: Reducing Uncertainty by Understanding the Cognitive Architecture of Intentions.* AuthorHouse, Bloomington.

Howard, Newton and Cambria, Erik. 2013. Intention Awareness: Improving Upon Situation Awareness in Human-centric Environments. *Human-centric Computing and Information Sciences*, 3(1):17. doi: 10.1186/2192-1962-3-9.

Hu, Jiawei; Wang, Yanfeng; and Zhang, Ya. 2015. IOHMM for Location Prediction with Missing Data. In *Proceedings of the IEEE International Conference on Data Science and Advanced Analytics*, pages 1–10, Paris. doi: 10.1109/DSAA.2015.7344851.

Huang, Wei; Li, Songnian; Liu, Xintao; and Ban, Yifang. 2015. Predicting Human Mobility with Activity Changes. 29(9):1569–1587. doi: 10.1080/13658816.2015.1033421.

Jain, Anil K; Murty, M Narasimha; and Flynn, P J. 1999. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323. doi: 10.1145/331499.331504.

Janoos, Firdaus; Singh, Shantanu; Irfanoglu, Okan; Machiraju, Raghu; and Parent, Richard. 2007. Activity Analysis Using Spatio-Temporal Trajectory Volumes in Surveillance Applications. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 3–10, Sacramento. doi: 10.1109/VAST.2007.4388990.

Jin, Chen; De-lin, Luo; and Fen-xiang, Mu. 2009. An improved ID3 decision tree algorithm. In *Proceedings of the 4th International Conference on Computer Science Education*, pages 127–130, Nanning. doi: 10.1109/ICCSE.2009.5228509.

Kaasinen, Eija. 2003. User Needs for Location-aware Mobile Services. *Personal and Ubiquitous Computing*, 7(1):70–79. doi: 10.1007/s00779-002-0214-7.

Kang, Jong Hee; Welbourne, William; Stewart, Benjamin; and Borriello, Gaetano. 2004. Extracting Places from Traces of Locations. In *Proceedings of the 2nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots*, pages 110–118, Philadelphia. doi: 10.1145/1024733.1024748.

Karimi, Hassan A. and Liu, Xiong. 2003. A Predictive Location Model for Location-based Services. In *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, pages 126–133, New Orleans. doi: 10.1145/956676.956693.

Kaufman, Leonard and Rousseeuw, Peter. 1987. Clustering by Means of Medoids. *Reports of the Faculty of Mathematics and Informatics*, 87(3):405–416.

Kim, Eunju; Helal, Sumi; and Cook, Diane. 2010. Human Activity Recognition and Pattern Discovery. *Pervasive Computing*, 9(1):48–53. doi: 10.1109/MPRV.2010.7.

Kirkpatrick, Scott; Gelatt, C. Daniel; and Vecchi, M. P. 1983. Optimization by Simulated Annealing. *Science*, 220(4598):671–680.

Kiukkonen, Niko; Blom, Jan; Dousse, Olivier; Gatica-Perez, Daniel; and Laurila, Juha. 2010. Towards Rich Mobile Phone Datasets: Lausanne Data Collection Campaign. In *Proceedings of the ACM International Conference on Pervasive Services*, pages 1–7, Berlin.

Klein, Philip N. 1998. Computing the Edit-Distance Between Unrooted Ordered Trees. In *Algorithms — ESA' 98*, volume 1461 of *LNCS*, pages 91–102. Springer. doi: 10.1007/3-540-68530-8_8.

Krumm, John and Horvitz, Eric. 2006. Predestination: Inferring Destinations from Partial Trajectories. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, pages 243–260, Orange County. doi: 10.1007/11853565_15.

Krumm, John and Rouhana, Dany. 2013. Placer: Semantic Place Labels from Diary Data. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 163–172, Zurich. doi: 10.1145/2493432.2493504.

Laurila, Juha K; Gatica-Perez, Daniel; Aad, Imad; Blom, Jan; Bornet, Olivier; Do, Trinh Minh Tri; Dousse, Olivier; Eberle, Julien; and Miettinen, Markus. 2012. The Mobile Data Challenge: Big Data for Mobile Computing Research. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, pages 1–8, Newcastle.

Laxhammar, Rikard and Falkman, Goran. 2011. Sequential Conformal Anomaly Detection in Trajectories Based on Hausdorff Distance. In *Proceedings of the 14th International Conference on Information Fusion*, pages 153–160, Chicago.

Laxhammar, Rikard and Falkman, Goran. 2014. Online Learning and Sequential Anomaly Detection in Trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1158–1173. doi: 10.1109/TPAMI.2013.172.

Le, Truc Viet; Liu, Siyuan; Lau, Hoong Chuin; and Krishnan, Ramayya. 2015. Predicting Bundles of Spatial Locations from Learning Revealed Preference Data. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1121–1129, Istanbul.

Lee, Jae-Gil; Han, Jiawei; and Whang, Kyu-Young. 2007. Trajectory Clustering: a Partition-and-group Framework. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, pages 593–604, Beijing. doi: 10.1145/1247480.1247546.

Lee, Seon-Woo and Mase, Kenji. 2002. Activity and Location Recognition Using Wearable Sensors. *Pervasive Computing*, 1(3):24–32. doi: 10.1109/MPRV.2002.1037719.

Lee, Sungjun; Choi, Yerim; Lim, Seongmin; and Park, Jonghun. 2015. A Spatio-Temporal Distance Based Clustering Approach for Discovering Significant Places from Trajectory

Data. In *Proceedings of the International Conference on Computer Science, Data Mining & Mechanical Engineering*, pages 130–134, Bangkok. doi: 10.15242/IIE.E0415071.

Lei, Po-Ruey; Shen, Tsu-Jou; Peng, Wen-Chih; and Su, Ing-Jiunn. 2011. Exploring Spatial-temporal Trajectory Model for Location Prediction. In *Proceedings of the 12th IEEE International Conference on Mobile Data Management*, pages 58–67, Lulea. doi: 10.1109/MDM.2011.61.

Lemlouma, Tayeb and Layaida, Nabil. 2004. Context-aware Adaptation for Mobile Devices. In *Proceedings of the IEEE International Conference on Mobile Data Management*, pages 106–111, Berkeley. doi: 10.1109/MDM.2004.1263048.

Lester, Jonathan; Choudhury, Tanzeem; Kern, Nicky; Borriello, Gaetano; and Hannaford, Blake. 2005. A Hybrid Discriminative/Generative Approach for Modeling Human Activities. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 766–772, Edinburgh.

Li, Quannan; Zheng, Yu; Xie, Xing; Chen, Yukun; Liu, Wenyu; and Ma, Wei-Ying. 2008. Mining User Similarity Based on Location History. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 34:1–34:10, Irvine. doi: 10.1145/1463434.1463477.

Liao, Lin; Fox, Dieter; and Kautz, Henry. 2007a. Extracting Places and Activities from GPS Traces Using Hierarchical Conditional Random Fields. *The International Journal of Robotics Research*, 26(1):119–134. doi: 10.1177/0278364907073775.

Liao, Lin; Patterson, Donald J; Fox, Dieter; and Kautz, Henry. 2007b. Learning and Inferring Transportation Routines. *Artificial Intelligence*, 171(5-6):311–331. doi: 10.1016/j.artint.2007.01.006.

LiKamWa, Robert; Liu, Yunxin; Lane, Nicholas D; and Zhong, Lin. 2011. Can Your Smartphone Infer your Mood? In *Proceedings of the 2nd International Workshop on Sensing Applications on Mobile Phones*, pages 1–5, Seattle.

LiKamWa, Robert; Liu, Yunxin; Lane, Nicholas D; and Zhong, Lin. 2013. MoodScope: Building a Mood Sensor from Smartphone Usage Patterns. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, pages 389–402, Taipei. doi: 10.1145/2462456.2464449.

Liu, Siyuan; Liu, Yunhuai; Ni, Lionel M; Fan, Jianping; and Li, Minglu. 2010. Towards Mobility-based Clustering. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, pages 919–928, Washington DC. doi: 10.1145/1835804.1835920.

Liu, Siyuan; Cao, Huanhuan; Li, L; and Zhou, MengChu. 2013. Predicting Stay Time of Mobile Users with Contextual Information. *IEEE Transactions on Automation Science and Engineering*, 10(4):1026–1036. doi: 10.1109/TASE.2013.2259480.

Liu, Siyuan; Chen, Lei; and Ni, Lionel M. 2014. Anomaly Detection from Incomplete Data. *ACM Transactions on Knowledge Discovery from Data*, 9(2):11:1–11:22. doi: 10.1145/2629668.

Liu, Wei; Zheng, Yu; Chawla, Sanjay; Yuan, Jing; and Xie, Xing. 2011. Discovering Spatio-Temporal Causal Interactions in Traffic Data Streams. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, pages 1010–1018, San Diego. doi: 10.1145/2020408.2020571.

Ljungstrand, Peter. 2001. Context Awareness and Mobile Phones. *Personal and Ubiquitous Computing*, 5(1):58–61. doi: 10.1007/s007790170032.

MacQueen, James. 1967. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Math, Statistics, and Probability*, pages 281–297, Berkeley.

Marmasse, Natalia and Schmandt, Chris. 2000. Location-Aware Information Delivery withComMotion. In *Handheld and Ubiquitous Computing*, pages 157–171. Springer. doi: 10.1007/3-540-39959-3_12.

Matekenya, Dunstan; Ito, Masaki; Tobe, Yoshito; Shibasaki, Ryosuke; and Sezaki, Kaoru. 2015. MoveSense: Spatio-temporal Clustering Technique for Discovering Residence Change in Mobile Phone Data. In *Proceedings of the 6th ACM SIGSPATIAL International Workshop on GeoStreaming*, pages 59–68, Seattle. doi: 10.1145/2833165.2833175.

Mathew, Wesley; Raposo, Ruben; and Martins, Bruno. 2012. Predicting Future Locations with Hidden Markov Models. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 911–918, Pittsburgh. doi: 10.1145/2370216.2370421.

Messing, Ross; Pal, Chris; and Kautz, Henry. 2009. Activity Recognition Using the Velocity Histories of Tracked Keypoints. In *Proceedings of the 12th International Conference on Computer Vision*, pages 104–111, Kyoto. doi: 10.1109/ICCV.2009.5459154.

Miller, George. 1995. WordNet: a Lexical Database for English. *Communications of the ACM*, 38(11):39–41. doi: 10.1145/219717.219748.

Mitchell, Thomas M. 1997. *Machine Learning*. McGraw-Hill. ISBN 978-0071154673.

Mohamed, A H and Schwarz, Klaus-Peter. 1999. Adaptive Kalman Filtering for INS/GPS. *Journal of Geodesy*, 73(4):193–203. doi: 10.1007/s001900050236.

Montoliu, Raul and Gatica-Perez, Daniel. 2010. Discovering Human Places of Interest from Multimodal Mobile Phone Data. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*, pages 12:1–12:10, Limassol. doi: 10.1145/1899475.1899487.

Morris, Brendan Tran and Trivedi, Mohan Manubhai. 2011. Trajectory Learning for Activity Understanding: Unsupervised, Multilevel, and Long-Term Adaptive Approach. *IEEE Transactions on Pattern Analysis and Machine Learning*, 33(11):2287–2301. doi: 10.1109/TPAMI.2011.64.

Moscato, Pablo. 1989. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. *Caltech Concurrent Computation Program*, 826:1–67.

Movebank. Movebank Database, 2017. URL `https://www.movebank.org`.

Musolesi, Mirco. 2014. Big Mobile Data Mining: Good or Evil? *IEEE Internet Computing*, 18(1):78–81. doi: 10.1109/MIC.2014.2.

Noulas, Athanasios; Scellato, Salvatore; Lathia, Neal; and Mascolo, Cecilia. 2012. Mining User Mobility Features for Next Place Prediction in Location-Based Services. In *Proceedings of the 12th IEEE International Conference on Data Mining*, pages 1038–1043, Brussels. doi: 10.1109/ICDM.2012.113.

Ochieng, Washington Y; Quddus, Mohammed A; and Noland, Robert B. 2003. Map-matching in Complex Urban Road Networks. *Brazilian Journal of Cartography*, 55(2):1–14.

Oliveira, Eduardo; Rezende, Mucelli; Viana, Aline Carneiro; Sarraute, Carlos; Brea, Jorge; and Alvarez-Hamelin, Ignacio. 2016. On the Regularity of Human Mobility. *Pervasive and Mobile Computing (In Press)*. doi: 10.1016/j.pmcj.2016.04.005.

OpenStreetMap. About OpenStreetMap, 2017a. URL `https://www.openstreetmap.org/about`.

OpenStreetMap. Overpass API, 2017b. URL `https://wiki.openstreetmap.org/wiki/Overpass_API`.

Ordnance Survey. OS MasterMap, 2017. URL `https://www.ordnancesurvey.co.uk/business-and-government/products/mastermap-products.html`.

Palma, Andrey Tietbohl; Bogorny, Vania; Kuijpers, Bart; and Alvares, Luis Otavio. 2008. A Clustering-based Approach for Discovering Interesting Places in Trajectories. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, pages 863–868, Fortaleza. doi: 10.1145/1363686.1363886.

Patterson, Donald J; Liao, Lin; Fox, Dieter; and Kautz, Henry. 2003. Inferring High-Level Behavior from Low-Level Sensors. In *Proceedings of the 5th International Conference on Ubiqutous Computing*, pages 73–89, Seattle. doi: 10.1007/978-3-540-39653-6_6.

Petzold, Jan; Bagci, Faruk; Trumler, Wolfgang; and Ungerer, Theo. 2006. Comparison of Different Methods for Next Location Prediction. In *Proceedings of the 12th International Euro-Par Conference on Parallel Processing*, pages 909–918, Dresden. doi: 10.1007/11823285_96.

Pink, Oliver and Hummel, Britta. 2008. A Statistical Approach to Map Matching Using Road Network Geometry, Topology and Vehicular Motion Constraints. In *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems*, pages 862–867, Beijing. doi: 10.1109/ITSC.2008.4732697.

Pirttikangas, Susanna; Fujinami, Kaori; and Nakajima, Tatsuo. 2006. Feature Selection and Activity Recognition from Wearable Sensors. In *Proceedings of the 3rd International Symposium on Ubiqutous Computing Systems*, pages 516–527, Seoul. doi: 10.1007/11890348_39.

Popescu, Elvira. 2010. Adaptation Provisioning with Respect to Learning Styles in a Web-based Educational System: An Experimental Study. *Journal of Computer Assisted Learning*, 26(4):243–257. doi: 10.1111/j.1365-2729.2010.00364.x.

Porikli, Fatih. 2004. Trajectory Distance Metric Using Hidden Markov Model Based Representation. In *Proceedings of the IEEE European Conference on Computer Vision*, pages 1–7, Zurich.

Qiu, Disheng; Papotti, Paolo; and Blanco, Lorenzo. 2013. Future Locations Prediction with Uncertain Data. In *Machine Learning and Knowledge Discovery in Databases*, volume 8188 of *LNCS*, pages 417–432. Springer. doi: 10.1007/978-3-642-40988-2_27.

Quddus, Mohammed A; Ochieng, Washington Yotto; Zhao, Lin; and Noland, Robert B. 2003. A General Map Matching Algorithm for Transport Telematics Applications. *GPS Solutions*, 7(3):157–167. doi: 10.1007/s10291-003-0069-z.

Quinlan, Ross. 1996. Bagging, Boosting, and C4. 5. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 725–730, Portland.

Rajaraman, Anand and Ullman, David. 2011. *Mining of Massive Datasets.* Cambridge University Press. ISBN 1107015359.

Ravi, Nishkam; Dandekar, Nikhil; Mysore, Preetham; and Littman, Michael L. 2005. Activity Recognition from Accelerometer Data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence*, pages 1541–1546, Pittsburgh.

Resnik, Philip. 1999. Semantic Similarity in a Taxonomy: An Information-based Measure and its Application to Problems of Ambiguity in Natural Language. *Journal of Artificial Intelligence Research*, 11:95–130. doi: 10.1613/jair.514.

Robusto, C Carl. 1957. The Cosine-Haversine Formula. *The American Mathematical Monthly*, 64(1):38–40.

Rosen, Olov and Medvedev, Alexander. 2012. An On-line Algorithm for Anomaly Detection in Trajectory Data. In *Proceedings of the American Control Conference*, pages 1117–1122, Montreal. doi: 10.1109/ACC.2012.6315346.

Rossi, Luca; Walker, James; and Musolesi, Mirco. 2015a. Spatio-temporal Techniques for User Identification by Means of GPS Mobility Data. *EPJ Data Science*, 4(11):1–16. doi: 10.1140/epjds/s13688-015-0049-x.

Rossi, Luca; Williams, Matthew J; Stich, Christoph; and Musolesi, Mirco. 2015b. Privacy and the City: User Identification and Location Semantics in Location-Based Social Networks. In *Proceedings of the 9th International AAAI Conference on Web and Social Media*, pages 1–10, Oxford.

Rousu, Juho; Saunders, Craig; Szedmak, Sandor; and Shawe-Taylor, John. 2005. Learning Hierarchical Multi-category Text Classification Models. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 744–751, Bonn. doi: 10.1145/1102351.1102445.

Roy, Abishek; Bhaumik, Soumya K. Das; Bhattacharya, Amiya; Basu, Kalyan; Cook, Diane J.; and Das, Sajal K. 2003. Location Aware Resource Management in Smart Homes. In *Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications*, pages 481–488, Fort Worth. doi: 10.1109/PERCOM.2003.1192773.

Rubio, Ginés; Pomares, Héctor; Rojas, Ignacio; and Herrera, Luis Javier. 2010. A heuristic method for parameter selection in LS-SVM: Application to time series prediction. *International Journal of Forecasting*, 27(3):725739. doi: 10.1016/j.ijforecast.2010.02.007.

Russell, Stuart and Norvig, Peter. 2009. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 4th edition. ISBN 978-0130803023.

Saini, Lalit M; Aggarwal, Sanjeev K; and Kumar, Ashwani. 2010. Parameter Optimisation Using Genetic Algorithm for Support Vector Machine-based Price-forecasting Model in National Electricity Market. *Generation, Transmission & Distribution*, 4(1):36–14. doi: 10.1049/iet-gtd.2008.0584.

Seo, Won-Il and Lim, Jae-Hyun. 2016. Implementation of Context Prediction System Based on Event Recurrence Time. *Cluster Computing*, 19(3):1671–1682. doi: 10.1007/s10586-016-0612-7.

Shen, Jianan and Cheng, Tao. 2016. A Framework for Identifying Activity Groups from Individual Space-time Profiles. *International Journal of Geographical Information Science*, pages 1785–1805. doi: 10.1080/13658816.2016.1139119.

Shiode, Shino and Shiode, Narushige. 2014. Microscale Prediction of Near-Future Crime Concentrations with Street-Level Geosurveillance. *Geographical Analysis*, 46(4):435–455. doi: 10.1111/gean.12065.

Shiode, Shino; Shiode, Narushige; Block, Richard; and Block, Carolyn R. 2015. Space-time Characteristics of Micro-scale Crime Occurrences: An Application of a Network-based Space-time Search Window Technique for Crime Incidents in Chicago. *International Journal of Geographical Information Science*, 29(5):697–719. doi: 10.1080/13658816.2014.968782.

Shye, Alex; Scholbrock, Benjamin; Memik, Gokhan; and Dinda, Peter A. 2010. Characterizing and Modeling User Activity on Smartphones: Summary. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 375–376, New York. doi: 10.1145/1811099.1811094.

Siła-Nowicka, Katarzyna; Vandrol, Jan; Oshan, Taylor; Long, Jed A; Demšar, Urška; and Fotheringham, A Stewart. 2015. Analysis of Human Mobility Patterns from GPS Trajectories and Contextual Information. *International Journal of Geographical Information Science*, 30(5):881–906. doi: 10.1080/13658816.2015.1100731.

Silla Jr, Carlos N and Freitas, Alex A. 2011. A Survey of Hierarchical Classification Across Different Application Domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72. doi: 10.1007/s10618-010-0175-9.

Sillito, Rowland and Fisher, Robert. 2008. Semi-supervised Learning for Anomalous Trajectory Detection. In *Proceedings of the British Machine Vision Conference*, pages 1–10, Leeds.

SNAP. Brightkite Dataset Information, 2008. URL `http://snap.stanford.edu/data/loc-brightkite.html`.

SNAP. Gowalla Dataset Information, 2010. URL `http://snap.stanford.edu/data/loc-gowalla.html`.

Steichen, Ben; Ashman, Helen; and Wade, Vincent. 2012. A Comparative Survey of Personalised Information Retrieval and Adaptive Hypermedia Techniques. *Information Processing & Management*, 48(4):698–724. doi: 10.1016/j.ipm.2011.12.004.

Subramanya, Amarnag; Raj, Alvin; Bilmes, Jeff; and Fox, Dieter. 2006. Recognizing Activities and Spatial Context Using Wearable Sensors. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 494–502, Cambridge.

Thanh, Nguyen and Phuong, Tu Minh. 2007. A Gaussian Mixture Model for Mobile Location Prediction. In *Proceedings of the 9th International Conference on Advanced Communication Technology*, pages 914–919. doi: 10.1109/ICACT.2007.358509.

Thomason, Alasdair; Griffiths, Nathan; and Leeke, Matthew. 2015a. Extracting Meaningful User Locations from Temporally Annotated Geospatial Data. In *Internet of Things: IoT Infrastructures*, volume 151 of *LNICST*, pages 84–90. Springer. doi: 10.1007/978-3-319-19743-2_13.

Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2015b. Parameter Optimisation for Location Extraction and Prediction Applications. In *Proceedings of the 2015 IEEE International Conference on Pervasive Intelligence and Computing*, pages 2173–2180, Liverpool. doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.322.

Thomason, Alasdair; Leeke, Matthew; and Griffiths, Nathan. 2015c. Understanding the Impact of Data Sparsity and Duration for Location Prediction Applications. In *Internet of Things: IoT Infrastructures*, volume 151 of *LNICST*, pages 192–197. Springer. doi: 10.1007/978-3-319-19743-2_29.

Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016a. Identifying Locations from Geospatial Trajectories. *Journal of Computer and System Sciences*, 82(4):566–581. doi: 10.1016/j.jcss.2015.10.005.

Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016b. Context Trees: Augmenting Geospatial Trajectories with Context. *ACM Transactions on Information Systems*, 35 (2):14:1–14:37. doi: 10.1145/2978578.

Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016c. Predicting Interactions and Contexts with Context Trees. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 46:1–46:4, San Francisco. doi: 10.1145/2996913.2996993.

Thomason, Alasdair; Griffiths, Nathan; and Sanchez, Victor. 2016d. The Predictive Context Tree: Predicting Contexts and Interactions. *arXiv:1610.01381 (pre-print)*.

Trasarti, Roberto; Guidotti, Riccardo; Monreale, Anna; and Giannotti, Fosca. 2015. MyWay: Location Prediction via Mobility Profiling. *Information Systems (In Press)*, pages 1–18. doi: 10.1016/j.is.2015.11.002.

Tsai, Yi-Han; Chang, Fan-Ren; and Yang, Wen-Chih. 2004. GPS Fault Detection and Exclusion Using Moving Average Filters. *IEE Proceedings - Radar, Sonar and Navigation*, 151 (4):240–247. doi: 10.1049/ip-rsn:20040728.

Van Kasteren, Tim; Noulas, Athanasios; Englebienne, Gwenn; and Krose, Ben. 2008. Accurate Activity Recognition in a Home Setting. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, pages 1–9, Seoul. doi: 10.1145/1409635.1409637.

Vinciarelli, Alessandro; Esposito, Anna; André, Elisabeth; Bonin, Francesca; Chetouani, Mohamed; Cohn, Jeffrey F; Cristani, Marco; Fuhrmann, Ferdinand; Gilmartin, Elmer; Hammal, Zakia; Heylen, Dirk; Kaiser, Rene; Koutsombogera, Maria; Potamianos, Alexandros; Renals, Steve; Riccardi, Giuseppe; and Salah, Albert Ali. 2015. Open Challenges in Modelling, Analysis and Synthesis of Human Behaviour in Human–Human and Human–Machine Interactions. *Cognitive Computation*, 7(4):397–413. doi: 10.1007/s12559-015-9326-z.

Vintan, Lucian; Gellert, Arpad; Petzold, Jan; and Ungerer, Theo. 2004. Person Movement Prediction Using Neural Networks. In *Proceedings of the 1st Workshop on Modeling and Retrieval of Context*, pages 1–12, Ulm.

Vukovic, Marin; Vujnovic, Goran; and Grubisic, Darko. 2009. Adaptive User Movement Prediction for Advanced Location-aware Services. In *Proceedings of the 17th International Conference on Software, Telecommunications Computer Networks*, pages 343–347, Hvar.

Wang, Dan; Xiang, Zheng; and Fesenmaier, Daniel R. 2014. Adapting to the Mobile World: A Model of Smartphone Use. *Annals of Tourism Research*, 48:11–26. doi: 10.1016/j.annals.2014.04.008.

Wang, Hui; Lenz, Henning; Szabo, Andrei; Bamberger, Joachim; and Hanebeck, Uwe D. 2007. WLAN-Based Pedestrian Tracking Using Particle Filters and Low-Cost MEMS Sensors. In *Proceedings of the 4th Workshop on Positioning, Navigation and Communication*, pages 1–7, Hannover. doi: 10.1109/WPNC.2007.353604.

Wang, Jingjing and Prabhala, Bhaskar. 2012. Periodicity Based Next Place Prediction. In *Proceedings of the Nokia Mobile Data Challenge (MDC) Workshop in Conjunction with Pervasive*, Newcastle.

White, Christopher E; Bernstein, David; and Kornhauser, Alain L. 2000. Some Map Matching Algorithms for Personal Navigation Assistants. *Transportation Research Part C: Emerging Technologies*, 8(1-6):91–108. doi: 10.1016/S0968-090X(00)00026-7.

Whitley, Darrell; Starkweather, Timothy; and Bogart, Chris. 1990. Genetic Algorithms and Neural Networks: Optimizing Connections and Connectivity. *Parallel Computing*, 14(3): 347–361. doi: 10.1016/0167-8191(90)90086-O.

Wolfson, Ouri and Yin, Huabei. 2003. Accuracy and Resource Consumption in Tracking and Location Prediction. In *Proceedings of the 8th International Symposium Advances in Spatial and Temporal Databases*, pages 325–343, Santorini Island. doi: 10.1007/978-3-540-45072-6_19.

Wu, Yun; Yang, Feng; Ren, Min; and Han, Le. 2016. Precision Advertising Based on the Scene of Trajectory Mining. *International Journal of Database Theory and Application*, 9 (4):135–142. doi: 10.14257/ijdta.2016.9.4.12.

Wu, Zhibiao and Palmer, Martha. 1994. Verb Semantics and Lexical Selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, pages 133–138. doi: 10.3115/981732.981751.

Xiao, Xiangye; Zheng, Yu; Luo, Qiong; and Xie, Xing. 2012. Inferring Social Ties Between Users with Human Location History. *Journal of Ambient Intelligence and Humanized Computing*, 5(1):3–19. doi: 10.1007/s12652-012-0117-z.

Xiong, Yibing and Lin, Huiping. 2012. Routine Based Analysis for User Classification and Location Prediction. In *Proceedings of the 9th International Conference on Ubiquitous Intelligence & Computing*, pages 96–103, Fukuoka. doi: 10.1109/UIC-ATC.2012.46.

Xu, Mengwen; Wang, Dong; and Li, Jian. 2016. DESTPRE: A Data-driven Approach to Destination Prediction for Taxi Rides. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 729–739, Heidelberg. doi: 10.1145/2971648.2971664.

Xue, Andy Yuan; Zhang, Rui; Zheng, Yu; Xie, Xing; Huang, Jin; and Xu, Zhenghua. 2013. Destination Prediction by Sub-Trajectory Synthesis and Privacy Protection Against Such Prediction. In *Proceedings of the 29th IEEE International Conference on Data Engineering*, pages 254–265, Brisbane. doi: 10.1109/ICDE.2013.6544830.

Yan, Zhixian; Chakraborty, Dipanjan; Parent, Christine; Spaccapietra, Stefano; and Aberer, Karl. 2013. Semantic Trajectories: Mobility Data Computation and Annotation. *ACM Transactions on Intelligent Systems and Technology*, 4(3). doi: 10.1145/2483669.2483682.

Yang, Bin; Fantini, Nicolas; and Jensen, Christian S. 2013. iPark: Identifying Parking Spaces from Trajectories. In *Proceedings of the 17th International Conference on World Wide Web*, pages 705–708, Genoa. doi: 10.1145/2452376.2452459.

Yang, Dingqi; Zhang, Daqing; and Qu, Bingqing. 2016. Participatory Cultural Mapping Based on Collective Behavior Data in Location-Based Social Networks. *ACM Transactions on Intelligent Systems and Technology*, 7(3):30:1–30:23. doi: 10.1145/2814575.

Yu, Chen; Liu, Yang; Yao, Dezhong; Yang, Laurence T; Jin, Hai; Chen, Hanhua; and Ding, Qiang. 2015a. Modeling User Activity Patterns for Next-Place Prediction. *IEEE Systems Journal*, PP(99):1–12.

Yu, Zhiwen; Wang, Hui; Guo, Bin; Gu, Tao; and Mei, Tao. 2015b. Supporting Serendipitous Social Interaction Using Human Mobility Prediction. *IEEE Transactions on Human-Machine Systems*, 45(6):811–818. doi: 10.1109/THMS.2015.2451515.

Yuan, Jing; Zheng, Yu; Zhang, Chengyang; Xie, Wenlei; Xie, Xing; Sun, Guangzhong; and Huang, Yan. 2010. T-Drive: Driving Directions Based on Taxi Trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 99–108, San Jose. doi: 10.1145/1869790.1869807.

Yuan, Jing; Zheng, Yu; Xie, Xing; and Sun, Guangzhong. 2011. Driving with Knowledge From the Physical World. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, pages 316–324, San Diego. doi: 10.1145/2020408.2020462.

Zhang, Daqing; Li, Nan; Zhou, Zhi-Hua; Chen, Chao; Sun, Lin; and Li, Shijian. 2011. iBAT: Detecting Anomalous Taxi Trajectories from GPS Traces. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, pages 99–108, Beijing. doi: 10.1007/978-3-642-30973-1.

Zhang, Tian; Ramakrishnan, Raghu; and Livny, Miron. 1997. BIRCH: A New Data Clustering Algorithm and Its Applications. *Data Mining and Knowledge Discovery*, 1(2):141–182. doi: 10.1023/A:1009783824328.

Zhang, Xiaoyu. 2013. Preference Modeling for Personalized Retrieval Based on Browsing History Analysis. *IEEJ Transactions on Electrical and Electronic Engineering*, 8(1):S81–S87. doi: 10.1002/tee.21922.

Zhao, Xinqiang; Li, Xin; Liao, Lejian; Song, Dandan; and Cheung, William K. 2015. Crafting a Time-Aware Point-of-Interest Recommendation via Pairwise Interaction Tensor Factorization. In *Knowledge Science, Engineering and Management*, volume 9403 of *LNCS*, pages 458–470. Springer. doi: 10.1007/978-3-319-25159-2.

Zheng, Kai; Zheng, Bolong; Xu, Jiajie; Liu, Guanfeng; Liu, An; and Li, Zhixu. 2016. Popularity-aware Spatial Keyword search on Activity Trajectories. *World Wide Web*, pages 1–25. doi: 10.1007/s11280-016-0414-0.

Zheng, Vincent W; Cao, Bin; Zheng, Yu; Xie, Xing; and Yang, Qiang. 2010a. Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach. In *Proceedings of the 24th Conference on Artificial Intelligence*, pages 236–241, Atlanta.

Zheng, Vincent W; Zheng, Yu; Xie, Xing; and Yang, Qiang. 2010b. Collaborative Location and Activity Recommendations with GPS History Data. In *Proceedings of the 19th International Conference on World Wide Web*, pages 1029–1038, Raleigh. doi: 10.1145/1772690.1772795.

Zheng, Yu. 2015. Trajectory Data Mining: An Overview. *ACM Transaction on Intelligent Systems and Technology*, 6(3):1:1–1:41. doi: 10.1145/2743025.

Zheng, Yu and Xie, Xing. 2010. Learning Travel Recommendations From User-generated GPS Traces. *ACM Transactions on Intelligent Systems and Technology*, 2(2):2:1–2:29. doi: 10.1145/1889681.1889683.

Zheng, Yu and Zhou, Xiaofang. 2011. *Computing with Spatial Trajectories*. Springer. ISBN 978-1-4614-1628-9.

Zheng, Yu; Li, Quannan; Chen, Yukun; Xie, Xing; and Ma, Wei-Ying. 2008a. Understanding Mobility Based on GPS Data. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, pages 312–321, Seoul. doi: 10.1145/1409635.1409677.

Zheng, Yu; Liu, Like; Wang, Longhao; and Xie, Xing. 2008b. Learning Transportation Mode from Raw GPS Data for Geographic Applications on the Web. In *Proceedings of the 17th International Conference on World Wide Web*, pages 247–256, Beijing. doi: 10.1145/1367497.1367532.

Zheng, Yu; Zhang, Lizhu; Xie, Xing; and Ma, Wei-Ying. 2009. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, pages 791–800, Madrid. doi: 10.1145/1526709.1526816.

Zheng, Yu; Xie, Xing; and Ma, Wei-Ying. 2010c. GeoLife: A Collaborative Social Networking Service Among User, Location and Trajectory. *IEEE Database Engineering Bulletin*, 33 (2):32–39.

Zhou, Changqing; Frankowski, Dan; Ludford, Pamela; Shekhar, Shashi; and Terveen, Loren. 2007. Discovering Personally Meaningful Places: An Interactive Clustering Approach. *ACM Transactions on Information Systems*, 25(3). doi: 10.1145/1247715.1247718.

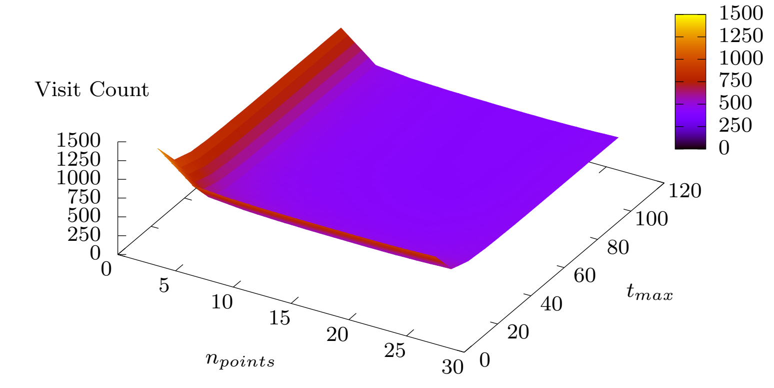
Zhou, Jun; Zhao, Qinpei; and Li, Hongyu. 2014. Integrating Time Stamps into Discovering the Places of Interest. In *Proceedings of the 10th International Conference on Intelligent Computing Methodologies*, pages 571–580, Taiyuan. doi: 10.1007/978-3-319-09339-0_58.

Zimmermann, Matthias and Bunke, Horst. 2002. Hidden Markov Model Length Optimization for Handwriting Recognition Systems. In *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition*, pages 369–374, Niagara on the Lake. doi: 10.1109/IWFHR.2002.1030938.

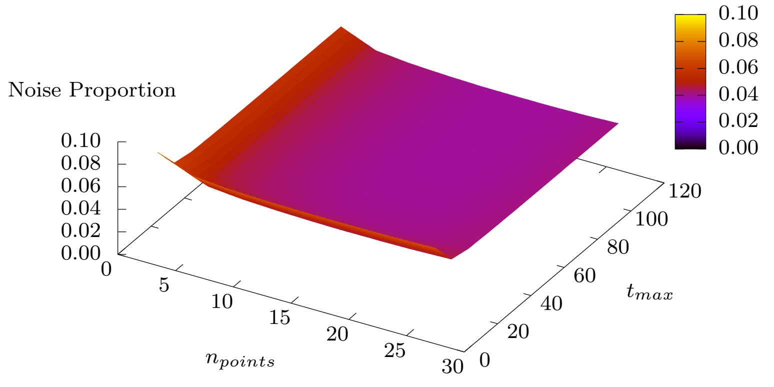
Appendices

Throughout this thesis, we have used two primary sources of geospatial trajectories for evaluating techniques: the Nokia Mobile Data Challenge (MDC) and Warwick datasets. In order to reduce repetition, however, several evaluation graphs have been omitted from the main chapters when trends from both sources of data are very similar. The previously omitted graphs are contained here. Specifically, Appendix A contains graphs relating to the Gradient-based Visit Extractor (GVE) algorithm presented in Chapter 4. The graphs in Appendix B relate to the Land Usage Identification (LUI) procedure from Chapter 5, and Appendix C contains graphs for the Predictive Context Tree (PCT), originally presented in Chapter 7. Finally, Appendix D contains graphs generated using the techniques presented in Chapters 5 and 7 but using data from the MDC data where periods of data with truncated latitude and longitude values have been retained, extending the discussion found in Section 7.4.

A Visit Extraction

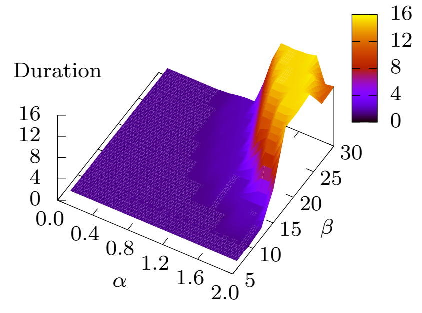


(a) Number of visits identified.

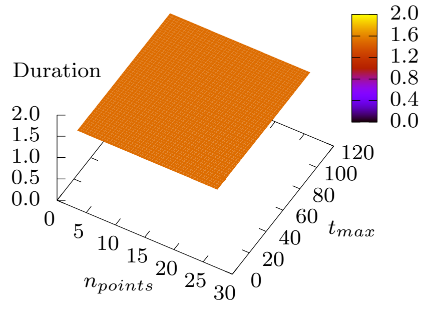


(b) Proportion of trajectory points designated as noise.

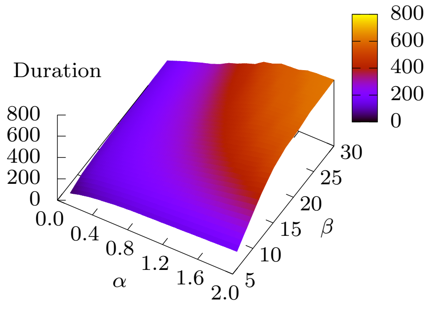
Figure A.1: The effect of the parameters n_{points} and t_{max} on the GVE algorithm when extracting visits over the Warwick dataset, with the remaining parameters fixed at $\alpha = 0.1,\ \beta = 30$. This figure mirrors the MDC graph in Figure 4.4.



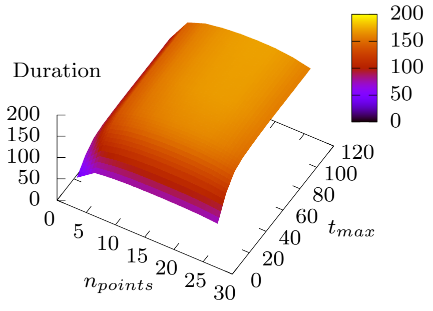
(a) Minimum visit duration (seconds).



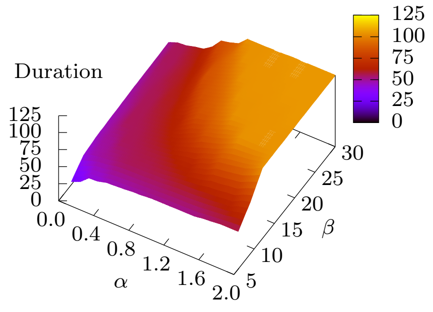
(b) Minimum visit duration (seconds).



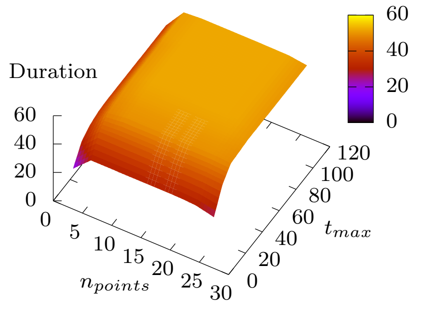
(c) Average visit duration (minutes).



(d) Average visit duration (minutes).

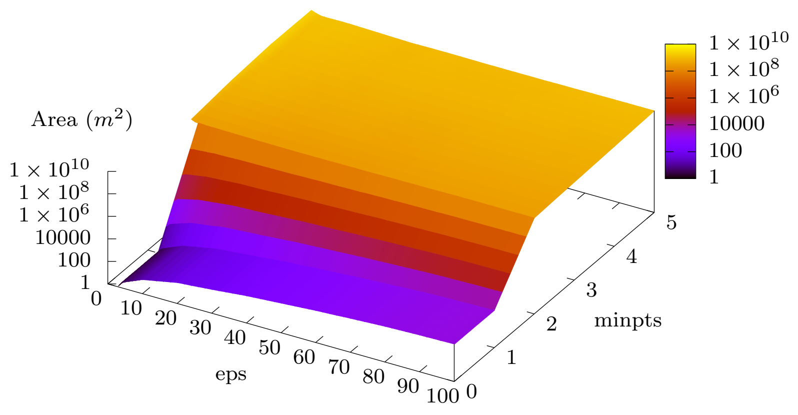


(e) Maximum visit duration (hours).

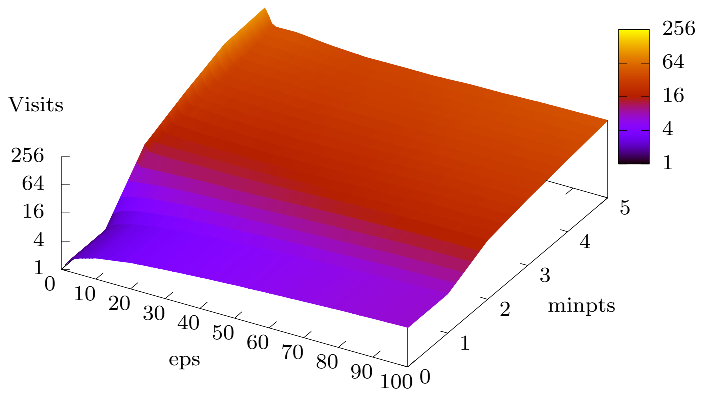


(f) Maximum visit duration (hours).

Figure A.2: The effect of parameters on the minimum, average, and maximum length of extracted visits for the GVE algorithm on the Warwick dataset, where constrained parameters were held at $\alpha = 0.1,\ \beta = 30,\ n_{points} = 5,\ t_{max} = 60$. This figure mirrors the MDC graph in Figure 4.5.



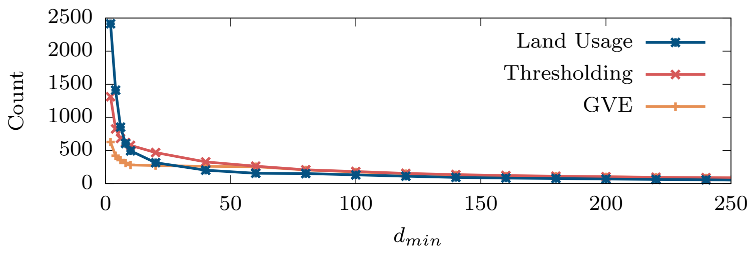
(a) Average location area.



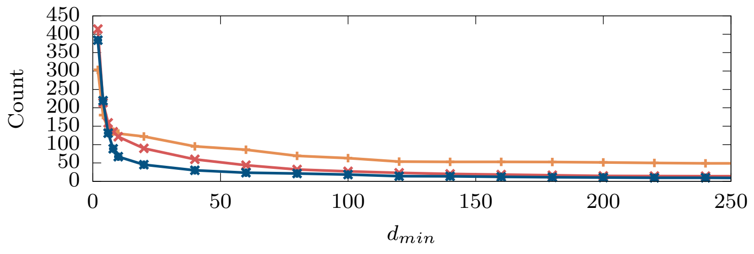
(b) Average number of visits per location.

Figure A.3: The effect of DBSCAN's *eps* and *minpts* parameters on the average size of locations and the average number of visits per location for the Warwick dataset on visits identified using GVE. This figure mirrors the MDC graph in Figure 4.7.

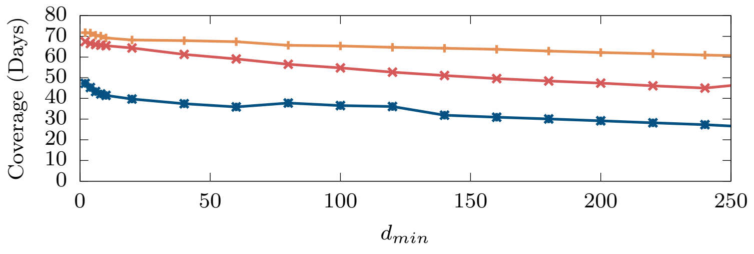
B Land Usage Augmentation



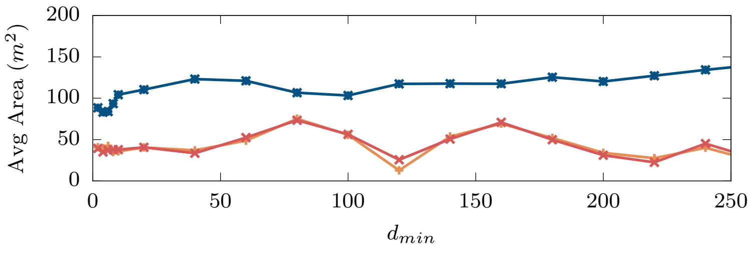
(a) Visit/Interaction Count.



(b) Element/Location Count.



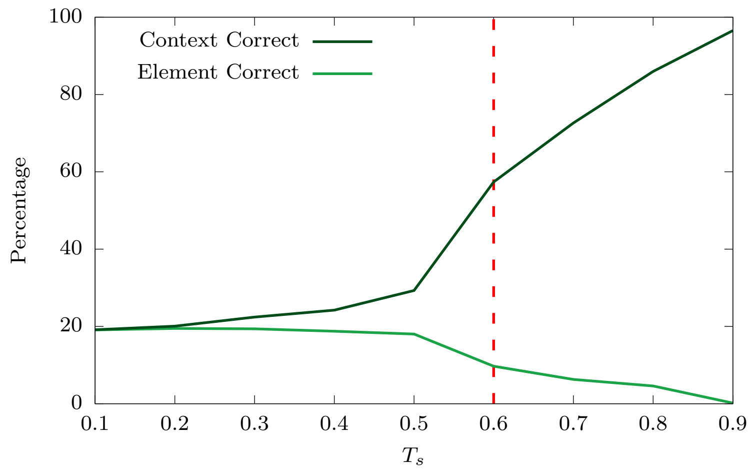
(c) Total Time.



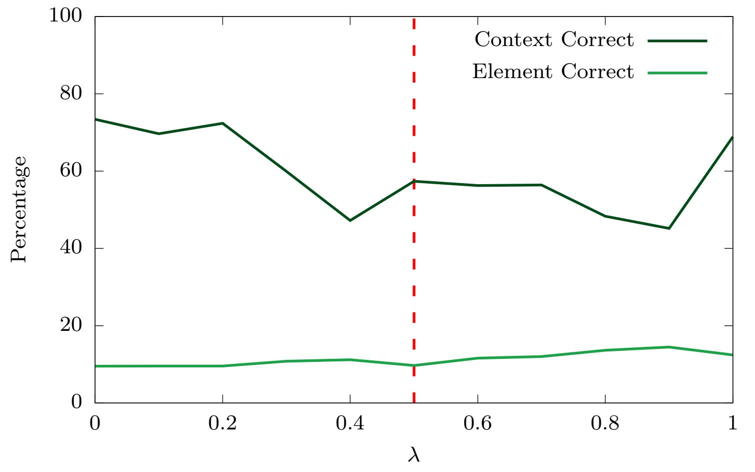
(d) Average Area.

Figure A.4: Summaries of comparisons between land usage elements identified through the LUI procedure and locations extracted through location extraction techniques over the MDC dataset. This figure mirrors the Warwick graphs in Figures 5.8-5.10.

C The Predictive Context Tree (PCT)

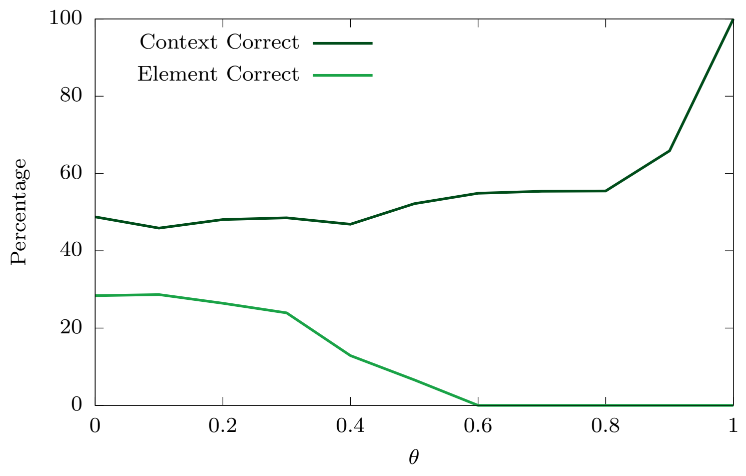


(a) Selection Threshold (T_s).

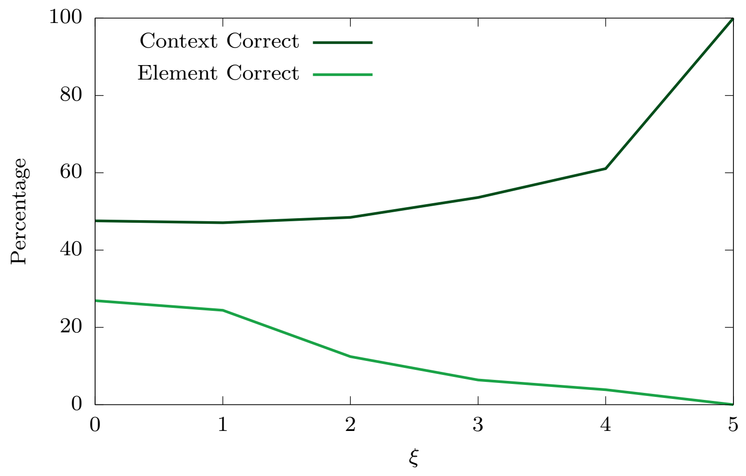


(b) Semantic Weighting (λ).

Figure A.5: The effect of parameters T_s and λ on context prediction, where d_{min} = 10min over the MDC dataset. This figure mirrors the Warwick graph in Figure 7.9.

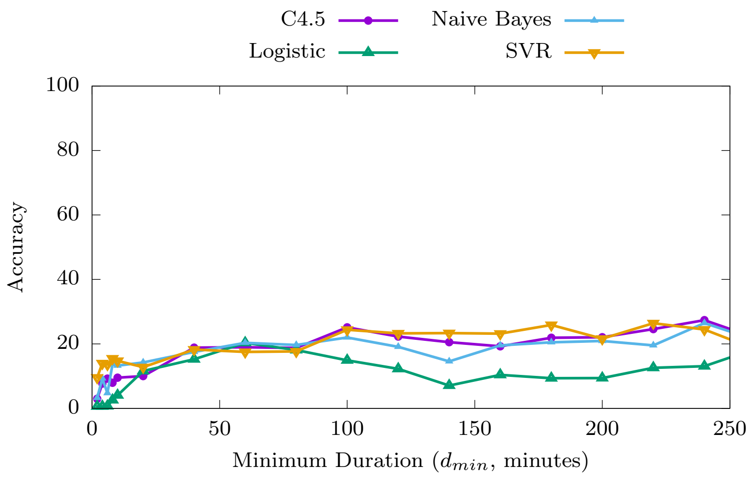


(a) Pruning Threshold (θ).

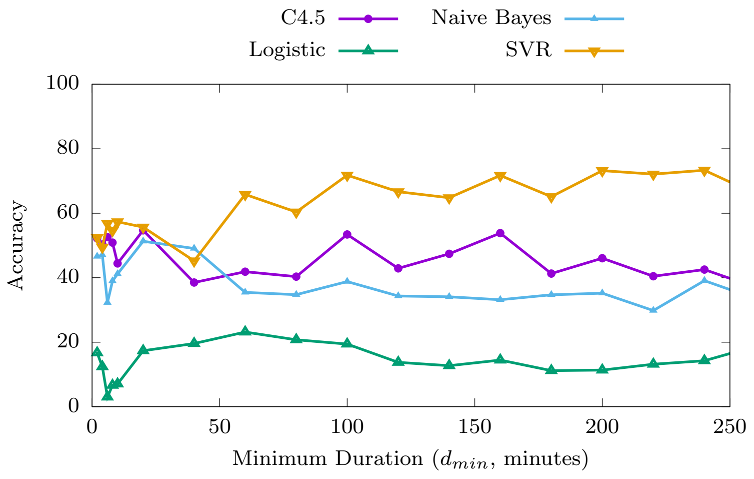


(b) Storage Overhead (ξ).

Figure A.6: Predictive accuracies observed when using pruned Context Trees generated from MDC data. This figure mirrors the Warwick graph in Figure 7.12.

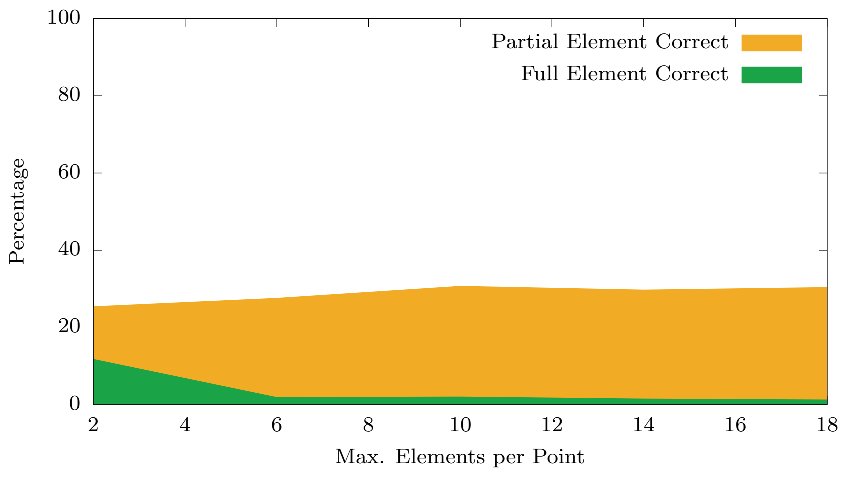


(a) Single element PCT, element correct.

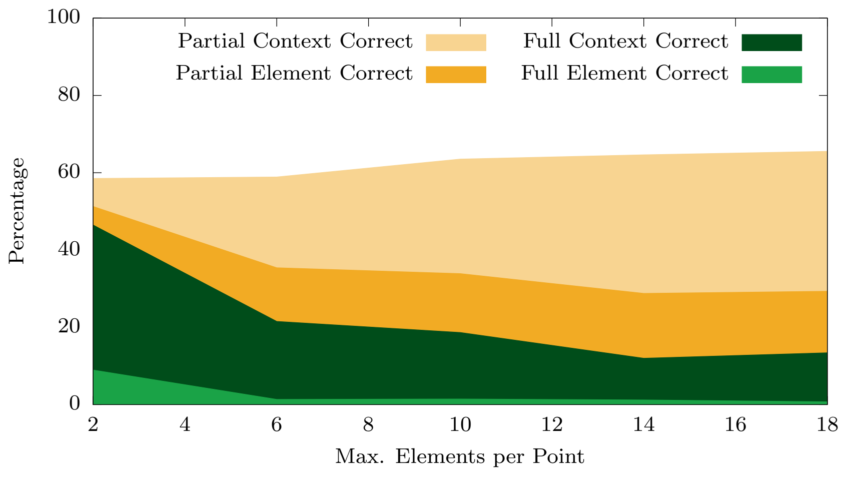


(b) Single context PCT, context correct.

Figure A.7: Predictive accuracies obtained when using different probabilistic models as classifiers in the PCT over the MDC data. This figure mirrors the Warwick graph in Figure 7.13.



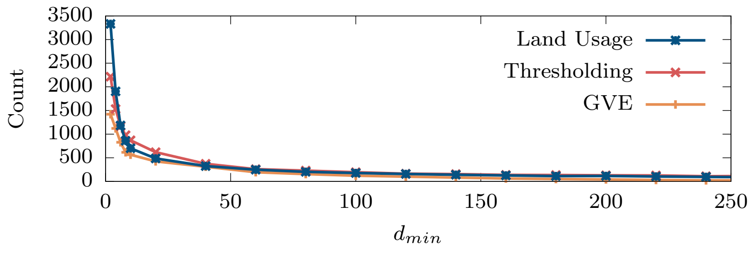
(a) Multi-element.



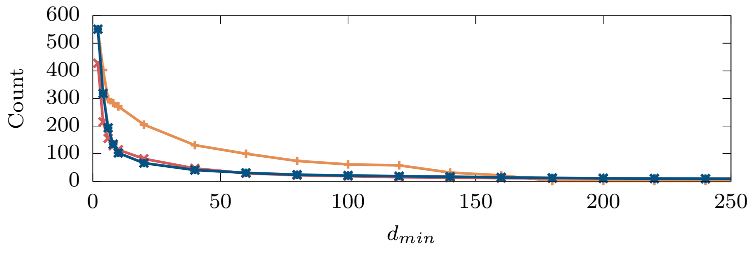
(b) Multi-context.

Figure A.8: The effect of n on predictive accuracy for the multi-element Context Tree over MDC data. This figure mirrors the Warwick graph in Figure 7.14.

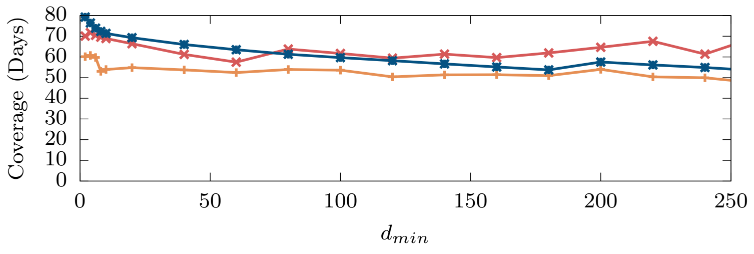
D MDC Data With Truncated Periods



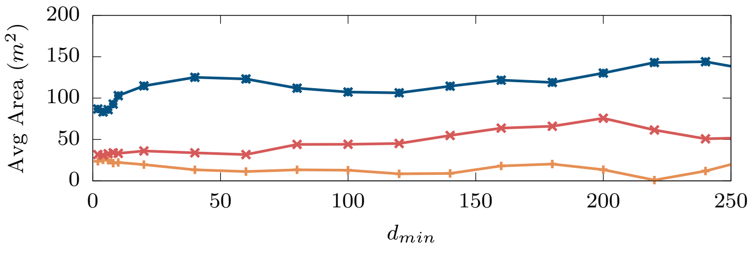
(a) Visit/Interaction Count.



(b) Element/Location Count.



(c) Total Time.



(d) Average Area.

Figure A.9: Summaries of comparisons between land usage elements identified through the LUI procedure and locations extracted through location extraction techniques over the MDC dataset where the truncated periods of data have not been removed.

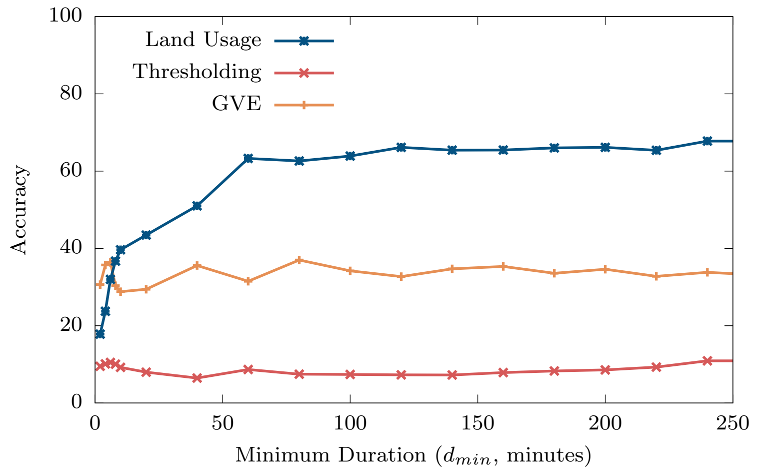


Figure A.10: Predictive accuracies for locations extracted with thresholding, and land usage elements identified through the LUI procedure, when predicting with SVMs over the MDC dataset with truncated periods.

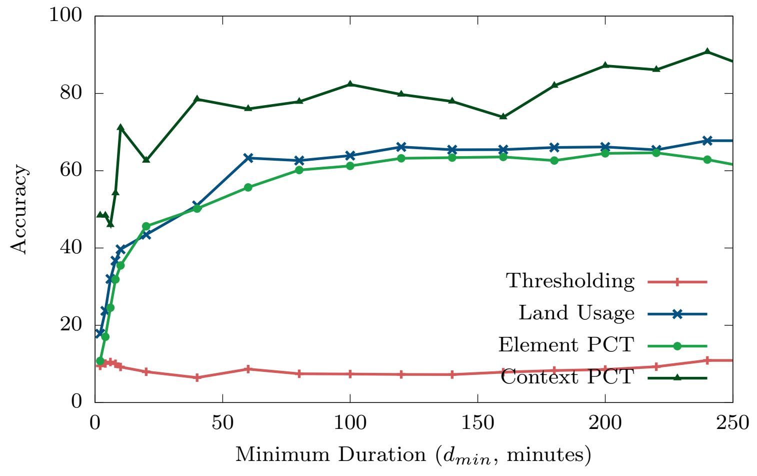


Figure A.11: Predictive accuracies for the different prediction techniques for the MDC dataset with truncated periods.