Mobility patterns for Context-aware ridesharing services

MINING BIG DATA TO IMPROVE SHARED MOBILITY SERVICES

Iván Mendoza Vázquez

Supervisor:

Prof. Chris M. J. Tampère

Co-supervisors:

Prof. Pieter Vansteenwegen

Prof. Pablo Vanegas

Members of the Examination Committee:

Prof. Joost Duflou

Prof. Tom Holvoet

Prof. Dana Borremans

Prof. Lieselot Vanhaverbeke

Luk Knapen, Ph.D.

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Science (PhD): Mechanical Engineering

ACKNOWLEDGEMENTS

Initially, I would like to express my sincere gratitude to my supervisor Prof. Chris Tampère for his continuous support during my Ph.D research, also for his motivation, and vast knowledge. His assistance was a key aspect during this last year and also when writing this thesis. My sincere thanks also go to Prof. Pieter Vansteenwegen, Prof. Pablo Vanegas and the members of the evaluation committee, since without their guidance it would not be possible to conduct this research.

Thanks to my colleges during the 5 years in the building of Mechanical Engineering: Marco, Paola, Farzad, Vaclav, Xin, Jeroen and Willem. I also want to express my gratitude to my Ecuadorian friends in Leuven: Andrés, Paola, Paul, Lorena, Jaime, Fernanda S., Edgardo, Mónica, Fernanda A and her husband Peter Willems.

This research project was financially supported by the National Secretariat of Higher Education, Science, Technology and Innovation of Ecuador (SENESCYT). At last, I would like to thank my wonderful wife and children, to whom I dedicate this document and who were my inspiration to achieve this stage after of my life almost six years of continuous effort.

ABSTRACT

Dynamic ridesharing is a mobility service where car drivers offer available seats in their car for sharing with passengers whose origin and destination happen to be close to their traveled path. It has gained interest as a potentially sustainable form of mobility, because it offers additional mobility simply by exploiting unused capacity that would otherwise be lost, and therefore with minimal or no marginal social cost – this in contrast to other alternatives for the private car like ridehailing services or public transport. Dynamic ridesharing however consist of a two-sided market, giving rise to a chicken-and-egg problem: without sufficient demand for trips, there is little incentive for a driver to offer his rides for sharing; without sufficient rides being supplied, there is little opportunity for passengers to choose the service.

In order to attain a critical mass to guarantee the operability of dynamic ridesharing, one option is to train context-aware software agents to recognize the driver's mobility patterns and automatically predict and offer upcoming rides for sharing. To achieve this, this thesis develops an architecture of such a ridesharing app and the workflow of mobility history data collection and processing functions it should perform. It then develops enhanced iterative methods to identify in travel histories the user's personal points of interest, typical arrival times or transitions between locations and other mobility patterns that could be exploited to anticipate a ride. Without constraining ourselves to specific prediction techniques, this thesis assumes that in general, the predictive performance of any learning method depends on the regularity and frequency of patterns in the travel history (among others), and it proposes and tests novel methods to extract these multi-day characteristics from empirical life-logging data.

A key asset to understand whether the set of all automatically predicted trips in a region of interest would form an attractive supply for candidate ridesharing travelers, is simulation in various scenarios of the trips made and shared by a synthetic population. However, in existing synthetic travel demand, multiday characteristics of the trips are lacking, and hence one cannot determine which subset of all trips would be predictable and shareable by an automated ridesharing agent.

To remedy this, the thesis proposes a method for generating synthetic multi-day trip demand for an urban region by merging through a newly developed form of statistical matching two complementary datasets: a set consisting of multi-day tracks from lifelogging data, and a second one with synthetic home tours for the same region network of Antwerp, Belgium. For successful matching, the intersection of both data sets should contain enough coincident characteristics; however, some of these matching features can only be discovered through appropriate data mining procedures. Through machine learning functions, describing the correlations that exist in the donor data between the matching features and the multiday characteristics, the missing multiday information was successfully transferred to the receptor database.

Combined, the methodologies described in this research provide a way, based on the analysis of big data collected by mobile devices, to study how dynamic ridesharing could act as a mature travel mode in transport planning, and herewith to obtain recommendations for shared-mobility systems.

ABSTRACT

Dynamic ridesharing is een mobiliteitsdienst waarbij automobilisten lege stoelen in hun auto aanbieden om te delen met passagiers van wie de herkomst en bestemming zich langsheen hun afgelegde pad bevinden. Deze vorm van mobiliteit is potentieel duurzaam, omdat het extra mobiliteit biedt louter door het benutten van ongebruikte capaciteit die anders verloren zou gaan, en dus tegen weinig tot geen marginale sociale kosten - dit in tegenstelling tot andere alternatieven voor de eigen auto, zoals deeltaxi's of openbaar vervoer. Dynamic ridesharing bestaat echter uit een tweezijdige markt, waardoor een kip-en-ei-probleem ontstaat: zonder voldoende vraag naar ritten is er weinig prikkel voor een chauffeur om zijn ritten te delen; zonder dat er voldoende ritten worden gedeeld, is er voor passagiers weinig neiging om voor deze dienst te kiezen.

Om een kritische massa te bereiken voor dynamic ridesharing, is het een optie om contextbewuste softwareagenten te trainen om de mobiliteitspatronen van de bestuurder te herkennen en automatisch aanstaande ritten te voorspellen en aan te bieden om te delen. Hiertoe ontwikkelt dit proefschrift een architectuur voor een dergelijke ridesharing app en de workflow die deze zou moeten uitvoeren bij het verzamelen en verwerken van de reishistorie van de gebruiker. De thesis ontwikkelt ook verbeterde iteratieve methoden om in de reishistorie van de gebruiker diens persoonlijke attractiepolen, typische aankomsttijden of overgangen tussen locaties en andere mobiliteitspatronen te identificeren die kunnen worden gebruikt om een aanstaande rit vooraf te herkennen. Zonder ons te beperken tot specifieke voorspellingstechnieken, gaat dit proefschrift ervan uit dat in het algemeen de voorspellende kracht van een leermethode afhangt van de regelmaat en frequentie van patronen in de reisgeschiedenis (onder andere), en stelt het nieuwe methoden voor om deze multi-dagkenmerken af te leiden uit empirische *life-logging*gegevens.

Een belangrijke troef om te begrijpen of het totaalaanbod van alle automatisch gedeelde ritten in een studiegebied een aantrekkelijk aanbod zou vormen voor kandidaat-ridesharing-gebruikers, is simulatie in verschillende scenario's van de ritten die gemaakt en gedeeld zouden worden door een synthetische populatie. Bij de bestaande synthetische verkeersvraag ontbreken echter de meerdaagse kenmerken van de ritten en daarom kan men niet bepalen welke subset van alle ritten voorspelbaar en deelbaar zou zijn door een automatische ridesharing-applicatie.

Om hieraan tegemoet te komen, stelt het proefschrift een methode voor voor het genereren van een synthetische vraag met meerdaagse ritkenmerken in een stedelijke regio. Via een nieuw ontwikkelde vorm van statistische matching voegt deze twee complementaire datasets samen: een set bestaande uit meerdaagse tracks van life-logging data, en een tweede met synthetische huisgebaseerde *tours* voor hetzelfde regionale netwerk van Antwerpen, België. Voor een succesvolle matching moet de doorsnede van beide datasets voldoende overeenkomstige variabelen bevatten; sommige van deze gemeenschappelijke variabelen kunnen echter alleen worden ontdekt via geschikte dataminingprocedures. Door machine learning van functies die de correlaties beschrijven die bestaan in de donorgegevens tussen de gemeenschappelijke variabelen en de meerdaagse kenmerken, werd de ontbrekende meerdaagse informatie met succes getransfereerd naar de receptordatabase. Gezamenlijk bieden de methodologieën die in dit onderzoek worden beschreven een manier om, op basis van de analyse van big data verzameld door mobiele apparaten, te bestuderen hoe dynamic ridsharing zou kunnen fungeren als een volwassen reismodus in transportplanning, en om hiermee aanbevelingen te doen omtrent gedeelde mobiliteitssystemen.

CONTENTS

ALGORITHMS

FIGURES

TABLES

ACRONYMS

Concept	Description
DBSCAN	Density-based spatial clustering of applications with noise
GPS	Global Positioning System
IPD	Iterative POI Detection
MSE	Mean Squared Error
OD	An Origin – destination combination
OPTICS	Ordering points to identify the clustering structure
POI	Point-of-interest
P-POI	Personal Point-of-interest

1 INTRODUCTION

As traffic congestion started rising some decades ago due to the increasing ownership of private vehicles and trips with low occupancy, for instance, high rates of commuting trips in cars with most seats being empty; ridesharing programs such as carpooling have been proposed by authorities as alternatives to reduce travel times, allowing users to share trips by splitting costs with other users having common destinations. Recent forms of shared mobility include several services that enable users to share transportation resources, including, but not limited to bikesharing, carsharing, ridesharing, have evolved due to development of mobile and communication technologies. A simple definition of ridesharing is given in (1), which states that ridesharing is a mode of transportation in which individual travelers, having similar itineraries and time schedules, share a vehicle for a trip and split travel expenses. Other than spatiotemporal constraints can be added to the matching models, sometimes detours are required to pick up or drop-off passengers, and finally costs can include gas, tolls, or parking fees. Perhaps the biggest difficulty to enable ridesharing as a service, refers to the rise of a chicken-and-egg problem due to a two-sided market (2): on the supply side, a sufficiently large mass of passengers is required to encourage drivers to participate; on the demand side, passengers request availability of the service to a destination of interest in certain time frame.

To tackle this problem, different strategies have been proposed in literature, including improvements to the matching algorithms(3,4), pricing schemes or evaluating the capability of ridesharing scenarios via simulation(5,6). Another approach, discussed in this research, can be to exhibit the potential aggregate supply to passengers for their trips, so that they can value the service and will be enthusiastic to use it. The omnipresence of mobile devices capable of exchanging information in real time, permits developing smart apps that can automatically discover this supply in mobility patterns of tracked users, providing smart assistance to passengers. For this, software agents must learn routines in patterns to provide estimates of the expected supply conditioned to specific passenger requirements.

The mobility patterns let multi-day characteristics to be extracted from travel logs, so that an enriched synthetic population can be produced to evaluate this approach via simulation. Then different scenarios can be tested which differ from current research efforts.

1.1 RESEARCH QUESTIONS

The main objective of this research is evaluating in empirical mobility data, if it is possible to automatically anticipate sharable trips in order to exhibit an aggregate supply for ridesharing.

To allow agents to predict mobility behavior, all possible types of context data must be mined to learn the patterns and the underlying relations, then the extracted multi-day data can be used to produce synthetic demand to simulate different scenarios. For this, multiple tasks must be completed which lead to the following questions:

- Is it possible to retrieve a consistent travel history from tracking data by inferring trips between locations from the spatio temporal information? Which are the best techniques?
- Is it possible to mine multiday mobility patterns from each user's travel history in order to build trip prediction models conditioned to context information? Which methods must be used?

- Is it possible to transfer these multiday characteristics to synthetic populations to represent an entire region's behavior in order to simulate different shared-mobility scenarios of interest? What are the possible ways to achieve this?
- Is it possible to create simulations that mimic context-aware software agents, predicting ride matches at short term to evaluate different ridesharing scenarios? What kind of analysis and visualizations can be produced?

The objective of this research is to answer these questions. They have been discussed in different papers that have been published (or at the time of writing have been submitted), and that are reproduced in integral form as chapters in this document. This manuscript is organized in the following way:

- An introduction to the research topic is given together with the research questions
- Full text of each publication is presented; they are introduced briefly in the remainder of this introduction.
- Conclusions about the entire PhD research are provided together with some future directions

1.2 LIST OF CHAPTERS

The upcoming chapters in this manuscript correspond to the following published and submitted papers, which are now briefly described in this section.

CHAPTER 1

Introduction to the research topic, stating the research questions.

CHAPTER 2

Title: *Anticipatory assistance for real time ridesharing in environments of pervasive computing*

This paper (7) sketches the entire workflow, providing a methodology to automatically detect a trip as a potential supply, since automated recognition and registration of shareable trips may contribute significantly to dynamic ride sharing deployment on a larger scale. For this, a tracking app was developed to detect each trip's endpoints. This app was tested by integrating different state-of-the-art algorithms for trip detection and concepts to anticipate the shareable trips. The departure and arrival times were compared to those detected by commercial apps such as MOVES, resulting in smaller errors.

CHAPTER 3

Title: *Discovering regularity in mobility patterns to identify predictable aggregate supply for ridesharing*

This paper (8) tackles the weaknesses found in previous approaches when mining data for mobility patterns. Furthermore, it describes a methodology to identify a predictable aggregate supply for ridesharing via routines discovered in users' travel histories. The methodology empirically quantifies measures like the regularity and frequency of these patterns on a dataset consisting of 967 users scattered across different geographical areas. The sample exhibited high heterogeneity with respect to these measures (hence, of predictability, regardless of the prediction method). This paper shows how frequency of trip patterns decreases, while conditional regularity increases, when additional dimensions such as departure times are added

to the analysis. It was concluded that the traveler flexibility when accepting fewer regular trips is crucial to discover a larger supply, even when trips become less predictable.

CHAPTER 4

Title: *An iterative method for extraction of trip endpoints from life-logging data*

This paper tackles the difficulties found in the data aggregation stage when retrieving the trip chains consisting of personal points-of-interest. It proposes a novel algorithm that combines spatial information with spatiotemporal patterns found in earlier chain links of the travel history, improving the detection of the upcoming locations by using the knowledge that is constantly learned. Tests were performed on a real-life dataset collected during a campaign in Ecuador, where students used an app that captured the user's position at regular intervals. At last, results were compared to a well-known density-based method. These unique locations which appear several times in travel histories are called in this document a personal point of interest (POI). Their detection can be greatly improved by adding more attributes in the clustering procedure besides the typical spatiotemporal components, as soon as some mobility patterns with recurrent behavior are observed in a history, such as frequent transitions between POIs.

CHAPTER 5

Title: *Producing multi-day synthetic populations for shared-mobility simulations from lifelogging datasets*

Given that producing data with multiday characteristics is a complex task; the last paper tackles the generation of this demand, providing a rather direct procedure to generate enriched synthetic populations for more realistic assessments. This can be achieved by combining typical single-day datasets, with travel behavior patterns extracted from lifelogging data collected by existing mobile apps. This augmented dataset suitable for ridesharing simulations, allows matching trips to be constrained by multiday characteristics. From this, novel complex scenarios not included in most previous works can be evaluated.

2 ANTICIPATORY AUTOMATED ASSISTANCE FOR REAL TIME RIDESHARING IN ENVIRONMENTS OF PERVASIVE COMPUTING

Mendoza, I., Tampère, C., & Mekerlé, P. (2016). Anticipatory Assistance for Real Time Ridesharing in Environments of Pervasive Computing. 95th Annual Meeting of the Transportation Research Board. 95th Annual Meeting of the Transportation Research Board, Washington, D.C.

2.1 ABSTRACT

Dynamic ridesharing is a service for the transportation of passengers inside cities with intense traffic, which at the same time provides environmental, social and economic benefits. Although usually there are plenty of cars on the road, one of the challenges though is to collect sufficient information on supply: who is going where and is willing to pick up passengers? Since it is difficult to motivate people to manually register every trip as a potential supply, automated recognition and registration of shareable trips may contribute significantly to dynamic ride sharing deployment on a larger scale.

Sharing rides in real time requires a fast agreement between drivers and passengers at very short notice. The reduction of human intervention via the automatic entry of the inputs required for a new ride request, under some assumptions, allows the extension of the generally small-time window available to attain the arrangement. This automation can be achieved by mining a user's history to detect stay points in order to learn the mobility and behavioral patterns, allowing autonomous agents to predict upcoming ride requests and then propose shareable trips.

This paper contributes by defining the architecture of such an agent, then testing a smartphone application that tracked a user's activity for an extended period. An approach for trip segmentation via detection of stay points found in the collected data was evaluated, then compared to existing tracking apps. The results show that the design of context-aware ridesharing apps using this approach is possible, at last different directions for future work are proposed.

Keywords: real time ridesharing, mobility patterns, data mining, context-awareness.

Credit author statement

Iván Mendoza: Conceptualization, methodology, software, writing original draft preparation, and visualization / data presentation.

Chris Tampere: Supervision, validation, Writing- Reviewing and Editing.

C. Mekerlé: Data collection, validation.

2.2 INTRODUCTION

The arrival of context-aware mobile applications in recent years has allowed the evolution of on-line services in different areas (9), improving the way information is provided based on current location, time, user profile information and other data obtained from sensors found today in modern mobile devices. More recent applications in the Transportation domain deliver location-based services as in (10) to smartphone users in order to simplify the search process for real-time traffic status and transport-related information when it is integrated with web services that retrieve data from regularly updated databases situated in remote servers.

Dynamic ride-sharing (DRS) is a particular type of cooperative travel service, where riders (passengers) let know their intentions to travel at very short notice, and then an agreement considering costs, pick-up and delivery details, has to be achieved among the participants before it is possible to serve the passenger. At the present time, ride-sharing has been identified as an important alternative with social and environmental benefits for the transportation of passengers (11). Some recent studied challenges about the topic can be found in (12). One notorious difficulty in peer-to-peer services like DRS is the chicken-and-egg problem: using the service is only interesting if there are sufficient suppliers offering rides; supplying rides is interesting if there is a reasonable chance of someone using it. This problem should be solved at the supply side of the rides: without supply, no passengers can choose to share rides; but without passengers, drivers may still offer the ride (albeit in vein) and hence create a basis for a community of DRS users to grow steadily. Yet, if any manual action or cost is involved in supplying rides, drivers will soon lose motivation to do so, especially when there are initially only few users. This calls for the action of offering rides to be supported by an automated system running in the background, so that it is transparent for the driver once installed.

A second motivation for automated assistance of DRS supply is time. The interval between the time a request is presented and the latest possible departure time a driver can be on route to the pick-up location of the passenger is known as the ride-matching window (1). Since finding the optimal match is not a trivial task and the available time is usually short, a prompt support by the application is critical. The contribution of this paper is to take full advantage of the relevant information contained in the context perceived by mobile device sensors, to allow agents make predictions about future trips as explained in (13) and then provide anticipatory-assistance to a passenger before a new trip is expected to start. By this, they improve the user experience through reducing human intervention.

This paper is organized as follows: section 1 introduces the problem and limits the scope of this research. In section 2 recent literature to help define the methods here mentioned is reviewed. Section 3 discusses the methodology and the implementation aspects to consider for a proposed context-aware architecture for anticipatory assistance in ride-sharing applications. The evaluation of the model and the results of a set of experiments on different scenarios are shown in section 4. Finally, section 5 presents the conclusions and defines some future work.

2.3 LITERATURE REVIEW

In this paper the identified mobility patterns (and the potential regular trips that can be shared) are intended to be used by agents in order to assist riders in the matching process by providing the driver candidates, that is, the input for the ride-matching optimization process. Literature related to mining raw data to identify travel patterns is summarized.

In (14) “wearable” computers (like a GPS receiver) can be used as intelligent agents to assist users in a variety of tasks. By using location as the agent’s context, they cluster the GPS data to

find transitions between meaningful places and then to use this information as input for a Markov model in order to predict the next user's location. In (15) the authors extent their work, by adding new contextual data to infer meaningful places by using a variation of k-means clustering. Reference (16) provides an architecture for the prediction of a user's mobility, combining contextual information from many sources (i.e. user's preferences) when the history of trip patterns is not available.

In (17), the same classical clustering algorithm is applied to a very large data set of sensor observations over an extended period to find historical traffic. An alternative to k-means for the same task is presented in (18), where the authors list some of the drawbacks of this method and introduce a density-based approach called DJ-Clustering algorithm which is a modification of the DBSCAN algorithm originally introduced by (19). Other alternative introduced in (20) presents a non-parametrized Bayesian alternative to k-means for the automatic extraction of places from discontinuous GPS measurements.

Some literature refers to these places as points-of-interest, this term is also widely used in navigation systems to refer important public places and will be used along this paper to avoid ambiguity, however in this context they are treated as personal and private places only relevant for a specific user.

More recently in (21) the authors describe a detailed method to generate a location history by mining for "stay points" and the travel sequences among them. The output is organized in a tree-based hierarchical graph. This is a complement for their previous work in (22) when they described the procedure for learning transportation modes from GPS raw data. An important contribution to automatic ride-sharing is presented in (23), in this work the authors mine travel records of several moving objects for trips with similar trajectories that were mapped to a road network to find shareable frequent routes.

A very recent survey in mobility patterns found in GPS datasets is presented in (24), where the authors are supported by new studies to sustain the assumption that human mobility behaves in a highly regular way. They review some popular methods for future move prediction, transport mode detection, trajectory patterns and location-based activities recognition and they provide some directions for further research. Besides a GPS, an extensive list of sensors can be used to gather more enriched contextual information as explained in (25) to make predictions that are not only location-based. A way of automation for dynamic ride-sharing by predicting future commutes, based on a range of data sources obtained from smartphones has been already treated in (26), here the authors use decision tree learning to extract logical rules from the dataset.

Our approach differs from previous research in the sense that context just captured is used to know at short notice whether a ride share opportunity can be exploited, allowing the agent to make real time decisions only when there is a high level of confidence about a user's future activity. Another contribution is the definition of a software architecture where dedicated modules run processes directly inside the smartphone, avoiding depending on persistent communication with a remote server and increasing the security level since user's history remains safely stored in the device. Finally, the storage of points-of-interest for a trip origin and destination instead of the whole trajectory information reduces the amount of data to be processed, reducing computation effort on the client side.

2.4 METHODOLOGY

The information an agent needs for providing a ride-matching candidate according to the literature is mainly:

From Passengers**,** the prediction of an upcoming trip, containing:

- Current and next locations (a trip origin and destination)
- Time estimates for arrivals/departures
- Passenger specific constraints

From Drivers, the list of recurrent shareable trips, containing:

- Current and next locations (a shareable trip origin and destination)
- Occupancy/capacity of private vehicle
- Time estimates for arrivals/departures
- Tolerable detour distance

Assuming the vehicle occupancy could be measured, the rest of information can be obtained by a procedure illustrated as an architecture (Figure 1) joining different modules. The most relevant components are described in this document.

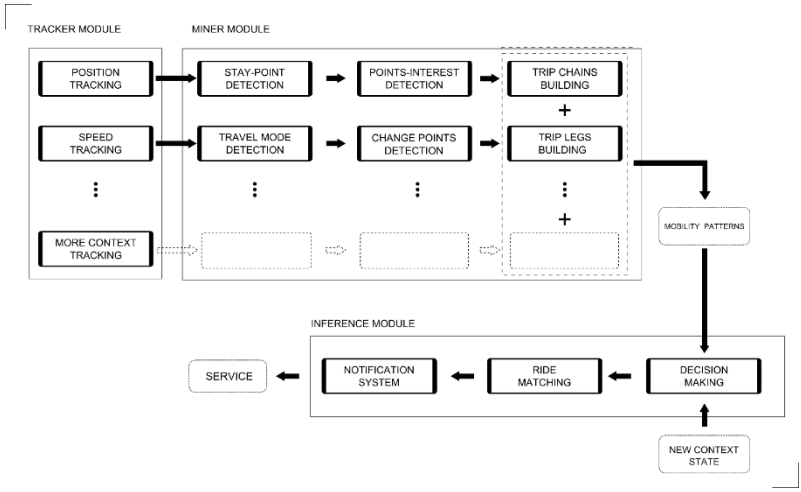

Figure 1. Architecture for Automatic Ridesharing

2.4.1 Tracking Module

This module is responsible of collecting contextual information from different sources. It is intended to be an interface between the physical (or virtual) sensors and the rest of the software modules, converting raw data into a more understandable format. As stated in the literature provided in the previous section, most of the traditional techniques took only advantage of spatial coordinates. However as explained in (25) and (27), the availability of other sources can increase the accuracy of the estimations of a user's position, by combining elements like Wi-Fi based positioning , mobile phone positioning, etc. In some cases when GPS is not available like

in indoor environments other sensors (accelerometers, gyroscopes, etc.) can help tracking a vehicle position as illustrated in (28).

The time interval between two consecutive GPS points is not always constant, mainly due to signal loss or because another source makes a point available earlier. In (29) the authors demonstrate the importance of having a "long enough gap" between GPS measures in order to preserve battery power, and based on some experimentation they suggest that an interval of 5 seconds for the majority of tracking applications can still provide reliable data. The time representations used in our prototype are UNIX timestamps that represent times in milliseconds, making them feasible for being used in formulas.

2.4.2 Mining Module

This module involves the detection of those locations that are relevant for the user and the transitions among these locations. As mentioned before, in order to make predictions about a user's next moves, the raw data collected by sensors have to be processed in order to identify some meaningful points in order to reduce the amount of data for further analysis.

2.4.2.1 Points-of-interest detection

A subset of consecutive GPS points that have been collected can be joined into traces on a map. It can be noticed that those segments where the points are denser indicate positions where the user has spent more time wandering around a same geographic region than other segments where points clearly suggest a trajectory (Figure 2(a)), as stated in (24). Reference (21) call these regions stay points which is the term that is going to be used along this paper.

Definition 1 A stay point is the representation of the geographical region formed by a subset of consecutive GPS point**s** found inside a GPS log $G = \{p_1, p_2, .., p_m, p_{m+1}, .., p_o, .., p_{n-1}, p_n\}$, starting at p_m up to p_o where the distance $d(p_m, p_i)$ for $m < i \leq o$ is not greater than a parameter D_{th}, and the time spent inside the region $p_o.t - p_m.t$ is at least T_{th}.

Which means that p_m and every p_i must be directly density-reachable, where p_m is called the arrival point and p_o the departure point, $d(p_1, p_2)$ is the great-circle distance between two points p_1 and p_2, D_{th} is a distance threshold that specifies the search radius which depends on the expected application (scale of interest), and T_{th} is the minimum time the user has to stay inside this region in order to be detected.

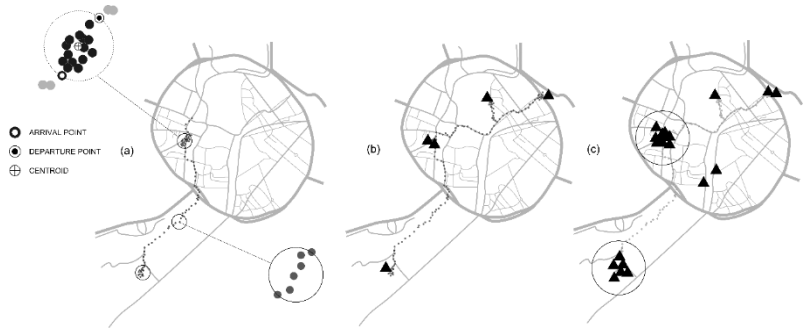

Figure 2. (a) A stay point defined by arrival and departure points and a centroid. (b) Collection stay points at the end of the day. (c) Stay points forming clusters at the end of larger time intervals.

Definition 2 The prototype of stay point s is a virtual point that represents the set of consecutive points $p_m, p_{m+1}, .., p_o$ for the stay point that starts at point m. This prototype has the following data structure and notation:

$s.lat = \frac{1}{\lvert P \rvert}\sum_{i=m}^{o} p_i.lat$	Average latitude
$s.lon = \frac{1}{\lvert P \rvert}\sum_{i=m}^{o} p_i.lon$	Average longitude
$s.arr = p_m.t$	Arrival time to the region
$s.dep = p_o.t$	Departure time

From this point on, the prototype will be used to refer to the corresponding stay point. The implementation of a mechanism to retrieve stay points prototypes is presented in Algorithm 1, where chronological scanning is not guaranteed but is not required as long as the neighborhood of an unobserved point can be found. Figure 2(b) shows an example of a one-day journey with the regular "extended" stops a user does during the day, which also let know the transitions between these stay points. Because a stay point is a single instance of an extended stop, a single location can have as many stay points as visits it has received. During larger intervals, stay points form relevant clusters on locations with high attraction levels for a single user, see Figure 2(c). Their size help obtaining a location's rank.

Algorithm 1. Heuristic for Stay point detection.

Let S be an empty set
For each unobserved point $p_m \in G$
    Let $N(m)$ be the set of consecutive points p_i (with $p_i.t > p_m.t$) where $d(p_m, p_i) \leq D_{th}$
    Let p_o be the last point p_i in $N(m)$
    If $(p_o.t - p_m.t) \geq T_{th}$ then
        Create a new stay point prototype for the points in $N(m)$ and add it to S.
Mark all visited points as observed.
Return S

Definition 3 A Point-of-interest (POI) is a frequently visited location by a single user, which owns a semantic meaning and can be obtained by joining all the stay points that are density-reachable at a distance not higher than $maxDist$ into a cluster C of size at least $minVisits$.

The data structure for a point-of-interest POI_k represented by a cluster C_k containing the corresponding stay points, has the following attributes.

$POI_k.lat = \frac{1}{\lvert C_k \rvert}\sum_{s_i \in C_k} s_i.lat$	Average latitude
$POI_k.lon = \frac{1}{\lvert C_k \rvert}\sum_{s_i \in C_k} s_i.lon$	Average longitude

In (30) and (31) the authors suggest different approaches to build "personal maps" by looking for transitions between these significant places from GPS data. These transitions allow the creation of association rules (32) that allow predictions of the future movements of a user.

2.4.2.2 Temporal Patterns

The clusters of stay points formed on each personal point-of-interest (POI), can be sub-clustered by the time dimension. This allows to find patterns in arrival and departure times. In other words, it is possible to know the time of the day a user regularly arrives and leaves a point-of-

interest. Let C_k represent the cluster of stay points on a point-of-interest k and $d_t(s_i, s_j)$ be the distance in time between two stay points s_i and s_j. For instance, for arrival times:

$$d_A(s_i, s_j) = |s_i.arr - s_j.arr| \qquad s_i, s_j \in C_k$$

Algorithm 2. Cluster stay points into points of interest.

Let $T = \{s_1, s_2, .., s_n\}$ be a collection of recognized stay points

Let C be the set of clusters $\{C_i\}$ which are the output of the algorithm.

For each unobserved random stay point s_i

  Get neighborhood $N_E(s_i) = \{ s_j \in T \mid d(s_i, s_j) \leq maxDist \}, s_j \notin C_i$

  Create a cluster C_i and add all $s_j \in N_E(s_i)$ to it.

  For each $s_j \in N_E(s_i)$

    Get neighborhood $N_E(s_j) = \{s_k \in T \mid d(s_j, s_k) \leq maxDist \}, s_k \notin C_i$

    If $count\left(N_E(s_j)\right) > 0$:

      Add all $s_k \in N_E(s_j)$ to C_i

      // Recursively call until no farther points can be reached

    Mark all visited points as observed

  If $|C_i| \geq MinPoints$:

    Add C_i to final clustering solution,

  Otherwise

    Label s_i as noise.

Return C

Therefore, a similar approach to that used when clustering the stay points that were close in space can be used. That is, for each stay point $s_i \in C_k$ inside cluster C_k the algorithm searches for the neighborhood,

$$N_A(s_i) = \{s_j \in C_k \mid d_A(s_i, s_j) \leq maxTime\}, \qquad s_i \neq s_j$$

Where $maxTime$ is a parameter that expresses proximity between two time measures, joining the stay points in C_k that are density-reachable by the arrival time attribute. Now for the departure times:

$$d_D(s_i, s_j) = |s_i.dep - s_j.dep| \qquad s_i, s_j \in C_k$$

$$N_D(s_i) = \{s_j \in C_k \mid d_D(s_i, s_j) \leq maxTime\}, s_i \neq s_j$$

One more time, joining only those stay points in C_k that are density-reachable by the departure time attribute. This procedure forms smaller clusters inside C_k, keeping only those which cardinalities are at least $minFreq$. Two different clustering arrangements for a single POI are then obtained: one for the arrival times and another one for the departure times namely $C_k^{(arr)}$ or $C_k^{(dep)}$.

Finally, the set of frequent (average) arrival times to POI_k is:

$$POI_k.arr^{(freq)} = \{f(C_1), f(C_2), .., f(C_n)\}, \qquad C_i \in C_k^{(arr)}$$

$$f(C_i) = \frac{\sum_{s_i \in C_i} s_i.arr}{|C_i|} \tag{1}$$

Equivalently, for the departure times from POI_k:

$$POI_k.dep^{(freq)} = \{g(C_1), g(C_2), .., g(C_n)\}, \qquad C_i \in C_k^{(dep)}$$

$$g(C_i) = \frac{\sum_{s_i \in C_i} s_i.dep}{|C_i|} \tag{2}$$

So by sub-clustering with respect to time-of-the-day, it is also possible to anticipate the departure time to a particular POI, when the user's current location is known (which happens to be the origin for the next trip).

2.4.2.3 Next Location Prediction

A set of different mobility patterns that represent the different transitions between points-of-interest together with the frequent arrival and departure times observed in the complete location history can be created. When a new stay point is identified by the smartphone, the agent will try to match this one with one of the recognized points-of-interest based on some similarity criterion.

Definition 4 Let $\Pi = \{POI_i\}$ be the set of all recognized points-of-interest for the same user, where i is an internal identifier. A reference point-of-interest for a new stay point s is the nearest neighbour in Π within a radius of at most $maxRadius$.

$$POI^{ref}(s) = \underset{po_i \in N(s)}{\operatorname{argmin}}\, d(po_i, s)\ , N(s) = \{po_j \in \Pi \mid d(po_j, s) \leq maxRadius\} \tag{3}$$

It can be assumed that if a reference point-of-interest cannot be found for a recently detected stay point, then the user has arrived to a brand new location, being a sufficient reason to conclude that the current user's behavior will not fit any known mobility pattern so far.

Taking a look to the location history, only for those transitions between frequent stay points, (those that have already been matched to a point-of-interest) the transition chains among points-of-interest visited at a pre-defined time interval (E.g. daily tours) are obtained. Each of these unique transition chains of length at least 2, will be called in this paper a mobility pattern, and Ω is the collection of all of them in the location history. With this information, a procedure to predict the next point-of-interest given the current one (or a previous sequence of two or more POIs) is illustrated below.

Before proceeding, only a subset of Ω' that includes those patterns that are similar to the current chain formed by the last m POIs is assumed to be used. This way, the search is reduced to only

those days where the user exhibited a mobility behavior similar to that during the current day (i.e. typically a same mobility pattern can be found among the same days of the week).

Now let $F_{\Omega'}(POI_i)$ give the number of times POI_i appears in Ω' followed by another point-of-interest, also $F_i(POI_k)$ denote the number of times POI_i directly precedes another point-of-interest POI_k in Ω'. Let $POI_{(t)}$ be the current known point-of-interest for a stay point identified at time step t, then the conditional probability that POI "j" $\mathrm{POI_j}$follows "i" in the chain becomes:

$$p_{ij} = Prob\left(POI_{(t+1)} = POI_j \middle| POI_{(t)} = POI_i\right) = \frac{F_i(POI_j)}{F_{\Omega'}(POI_i)} \quad (4)$$

This way, the next point-of-interest at time step $t+1$ only depends on the current one at time t, becoming a Markov chain, with a state-space equal to the list of points-of-interest in Ω'. Because it can be assumed that trips headed to the same POI are not possible (or useful), then p_{ii}=0. Finally, the selected point-of-interest to follow the current one, when $POI_{(t)} = POI_i$, is:

$$POI_{(t+1)} = \underset{POI_j \in \Omega}{\operatorname{argmax}} p_{ij} \qquad i \neq j \quad (5)$$

Also, let $nextArr(t) = \left\{t_k \in POI_k.arr^{(freq)} \;\middle|\; t_k > t\right\}$, be the list of recurrent arrival times to POI_k after time t, only for transitions from $POI_{(t)}$ in Ω'. Then the time a user will arrive to $POI_{(t+1)}$ is the closest time in the future.

$$POI_{(t+1)}.arr = \min_{t_k \in nextArr(t)} t_k \quad (6)$$

The set of association rules with an implicit probability can be built and updated every time a new mobility pattern is added, this way the agent will refer to these rules to make decisions about the further actions to take.

2.4.2.4 Finding shareable trips

The next location prediction allows to infer a coming request by a potential passenger so that the demand of the whole set of riders can be obtained for a close time in the future. On the supply side (drivers), future shareable trips are also inferred from their mobility patterns with the only difference that users with the role of driver make trips by using a private vehicle (besides assuming other constraints to be explained later). It must be taken into consideration that opposed to most of the current popular ride-sharing apps, neither drivers are part of a transportation company nor did they specify to serve passengers. Also, mapping trajectories to a road network is essential to reduce complexity.

A user (and the attached agent) can change the role based on the recognition of the travel mode. Some work on this topic can be found in (22) or (33). The following notation will be used to define the ride sharing problem.

- $det_{(max)}$, Maximum detour distance (km). γ, Vehicle occupancy (busy seats inc. driver),
- Q, Vehicle capacity. $Po^{(r)}$,$Pd^{(r)}$ Passenger's origin and destination,
- $Po^{(d)}$,$Pd^{(d)}$ Driver's origin and destination. $Pp^{(r)}$, Passenger's pick-up location,
- $Pq^{(r)}$, Passenger's drop-off location. $Pi.a_{(min)}$, Earliest possible arrival time to Pi,
- $Pi.a_{(max)}$, Latest possible arrival time to Pi. $Pi.d_{(min)}$, Earliest possible departure time to Pi, $Pi.d_{(max)}$, Latest possible departure time to Pi.

It is also appropriate to define the total detour from the original path a driver d follows to travel from an origin POI $Po^{(d)}$ to a destination POI $Pd^{(d)}$ in order to pick-up and drop-off a passenger r at different locations as:

$$h_V = d_V\left(Po^{(d)}, Pp^{(r)}\right) + d_V\left(Pp^{(r)}, Pq^{(r)}\right) + d_V\left(Pq^{(r)}, Pd^{(d)}\right) \quad (7)$$

$$det_{(tot)}(d,r) = h_V - d_V\left(Po^{(d)}, Pd^{(d)}\right) \quad (8)$$

Here, $d_V\left({P_1}^{(r)}, {P_2}^{(r)}\right)$ represents the total travelled distance by car from points-of-interest ${P_1}^{(r)}$ to ${P_2}^{(r)}$ by using the shortest possible (most efficient) path inside a network N. Also $Pp^{(r)}$ and $Pq^{(r)}$ are the corresponding pick-up and drop-off locations. In other words, the total detour is the amount of extra distance the driver's vehicle has to travel to serve a passenger compared to the original trip. Next the total-walk-distance a rider still must travel if being served is computed (i.e. the overall residual distance a passenger has to walk in the trip), defined as the distance using the shortest possible path by foot in N from the current location (origin) to the pick-up location plus the distance from the drop-off location to the actual final destination.

$$w_{(tot)}(d,r) = d_W\left(Po^{(r)}, Pp^{(r)}\right) + d_W\left(Pq^{(r)}, Pd^{(r)}\right) \quad (9)$$

This means that in some situations, a rider requires an extra effort to be served by a potential driver. This distance is different than that by car since links can be used in both directions and is normally symmetric. A maximum walk distance $w_{(max)}$ can then be another constraint the passenger can specify when requesting a new ride. Reference (1), defines some ride-sharing patterns for single and multiple passengers with and without detours.

Definition 5 A shareable trip is a supply-side trip by a driver d with a private vehicle from points-of-interest $Po^{(d)}$ to $Pd^{(d)}$, where $\gamma < Q$ and assuming the desire of the driver to participate. This definition can be extended to a shareable trip that can be used to fulfil a specific request by a rider r at time t. The feasible set of candidate drivers must follow these constraints for a ride-sharing pattern with no walk segments, that is $w_{(tot)}(d,r) = 0$ and $Po^{(r)} = Pp^{(r)}$, $Pd^{(r)} = Pq^{(r)}$.

The earliest time the driver can leave the current location to serve a passenger must be in the future, that is $Po^{(d)}.d_{(min)} > t$

The latest time a passenger can be picked up, must still make possible to arrive to the drop-off location at the latest possible arrival time under the expected travel time.

$$Po^{(r)}.d_{(max)} = Pd^{(r)}.a_{(max)} - t_v\left(Po^{(r)}, Pd^{(r)}\right) \quad (10)$$

The arrival time at the pick-up location must be compatible with the expected time window.

$$Po^{(r)}.d_{(min)} \leq Po^{(d)}.d_{(min)} + t_v\left(Po^{(d)}, Po^{(r)}\right) \leq Po^{(r)}.d_{(max)} \quad (11)$$

The arrival time at the drop-off location must be compatible with the expected time window.

$$Pd^{(r)}.a_{(min)} \leq Po^{(r)}.d_{(max)} + t_v\left(Po^{(r)}, Pd^{(r)}\right) \leq Pd^{(r)}.a_{(max)} \quad (12)$$

This set will be the input for a ride-matching optimization problem performed on an external application or module. An extensive list of methods for this optimization problem can be found in (34), (35), (36) and (3), however they are out of the scope of this paper. Some common objectives include minimizing $det_{(tot)}(d,r)$, or $w_{(tot)}(d,r)$, as well as costs, the total travelled

time or distance. For an agent that tries to start the request automatically as soon as context has been changed (a visit to a POI has been identified), variable t can be replaced by $Po^{(r)}.dep_{(min)}$ that together with other time estimations can be inferred directly from the temporal patterns discovered in the location history.

2.4.3 Inference module

Because a user does not want to receive notifications unless the agent is confident that it is important, the agent must take the decision of going ahead with providing the service or desisting only when the certainty of requiring the service is high.

Supervised learning techniques for classification can be applied to make this decision, as long as some observations could be collected from previous use of the application. Each observation can be labeled as positive whenever the user has decided to use the service and negative otherwise. Besides this label, other values of attributes captured from the current context are preserved, namely time of the day, current and next location, weather report, travel mode, related activity, etc. This way as soon as the agent expects a new trip and finds some potential shareable trips for a ride, a rank classifier will tell how strong a new ride requirement prediction is and proceed to notify the user if a certain threshold is exceeded.

The best value for this threshold can be evaluated with a ROC curve to find that value that produces the highest recall and the lowest false positive rate at the same time. The notification system details are a topic for future research. Although prediction quality is required to anticipate a shareable trip and a passenger's request, this module must also have be in charge of finding similarities among trips' schedules and paths, in order to propose matches for ridesharing.

2.5 EVALUATION

A first test to assess the accuracy of our heuristic for identifying the stay points from data collected by a mobile device GPS during a whole day was done with a controlled experiment.

In (34), a round-trip journey also referred in literature as a tour is mentioned as a potential requirement for a ride-sharing scenario, where riders place two trip announcements so the agreement is accepted only if a confirmed return to the origin is also offered. Our experiment will be designed as a tour plan defined in the following way.

A tour T can be defined as a chain of trips between pairs of locations that starts and terminates at the same point, that is, a sequence of n trips among n different locations where the destination of the last trip is identical to the origin of the first one.

$$\mathrm{T} = \langle \tau_{1\to2}, \tau_{2\to3}, .., \tau_{n-2\to n-1}, \tau_{n-1\to n}, \tau_{n\to1} \rangle$$

Where $\tau_{i\to i+1}$ represents the trip between locations i and $i+1$. A tour can be also represented by a sequence of $n-1$ visited locations where the user spends a time greater than T_{th} (i.e. stay points) together with their arrival and departure time, after excluding the unique arrival/departure location which is not considered for the detection procedure.

$$\mathrm{T} = \langle s_1, s_2, \dots, s_{n-1} \rangle$$

A design of a tour plan in this format can be built, however the arrival and departure times must be updated after the tour is completed according to the actual times registered manually by the user, since they can vary from those in the original plan.

2.5.1 Initial Calibration

The success of the heuristic for detecting stay points depends on the correct identification of the arrival and departure GPS points. This requires the proper calibration of the two thresholds T_{th} and D_{th} and can be stated as an optimization problem.

Let T′ $= \langle \hat{s}_1, \hat{s}_2, \dots, \hat{s}_k \rangle$ with $k \neq n-1$ be a sequence of k ≠ n-1detected stay points identified with our heuristic. Each of them confronted with the original tour plan (i.e. $\hat{s}_i \in T'$ is identified as stay point $s_i \in T$ based on their high proximity).Moreover, some actual stay points may not be detected and some "ghost" stay points corresponding to short unavoidable stops could be included. A possible objective is the minimization of the total time error expressed as the sum of the differences between the detected and the actual times of arrivals and departures (expressed as UNIX timestamps in milliseconds).

$$TTE = \sum_{s_i \in (T \cap T')} (|\hat{s}_i.arr - s_i.arr|) + \sum_{s_i \in (T \cap T')} (|\hat{s}_i.dep - s_i.dep|) + \alpha |T - (T \cap T')| \qquad 13$$

The parameter α is a penalty in milliseconds for the actual stay points in the tour plan that couldn't be discovered (assuming there are no problems due to a bad GPS signal). A constraint could be the mean proximity error, defined as:

$$MPE = \frac{\sum_{s_i \in (T \cap T')} d(\hat{s}_i, s_i)}{|T \cap T'|}, MPE \leq P_{th} \qquad 14$$

Which demands that the detected stay point's average latitude and longitude must be close enough to the actual stay point's coordinates.

A tour can be defined as a journey consisting of various trip legs and starting at home location. To test the stay point detection, a home-based tour consisting of 5 unique destinations was tracked to fetch coordinates when using an average 5-second interval as suggested in (29). As expected, the visits to a same place were registered as different stay points, located close to each other forming clusters of different sizes. With $maxDist = 20m$ and $minPoints = 3$, four out of five different points-of-interest were identified. Discarding the non-detected location, the total time error for different values of T_{th} and D_{th} comprising the 2342 tracked points was computed. A good approximation was found at $T_{th} = 5\,min$, $D_{th} = 40\,meters$ when constrained to $P_{th} = 35\,meters$, resulting in a *TTE* of 618 seconds. With this parameter values, the detected departure and arrival times for each location can be found in Table 1. It has to be taken into account that the app attempts retrieving the user's position every 5 seconds, but if accuracy is low because it is cloudy or users are indoors, the point is discarded; so that the number of collected points is generally smaller than expected.

The process reported a mean distance error of 25 meters between the actual locations and the detected coordinates, moreover an average time error of about 2 minutes when detecting the arrival and departure times. This seems to be an acceptable error for applications that do not demand high precision estimating times. For instance dynamic ride-sharing, since the agent will try to make an arrangement for a ride much earlier than the estimated departure time and then the pick-up and delivery times will be confirmed and updated by the participants before achieving an agreement.

Table 1. Detected time windows for the studied tour

Location	Actual time window	Detected time window
Home	9:15 – 9:30	9:17 – 9:30
Work	9:40 – 12:20	9:41 – 12:21
Alma 3	12:25 – 13:25	12:26 – 13:26
VUB Auditorium	14:59 – 17:30	14:57 – 17:30
VUB Complex	17:37 – 18:45	17:37 – 18:46

2.5.2 Further Experiments

Additional GPS data from 15 anonymous users during 17 days with different GPS usage rates in home-based tours were analyzed for stay points. Results are summarized in Table 2. The small number of points-of-interest detected during this time interval, indicate that the number of locations recurrently visited was small and suggests that the patterns during a larger interval will still be very limited. These results should be combined with information about the travel mode used on each regular trip in order to identify shareable trips among the participants. The mobility patterns for all these users are not enumerated, but given the proximity found among the points-of-interest in space and time, chances for matches are high.

Table 2. Stay points detection and unique locations for 17 days of tracking data on different users.

User	Processed points	Detected stay points	Unique Points of Interest
1	2530	29	3
2	6677	15	3
3	12614	20	2
4	4030	21	3
5	541	3	1
6	2103	6	2
7	3343	2	0
8	4840	10	1
9	5063	7	2
10	6761	36	5
11	2725	10	1
12	1791	4	1
13	1310	11	2
14	2739	15	2
15	1027	7	3
total	**58094**	**196**	**31**

2.6 CONCLUSIONS

A list of experiments to validate the approach presented in this paper were performed. It could be evidenced that with a proper calibration of the parameters a decent approximation of the actual localization of the points-of-interest can be achieved, together with a close estimate of the arrival and departures times. However, sometimes the way a user moves inside a same location avoids capturing stay points. It was observed that those places where users stay "motionless" are more likely to be detected, and also that using a low time threshold is preferred.

Richer contextual information like detecting the connection to a known WI-FI network, detecting when the smartphone has been plugged to the AC outlet, detecting a relevant reduction in speed, etc., should be added to the mining process in order to increase the opportunities of detecting relevant points. It could be observed that the number of mobility patterns and recurrent shareable trips of a user is rather small due to the uniformity in a person's mobility behavior, and that applying a simple Markov chain of higher order make the agent to take decisions with a relative confidence on determining forthcoming trips, however better AI methods for sequence learning and prediction should be applied including not just location for the classification but also more other attributes.

It can be determined that taking into account other forms of contextual information from different sources, not just from physical sensors in the smartphone but from web services and remote databases, an early assistance at short notice by an agent seems completely possible. The match between passengers and drivers through the detection of recurrent regular trips in the demand side and the discovery of shareable trips in the supply side in an automatic way, is a realistic contribution for dynamic ride-sharing systems, thanks to the ubiquitous environments present in modern cities.

The approach described in this paper to detect stay points and then to perform trip segmentation, permits defining a general architecture for context-aware apps for ridesharing, which can provide automatic suggestions based on tracking data, easily retrieved by most existing apps in modern mobile devices.

2.7 FUTURE WORK

At least three main concerns can be addressed. First, the addition of new context information, specifically the detection of transport mode in order to improve the method to recognize stay points when a Walk-leg is detected. Other sources like WI-FI network, detecting when the smartphone has been plugged to the AC outlet among others are also important. The methods for prediction of next locations can be improved and updated to include this new enriched mobility patterns.

Second, the change of scenario from ridesharing to another transportation application. Context-awareness increase the automatic tasks that can be executed by existing systems, so that autonomous agents can provide a better assistance to users. Some possible scenarios to be enriched may be: trip planning, traffic alert systems, self-driving vehicles, etc. A disadvantage of the approach used in Algorithm 2 to identify a point-of-interest from a cluster of stay points, is that chains of stay points being spatially close (which can be expected in large travel histories consisting of several unique locations), could be incorrectly merged into very large points-of-interest. Adding more attributes to the clustering procedure or using a different approach not only based on distances could be helpful in future applications.

Thirdly, the use of better state-of-the-art methods in the rest of the process, for instance in the prediction of the next location, could be included. One may think of patterns in other context data than location information, e.g. calendar data, in actions typically preceding a trip (like checking only for traffic or weather conditions).

Also, machine learning techniques for the inference stage should be selected and tested. Moreover, it may be needed to not only predict next location and departure time of upcoming shareable trips, but also if there are constraints that may render a trip non-shareable, e.g. because the travel mode will not be the car or because the trip will be under too tight time constraints to allow for detours for pick-up and drop-off (e.g. to be discovered from calendar items or schedules of activities at the destination location).

3 DISCOVERING REGULARITY IN MOBILITY PATTERNS TO IDENTIFY PREDICTABLE AGGREGATE SUPPLY FOR RIDESHARING

Mendoza, I., Rydergren, C., & Tampère, C. M. J. (2018). Discovering Regularity in Mobility Patterns to Identify Predictable Aggregate Supply for Ridesharing. *Transportation Research Record*.

3.1 ABSTRACT

Heterogeneous data collected by smartphone sensors offer new opportunities to study a person's mobility behavior. The mobility patterns extracted from the travel histories found in these data, allow agents residing in mobile devices to model transitions between visited locations. This permits upcoming trips to be predicted after observing a set of events and assistance can be planned in advance. When several agents cooperate, the forecasted trips made by multiple users can provide a potential supply for shared mobility systems such as dynamic ridesharing. These trips must be sufficiently regular and frequent to be reliably announced as shareable trips.

This paper describes a methodology to identify a predictable aggregate supply for ridesharing via mobility patterns discovered in users' travel histories. It empirically quantifies measures like regularity and frequency of these patterns, on a dataset consisting of 967 users scattered in different geographical areas. The sample exhibits high heterogeneity with respect to these measures (hence, of predictability, regardless of the prediction method). This paper shows how frequency of trip patterns decreases while regularity increases, when additional dimensions such as departure times are added to the analysis. It was concluded that the flexibility of the travelers on accepting less regular trips, is vital to discover a larger supply. These results provide insights to develop future applications taking advantage of this approach, to increase ridesharing rates, allowing a critical mass to be more easily attained.

Keywords: Ridesharing, Mobility Patterns, Trip Prediction, Mobility Behavior.

Credit author statement

Iván Mendoza: Conceptualization, methodology, software, writing original draft preparation, and visualization / data presentation.

Chris Tampere: Supervision, validation, Writing- Reviewing and Editing.

C. Rydergren: Data collection, validation.

3.2 INTRODUCTION

Mobile applications on modern devices can track a user's activity, producing logs that are used for sports and health purposes; other apps in the "life-logging" category use sensors that allow users to track themselves for an entire day, recording a full chain of events. The heterogeneous data collected by these apps through the sensors of a mobile device, including but not limited to global positioning systems (GPS), accelerometers or gyroscopes, can also provide insights about the person's mobility behavior after being processed. The distinctive information found in these data is the chain of activities carried out during the day, including visited locations, the time spent on these places and more characteristics depending on the complexity of the algorithms.

Data mining techniques, allow extracting mobility patterns from travel histories constructed from these logs, making it possible to identify regular transitions between repeatedly visited locations. These transitions can be conditioned to some detected state or context defined by attributes such as: the current location, time of the day or the day of the week. The regularity of these patterns enables the prediction of future trips; so that software agents residing in a user's mobile device can autonomously decide about planning a service assistance in advance.

One of the transport services that could potentially benefit from this approach is dynamic ridesharing, when multiple agents cooperate within a region. If daily car trips with available seats made by multiple users could be predicted for a certain time interval, day or destination; information about a potential supply for ridesharing matching an upcoming ride request, could be inferred and announced earlier. Ridesharing passengers could for instance visualize the expected supply, consisting of potential shareable trips to a destination of interest compatible with their own schedules. This visual information could display hotspots where the expected trips will occur, so that passengers can find suitable pickup points to enable ridesharing. The possible software applications using this approach, may provide an instrument to attain a critical mass for ridesharing services by promoting an increase in number of participants. In this context, a critical mass refers to having a minimum number of drivers, so that passengers can find supply for a ride requirement; but at the same time, having a minimum number of passengers, letting drivers that want to share their trip costs find sufficient demand for requests. This paper proposes a solution from the supply perspective: assuming that without supply, no passengers can choose to share rides; but without passengers, drivers may still offer the ride and hence create a basis for a community of ridesharing users to grow steadily.

The objective of this paper is to define a methodology to estimate a supply of shareable trips for ridesharing, which can be conditioned to a context, based on the analysis of trip frequency and regularity in travel histories. Then, a subset of these trips with a minimum level of regularity is used to produce visualizations of the potential supply. The key concept of trip regularity is explored empirically in a dataset of travel histories of 967 users scattered in different geographical areas. The present paper is organized as follows: the first section introduces the context of the research and states the paper's objective, then a literature review of related efforts is presented. The next section describes the proposed methodology, followed by a discussion of the results obtained from the empirical data, and in the end the conclusions and potential future work is presented.

3.3 LITERATURE REVIEW

Ridesharing has received considerable attention in recent transportation-related research; This alternative mode of travel helps mitigate traffic congestion as the number of vehicles is reduced, providing environmental and social benefits. A review of some recent research and possible future directions on ridesharing can be found in Furuhata et al.(1), where authors also provide the following unified definition: "ridesharing is a transportation mode in which individual travelers share a vehicle for a trip, with the purpose of splitting travel costs among users with similar itineraries and time schedules". Some of the suggested improvements included better ride-matching strategies, pricing systems and multi-modal integration so that ridesharing is combined with another travel mode to complete a trip. An important statement which led to the current research, states that "the complexity of this problem increases the importance of assistance by software agents to enable personalized travel planning and execution". Some relevant literature related to this research is studied in the following paragraphs.

An attempt to enable ridesharing through a recommender system, implemented as a web-based platform that uses large-scale smartphone's mobility data is presented in Bicocchi & Mamei (37). Here, the authors extracted information from two datasets containing mobility traces to identify potential rides. Routines consisting of repeated transitions on certain days of the week between a user's frequent locations, were found when mining trips endpoints; then rides were matched based on similarities between the origins and destinations of different users' routines. Recently in (38), the previous work was extended through a set of methods, which analyse urban mobility traces to recognize matching rides along similar routes. A list of optimization alternatives for ride-matching can be found in Agatz et al. (34)and a review of techniques to extract different mobility patterns can be found in Lin & Hsu (24).

In Cici et al. (39), the authors provided an upper bound for the potential reduction of traffic in three different cities through ridesharing. Mobility patterns concerning home and work locations found in data from different heterogeneous sources were extracted, then an algorithm for matching users with similar patterns considering additional constraints such as social distance was proposed. Another research exploring the impact of ridesharing on congestion using mobile phone data is presented in Alexander & González (5). Here, the authors extracted the average daily origin-destination (OD) matrix per travel mode from mobile phone data to match trips with spatiotemporal similarities. Then, the impacts on congestion were evaluated by considering different adoption rates.

The potential benefits of introducing meeting points in a ridesharing system to attain a critical mass is evaluated in Stiglic et al. (40). There, the authors used simulation to measure the impact of picking up and dropping off passengers at locations different than the actual origins or destinations, obtaining a significant increase in the number of matched trips. Later in (41), the research was extended by adding flexibility in departure and detour times. At last, a research performed in Goel et al. (42) provides a method to choose the best locations for these pickup points based on Voronoi diagrams. Some existing approaches for evaluating regularity in mobility patterns can be found in Williams et al. (43),Wang et al. (44) or Zhong et al. (45). High regularity of trips leads to a low randomness or entropy in a choice set of destinations; these metrics are extensively used in literature for attribute selection, particularly in classification models such as decision trees(46).

The present paper extends these contributions by using travel data acquired from modern smartphone's tracking apps; later, after a multi-step datamining process, spatial characteristics of potential "sharable" trips are inferred and announced as hotspots to find ridesharing opportunities if patterns are sufficiently regular.

3.4 METHODOLOGY

In this section, the required methods to attain the paper's objective are elaborated. These methods include the extraction of personal points of interest and mobility patterns, when trips occur under certain conditions. Then, frequency and regularity measures are formulated so that a subset of candidate trips can be proposed as the potential aggregate supply for ridesharing.

3.4.1 Extracting Personal Points of Interest

To recognize recurrent visits to a same location in a travel history, the captured spatial coordinates of the trip's endpoints cannot be used directly. Given that coordinates are estimates of a user's position, they can be different on every new visit to a same location. Some methods for detecting trip's endpoints can be found in (21) or (7), and the problem of labelling endpoints as visits to known or new locations in (14), (15). In this paper, a frequently visited location in a user's travel history will be called a personal point of interest (POI). Typical approaches in literature to identify these spatial patterns, use density-based clustering techniques such as DBSCAN or OPTICS. The latter also allows a hierarchy to be obtained, allowing POIs to be extracted at a desired scale (i.e. buildings, neighborhoods, regions, etc.). A general density-based procedure used for the extraction of POIs is now explained.

Let us assume a dataset H consisting of travel histories from multiple users with data collected by a smartphone app. Then, a travel history H^v defined as a chain of trips $b_1, b_2, \dots b_N$ between endpoints where a user v stops during the day to start a new activity is created. The following notation describes each trip i's characteristics.

- $b_i.o$ Origin coordinates {latitude, longitude}, $\forall b_i \in H^v$
- $b_i.d$ Destination coordinates {latitude, longitude}, $\forall b_i \in H^v$
- $b_i.m$ Travel mode, $\forall b_i \in H^v$
- $b_i.s$ Departure time in 24-hour notation, $\forall b_i \in H^v$
- $b_i.e$ Arrival time in 24-hour notation, $\forall b_i \in H^v$
- $b_i.w$ Day type, $w \in \{weekday, weekend\}$, $\forall b_i \in H^v$

Then, the spatial dataset containing coordinates of trips' endpoints to be clustered is:

$$S = \{x_1, x_2, \dots, x_{2N} \mid \forall b_i \in H^v : x_i \in \{b_i.o, b_i.d\}\}$$

The distance between two points, which is typically the Euclidean distance after transforming coordinates to a Cartesian system (47) is denoted by $dist(x_1, x_2)$, then the neighborhood of a point x_i as shown in Figure 3**Error! Reference source not found.**(a) with points directly reachable at a radius $\varepsilon \geq 0$ is:

$$N_\varepsilon(x_i) = \{x_j \in S \backslash \{x_i\} \mid dist(x_i, x_j) \leq \varepsilon\}$$

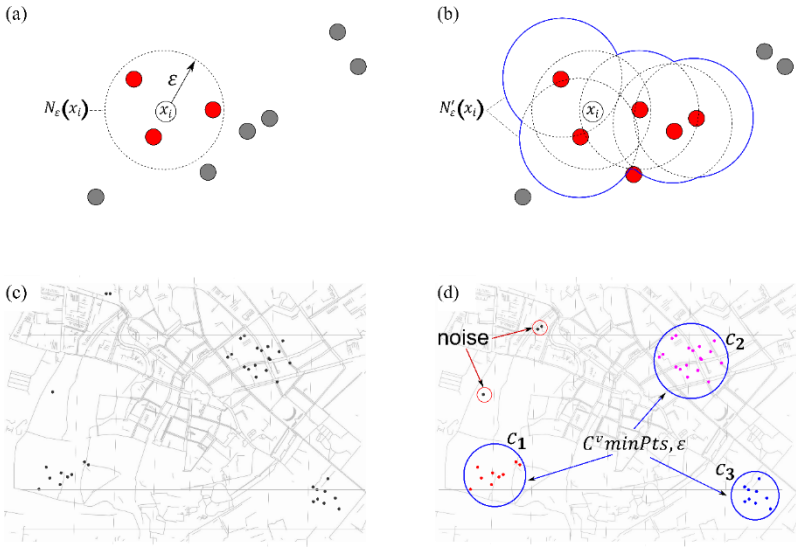

Figure 3. (a) Neighborhood of a point. (b) Complete neighborhood of the same point. (c) Trip endpoints in a travel history and (d) their clustering structure when $minPts = 5$

The complete neighborhood $N'_\varepsilon(x_i)$ shown in Figure 3 (b), consists of all merged points in neighborhoods of those previously discovered, that is,

$$N'_\varepsilon(x_i) = N_\varepsilon(x_i) \cup N_\varepsilon(x_j) \cup N_\varepsilon(x_k) \cup .. \qquad \forall x_j \in N_\varepsilon(x_i), \forall x_k \in N_\varepsilon(x_j), ..$$

Algorithm 3. Finding the complete neighborhood.

Function: Neighborhood $(x_i, N_\varepsilon(x_i)$, dataset S)

For each unvisited $x_j \in S$

Label x_j as visited

If $dist(x_i, x_j) \leq \varepsilon$

Set $N_\varepsilon(x_i)$ = Neighborhood $(x_j, N_\varepsilon(x_i) \cup \{x_j\}, S \setminus \{x_j\})$

Return $N_\varepsilon(x_i)$

Algorithm 4. Entire density-based procedure.

Function: DB-Clustering (dataset S)

Set $C = \emptyset, k = 0, \varepsilon \geq 0, minPts \in \mathbb{Z}$

For each unvisited $x_i \in S$

Label x_i as visited

Set $N'_\varepsilon(x_i)$ = Neighborhood$(x_i, \{x_i\}, S \setminus \{x_i\})$

Set $k = k + 1$

Set $C = C \cup new\ cluster(N'_\varepsilon(x_i), k)$

Return C

The process to obtain $N'_{\varepsilon}(x_i)$ is presented in

Algorithm 3, and the final density-based procedure to find the set C of repeated locations in Algorithm *4*, where complete neighborhoods are included in the final result only if a minimum number of points (visits) specified by parameter $minPts \in \mathbb{Z}$ is reached, otherwise they are tagged as noise, see Figure 3 (c ,d). Then, the set of POIs in user v's travel history is:

$$C^v minPts, \varepsilon = \{c_i \in C \mid minPts \leq |c_i|\}$$

3.4.1.1 Identifying Transitions between POIs

Following this process, the trip endpoints are tagged to identify the cluster they belong to, so that all visits to the same location have the same label. That is, assuming u_k is a unique identifier for cluster c_k and x'_i the label on point x_i, then:

$$x'_i = \begin{cases} u_k & if\ x_i \in c_k \\ undefined & otherwise \end{cases}$$

Let u_0 be a label to denote endpoints at non-frequent locations, that is those not included in a cluster. The following new notation is required for the updated trip i's characteristics.

- $b_i.o'$ Trip origin's label, $\forall b_i \in H^v$
- $b_i.d'$ Trip destination's label, $\forall b_i \in H^v$
- u_0 Label for "noise" points, where $|c_0| < minPts$

Two trips b_i and b_j by the same person are then assumed to have the same origin-destination (OD) pair if $b_i.o' = b_j.o' \wedge b_i.d' = b_j.d'$, even though their actual endpoints' coordinates could be different. If the travel history is consistent it should be expected that $b_i.d' = b_{i+1}.o'$, $\forall b_i \in H^v$, producing a reliable chain of trips. After transitions between pairs of POIs are identified, regularity and frequency of these transitions with respect to their travel histories can be quantified.

3.4.1.2 Regularity of Transitions between POIs

In this paper, regularity of a certain trip characteristic (i.e. destination) is defined as the relative frequency of that characteristic in a user's travel history, conditioned to a state defined by other characteristics; this relative aspect distinguishes regularity from frequency. Frequency is the number of events per unit of time; it denotes the probability that something happens, such that the expected number of events in a period is found as $frequency\ x\ length\ of\ period$. On the other hand, regularity is the relative frequency of an event i within a user's pattern: it denotes the probability that among all possible events in the given conditions, i is the current one. Values closer to 1 are more regular and hence easier to predict.

A high frequency does not necessarily imply high regularity or vice versa. For instance, an event (e.g. to travel from A to B) may be rare (e.g. only once per month), yet any time the person resides in A his next move may be to go to B, hence his move is 100% regular, conditional on residing in A. An agent observing this person for a while and identifying its current location to be A, may therefore have an easy job predicting B as the next destination. Inversely, a person may very frequently travel between C and D (e.g. twice per day); nevertheless, when residing in C, there may be 5 more very frequent destinations other than D, which makes the trip C-D frequent but not very regular when only conditioned on residing in C. Therefore, an agent that knows its current position in C, cannot predict the next destination with confidence. The

following functions provide different measures of regularity of a destination in a user v's travel history.

Let $y_{i,j}$ be the number of trips in H^v between POIs u_i and u_j:

$$y_{i,j} = \left|\left\{b \in H^v \mid b.o' = u_i \wedge\ b.d' = u_j\right\}\right| \tag{15}$$

Let V be the set of all users with consistent travel histories. Passengers requesting a ride to a location q could potentially use trips toward the destination's neighborhood.

$N_\varphi(q) = \{b.d' \in\ H \mid dist(b.d', q) \leq \varphi\}$, where

$$H = \bigcup_{v\,\in\,V} H^v$$

That is, $N_\varphi(q)$ contains all POIs destinations in every user's travel history nearby location q, where φ could denote the maximum distance a passenger is willing to walk from the drop-off location. Let P be the number of unique POIs in a user's travel history, the total number of inter-cluster trips to destination $u_j \in N_\varphi(q)$, also including origins in u_0 is:

$$y_{*,j} = \sum_{\substack{p=0 \\ j\neq p}}^{P} y_{p,j} \tag{16}$$

The regularity of a u_j with respect to other destinations in H^v, is defined by:

$$R^v(u_j) = P(u_j|v) = \frac{y_{*,j}}{|H^v|} \tag{17}$$

That is, the probability that user v visits location u_j without being conditioned to any current state. On the other hand, the frequency, measured in number of visits per day to u_j is:

$$f(u_j) = \frac{y_{*,j}}{T^v} \tag{18}$$

where T^v, is the size of user v's travel history in days. The total number of trips from an origin POI u_i, including destinations in u_0 is:

$$y_{i,*} = \sum_{\substack{p=0 \\ i\neq p}}^{P} y_{i,p} \tag{19}$$

The regularity of transitions to u_j (OD regularity) from an origin u_i is then:

$$R^v(u_j|u_i) = \frac{y_{i,j}}{y_{i,*}} \tag{20}$$

Also, transitions conditioned to a certain departure time interval can be evaluated. First, the 24-h format of departure times used in the dataset is transformed to a discrete value, so that trips with similar times can be clustered in a same group. Consider a finite number of time segments ω during the day. For instance, with a time interval $\Delta t = 15\,min$, a day consists of $\omega = 96$

segments. The resulting departure and arrival time periods are $b.s' = \left\lfloor \frac{b.s}{\Delta t} \right\rfloor + 1$ and $b.e' = \left\lfloor \frac{b.e}{\Delta t} \right\rfloor + 1$ respectively, with values: $\{x \in \mathbb{Z} \mid 1 \leq x \leq 96\}$. Then the regularity of destination u_j conditioned to an origin and departure is:

$$R^v(u_j|u_i, s_k) = \frac{y_{i,j,k}}{y_{i,*,k}} \tag{21}$$

where $y_{i,*,k}$, represents the number of trips starting from u_i at departure period s_k, defined as:

$$y_{i,*,k} = |\{b \in H^v \mid b.o' = u_i \wedge b.s' = s_k\}| \tag{22}$$

while $y_{i,j,k}$, considers only those to destination u_j, that is:

$$y_{i,j,k} = \left|\left\{b \in H^v \mid b.o' = u_i \wedge b.d' = u_j \wedge b.s' = s_k\right\}\right| \tag{23}$$

At last, the frequency of these trips would be:

$$f(u_j|u_i, s_k) = \frac{y_{i,j,k}}{T^v} \tag{24}$$

More dimensions can be added, although the number of trips constrained to the new conditions is significantly reduced, as well as their frequency. These new conditions can include day type: $b_i.w \in \{weekday, weekend\}$ and travel mode: $b_i.m \in \{car, public\ transport\}$.

Definition. The overall regularity $R^v(u_j|\theta)$ of destination u_j in travel history of user v, conditioned to a multidimensional state $\theta = (\theta_1, \theta_2, ..)$, is defined as the regularity (and hence predictability) of a trip's destination identified by u_j, based on all the information in θ an agent can access when deciding to announce a potential trip.

3.4.2 Identifying the Aggregate Supply

Contrary to selecting the location with the highest probability to be the next trip's destination, as done by a typical predictor (i.e. a classifier), all patterns above a minimum regularity value are considered by the agent. So that, when a threshold ρ is applied to filter out unreliable trip predictions, the expected trips of user v conforming conditions stated in θ are:

$$H^{v,\rho}(\theta) = \{b \in H^v \mid \forall j \in \theta: b.\theta_j = \theta_j \wedge R^v(b.d'|\theta) \geq \rho\}$$

For a specific passenger's request to location q; the fraction of the aggregate supply, that is, the car trips subset that could potentially be used for ridesharing is:

$$H^\rho(\theta, q) = \{b \in H^\rho(\theta) \mid b.d' \in N_\varphi(q) \wedge b.m = 'car'\}, \text{ where}$$

$$H^\rho(\theta) = \bigcup_{v \in V} H^{v,\rho}(\theta)$$

The relevance of a trip pattern when presented to a passenger, depends on how accurately it can be predicted and how frequent it is. Then the relevance of trip b_i can be quantified by: $predictability \times frequency$. Regularity denotes the probability of making a trip conditioned to some known state, and allows identifying which characteristics are relevant to correctly predict a specific destination; then predictability is a function of regularity and correlates positively (the more regular the more predictable), so that they can be used interchangeably.

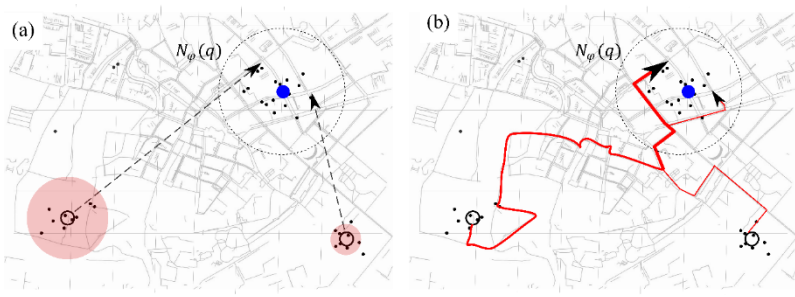

Figure 4. (a) Origins (red circles) of trips neighboring a location (blue circle), acting as ridesharing hotspots which relevance is denoted by the radius size, (b) Trip's paths, where relevance is specified by line weight.

A first visualization highlights the origins of this predicted aggregate supply, so that these hotspots can be used by passengers as pickup points. A hypothetical example (including real trips with assumed paths) is shown in Figure 4 (a), where origins of trips in $H^{\rho}(\theta, q)$ denoted by red circles, have different relevance for passengers, which is indicated by their radius size. Let r_i be the circle's radius for trip i's origin, then:

$$r_i = \lambda R^{v}(b_i.d'|\theta) \times f(b_i.d'|\theta), \qquad \forall b_i \in H^{\rho}(\theta, q) \tag{25}$$

where λ, is a configurable scaling parameter to correctly visualize the plot. Another useful information are the predicted paths used by those trips so that passengers can find a ride on route as shown in Figure 4 (b). The line weight of links in a network $G = (V, E)$ can represent the link relevance, calculated by:

$$w_e = \lambda \sum_{b_i \in H^{\rho}(\theta,q)} R^{v}(b_i.d'|\theta) \times f(b_i.d'|\theta)\, z_{i,e} \tag{26}$$

where w_e is the line weight for link $e \in E$, $z_{i,e} \in \{0,1\}$ is a binary variable indicating whether link e is included in the predicted path of trip b_i and λ again a scale factor for visualization. The line weight is then determined by the number of trips using that link, and also by each trip's predictability and frequency.

3.5 Results and Discussion

The above methods are now applied to a dataset with the travel histories of various users collected through a smartphone tracking application called MOVES. The dataset is briefly described below.

3.5.1 Dataset Description

The MOVES dataset contains travel histories of users in different geographical areas represented by unimodal trip chains. The variation in size of travel histories with respect to the number of days and trips is presented in Figure 5 (a, b).

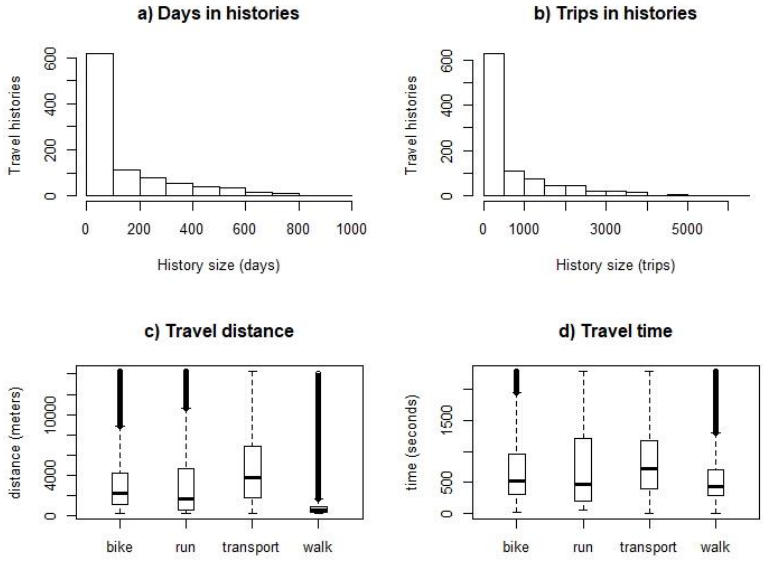

Figure 5. Heterogeneity in travel histories in MOVES dataset. (a) Size in days, (b) in number of trips. (c) Distribution of travel distance and (d) time in the dataset.

To evidence the need of further filtering, as expected when treating big data, a summary of the main characteristics of this dataset is given:

- 967 unique registered users as well as travel histories.
- 692,306 registered trips in 1,070 days.
- The time frame of travel histories ranges from 2 to 928 days.

The travel modes automatically inferred by the app are walking, running, cycling and motorized vehicles; the app cannot differentiate trips by private cars from public transport.

The distributions of travel distance and time per travel mode are plotted in Figure 5 (c, d); outliers corresponding to very infrequent trip's characteristics have been filtered out by using Tukey's method (48). These plots exhibit high heterogeneity in data, nevertheless most trips are short both in time and distance.

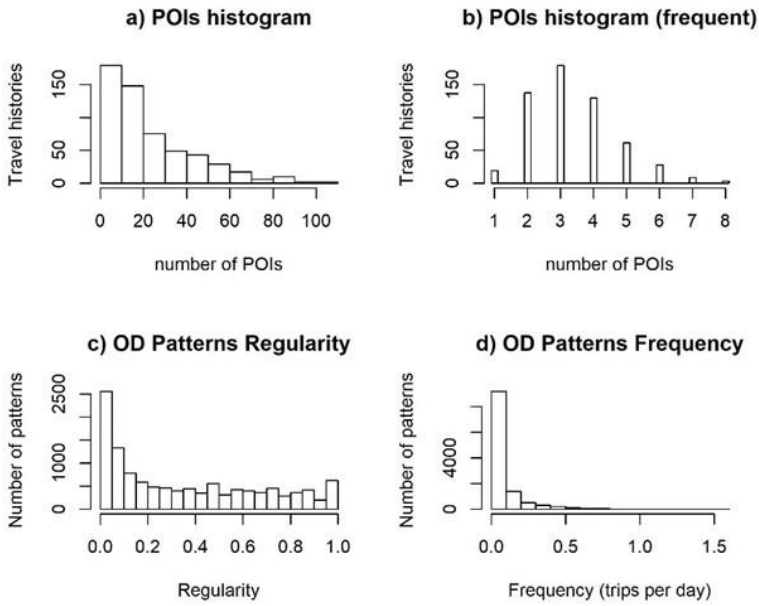

Figure 6. Variation in number of POIs per travel history: (a) those with a minimum of 5 occurrences in the entire history, (b) those with at least 2 visits per week. (c) Regularity and (d) frequency of the OD patterns

The patterns studied in the next section were extracted from histories with at least 30 days of tracking, reducing the number of users to 573. The travel mode of trips was not considered in the study either.

3.5.2 Extraction of mobility patterns

Since parameter ε, corresponding to the search radius for the data mining procedure must be previously calibrated, trips which lengths are found at a distance smaller than this value were previously filtered out since they are possibly not useful for ridesharing. This avoids including very short trips (not useful for ridesharing), nevertheless circuits with the same origin and destination are still allowed. The heterogeneity in number of POIs per user, when $\varepsilon = 200\ meters$ and $minPts = 5$ is presented in Figure 6 (a). Here, many POIs per user were found due to the low number of required visits specified by parameter $minPts$; nevertheless, the number of actual points of interest a user regularly visits is much smaller, as seen in Figure 6 (b) when only those with at least two visits per week were considered.

The following patterns correspond to repeated trips between the same pair of origins and destinations (OD patterns) after removing intra-cluster trips, when constrained to a minimum of observations denoted by a parameter $minTrips$. The pattern's regularity, indicating the probability of the trip's destination conditioned to an origin, and the frequency denoting the average number of times the pattern was observed per day are presented in Figure 6 (c, d).

Table 3. List of Most Regular Users with respect to their OD Patterns

User ID	OD patterns	Trips in patterns	Average frequency	Average regularity	POIs
U777	3	67	0.35	0.81	2
U992	3	109	0.54	0.71	3
U603	3	562	0.39	0.68	3
U000	3	236	0.3	0.65	2
U873	3	142	0.68	0.61	3
U211	3	166	0.32	0.6	3
U980	3	43	0.41	0.58	4
U552	2	87	0.29	0.93	2
U589	2	159	0.4	0.9	3
U519	2	176	0.61	0.85	3

Table 4. List of Most Regular Users with respect to their OD-Departure Patterns

User ID	ODD patterns	Trips in patterns	Average frequency	Average regularity	POIs
U516	4	53	0.44	0.99	3
U254	2	44	0.33	0.98	3
U925	2	21	0.31	0.96	3
U992	2	74	0.55	0.95	3
U098	2	33	0.34	0.94	3
U777	2	50	0.39	0.88	3
U040	2	93	0.37	0.87	3
U730	2	42	0.34	0.86	3
U282	2	40	0.43	0.85	3
U661	2	67	0.47	0.83	3

The users with the largest number of OD patterns are now presented in Table 3, when a minimum regularity level of 0.5 and a minimum frequency of two trips per week are applied. The columns have the following meaning: a unique user pseudo-identifier, the number of OD patterns, the number of trips in history contained in the OD patterns, the average daily frequency, the average regularity and at last, the number of unique POIs observed in the patterns.

Next, with respect to spatiotemporal regularity, the users with most patterns are displayed in Table 4 when adding departure time intervals, with parameters $minTrips = 5$ and $\Delta t = 15$ minutes. The corresponding regularity and frequency of these patterns can be seen in Figure 7 (a, b). As noticed, when more dimensions are added to the analysis, the number of trips and their frequency are reduced, therefore, the supply for more specific requests (e.g. those also

including a desired arrival time) will be reduced; nevertheless, regularity and therefore predictability of patterns rises due to the increased availability of context information.

Table 5. List of Most Regular Users with respect to their OD-Arrival Patterns

User ID	ODA patterns	Trips in patterns	Average frequency	Average regularity	POIs
U516	4	57	0.48	1	3
U966	3	80	0.33	1	3
U873	3	65	0.31	0.96	3
U867	2	38	0.35	1	3
U992	2	77	0.57	0.97	3
U254	2	43	0.33	0.96	3
U980	2	22	0.31	0.92	3
U282	2	37	0.39	0.89	3
U777	2	48	0.38	0.88	3
U040	2	91	0.36	0.86	3

The most regular users with respect to OD-arrival patterns, which includes trips between repeated OD pairs arriving at regular times are displayed in Figure 7(c, d) shows the corresponding frequency and regularity of these patterns. Accuracy in predictions is always wanted; nevertheless, if agents are less demanding when announcing upcoming trips, the potential aggregate supply would be sufficiently large to encourage passengers to participate. The findings support the idea that stronger trip patterns only involve a few POIs per travel history, as observed in the last two tables and previously suggested by Figure 6 (b). As the pickup hotspots of the potential aggregate supply conditioned to a hypothetical ride request, include all predictable patterns; collecting several data about the current context is necessary, however, an agent should select only a few characteristics to announce the predicted trips, since looking for too specific and predictable patterns would produce a short or even empty supply.

3.5.3 Dealing with other datasets

The methods used in this paper can be used with datasets from other existing tracking apps, as long as trip chains can be retrieved containing spatiotemporal characteristics. Accuracy regarding mainly the collection of spatial data may differ; because even though sensors and operating system programming interfaces are shared among mobile apps, each app may have its own inference and aggregation methods. Another wanted information, not included in the previous analysis is the used travel modes. MOVES unfortunately does not differentiate motorized vehicles; though in some cases, modes can be inferred from existing data such as waypoints and instant speeds. It must be noticed that databases consisting of raw data directly retrieved from sensors, should first be converted to a trip chain format; this means that trip endpoints (stay points) must be identified prior to data mining procedures.

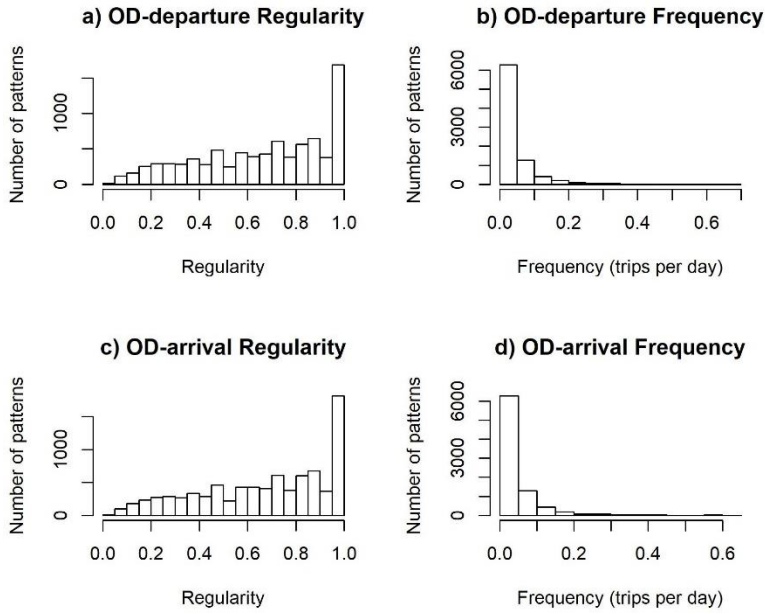

Figure 7. (a) Variation in regularity when adding departure period, (b) frequency of the OD-departure patterns. c) Variation in regularity when adding arrival period, (d) frequency of the OD-arrival patterns.

A main concern for research using user data is privacy. An appropriate data collection mechanism to certify that information is stored anonymously and with the explicit user's consent is compulsory. The methodology described in this paper, does not require tracks to be stored in a remote server; instead these can reside in the smartphone for their further use by software agents, avoiding the needless flow of data through insecure channels. At last, the predicted aggregate supply consisting of the announced trips by other agents, does not need to include any driver's contact information.

3.6 CONCLUSIONS AND FUTURE WORK

A methodology to learn mobility patterns at different levels from smartphone data collected by apps tracking a user's daily activities has been presented, together with different measures to evaluate their relevance for ridesharing. The origins and trajectories of the predicted trips through these patterns, represent hotspots for ridesharing where passengers may find potential aggregate supply conditioned to their ride requirements. For ridesharing, regularity that allows accurate predictions of mobility patterns, and frequency are important to discover a realistic supply. It has been shown that mobility regularity can be identified from smartphone data via data mining procedures, although heterogeneity in the travel histories is high. The number of patterns and their frequency have been found to decrease when more trip characteristics are considered, although regularity and prediction accuracy with respect to a set of conditions will increase. Flexibility on accepting trips with lower regularity, as well as different pickup windows or locations from their actual origins, is important to discover a larger aggregate supply. The method described in this paper is believed to allow the design of future apps that will help increasing ridesharing rates.

Some future directions may include adding travel mode in mobility patterns, since the users' role in ridesharing depends on this characteristic. The regular trajectories of an OD would be a nice addition in order to share parts of the trip, instead of matching users only by the trip endpoints.

The potential of ridesharing may be evaluated on specific regions via other datasets or activity-based simulators. The latter approach can be attained through the generation of a synthetic population consisting of users with heterogenous mobility behavior; for instance, by transferring the distributions found in patterns extracted from datasets generated by tracking apps to another dataset containing demographic data. Each possible scenario for the simulation would involve different ridesharing penetration rates, as well as thresholds of regularity and frequency, so that in each case the supply announced by an agent is different. Additionally, given the importance of having consistent travel histories (which affects efficiency of proposed approach), and that this highly depends on the correct detection of the unique locations visited in a history, using better techniques for this task or producing new algorithms is a key aspect in future research.

3.7 ACKNOWLEDGEMENTS

This research project was financially supported by the National Secretariat of Higher Education, Science, Technology and Innovation of Ecuador (SENESCYT).

AUTHOR CONTRIBUTION STATEMENT

The authors: Iván Mendoza (I.M.), Clas Rydergren (C.R.) and Chris M.J. Tampère (C.T.) confirm contribution to the paper as follows:

Study conception and design by I.M. & C.T.

Data collection by C.R. (data collected via MOVES app).

Analysis and interpretation of results: I.M, C.R. & C.T. (on multiple stages and revisions).

Draft manuscript preparation by I.M.

All authors reviewed the results and approved the final version of the manuscript.

4 AN ITERATIVE METHOD FOR EXTRACTION OF PERSONAL POINTS-OF-INTEREST FROM LIFE-LOGGING DATA

Ivan Mendoza [a, b, *], Chris M. J. Tampere [a]

[a] KU Leuven, L-Mob, Leuven Mobility Research Center, CIB, Leuven, Belgium

[b] Faculty of Science and Technology, Universidad del Azuay, Cuenca, Ecuador

Submitted to Pervasive and Mobile Computing

4.1 ABSTRACT

Retrieval of travel chains from users' travel histories is required for the construction of travel behavior models; detecting the end points of a trip in places like home or workplace is one of the initial and possibly most important steps to acquire these chains. Several research efforts have been reported in this field, mainly using density-based data mining algorithms. However, they are sensitive to spatial parameters that must be calibrated beforehand, providing wide-ranging results in different settings in travel stories. This document proposes a novel algorithm that combines spatial information with spatio-temporal patterns found in previous link chains of a travel history, automatically constructed from life-logging data, improving prediction of the next trip using the knowledge that is continuously learned. Finally, experiments are carried out on real-life data sets to test the proposed method and the results obtained are compared with a popular density-based method, showing an improvement in the detection of the real end points of the trip, as long as patterns or regular behavior can be detected in travel stories.

***Keywords**:* trajectory mining, significant places, travel behavior, context-awareness

Credit author statement

Iván Mendoza: Conceptualization, methodology, software, writing original draft preparation, and visualization / data presentation.

Chris Tampere: Supervision, validation, Writing- Reviewing and Editing.

* Corresponding author, email address: imendoza@uazuay.edu.ec

4.2 INTRODUCTION

In the field of transportation planning, retrieval of travel chains from users' travel histories is a key aspect of modeling travel behavior; This ordered sequence of visited locations allows anticipating the next stop(s) in a travel chain through trained models with the observed data, allowing a decision-making plan for future mobility behavior.

Detecting travel endpoints in frequent places such as: home or workplaces is one of the initial and possibly most important steps to correctly identify these chains. Several approaches have been reported. However, they are sensitive to spatial parameters that need to be calibrated beforehand, providing wide-ranging results across different scenarios in travel histories, especially when locations of different sizes exist. For example, different locations (houses, shopping malls, stadiums) can naturally cover areas with different dimensions. This makes it difficult to calibrate the detection algorithms and may produce incorrect travel chains, since using a large clustering radius will merge neighboring locations covering small areas, and the use of a small radius will detect several independent locations instead of one single large place.

4.2.1 Definitions

To understand the complexity of translating spatio-temporal data to travel chains, a few definitions must be established based on current literature about the topic.

A **stay point**, or simply "stay" is defined as a small geographic region around a centroid with x, y coordinates, where a user dwells for a minimum time t to carry out some activity, some common examples are a school, workplace or shopping. This concept is important because it allows estimating the time and location of the end points of a trip between two successive stays; so that the origin, destination, departure, and arrival times can be inferred. In addition, the captured points can define the trajectory of the trip. In general, x and y are obtained by calculating the centroid of the data points collected during the stay (when the user is not moving) and t should be chosen wisely so that very short breaks are discarded, also requiring to calibrate this parameter beforehand.

A **personal point of interest** (POI) is defined as a recurring destination in a user's history with respect to a minimum number of visits (in absolute terms or relative to a time unit). They can be common among various users (e.g. schools, shopping malls) but also personally relevant (e.g., home, friend's house). Every time a same location is visited, a new stay with different coordinates is added (since the position of a user can only be approximated by triangulation). Identifying a POI requires grouping the coordinates of several stays through a search radius ε depending on how small or how large a POI can be and requires a minimum number of points γ (i.e. visits). Note that with $\gamma = 1$, one would obtain all possible unique destinations visited by a user in the travel history. In the reviewed literature, POIs are also referred to as significant places, significant locations, semantic locations, and more.

A **travel chain of order *n*** is a sequence of *n* subsequent POI's, therefore a trip is the smallest of these chains when $n = 2$, that is a single link joining the origin and its destination. Then, a **travel history** can be defined as the largest possible travel chain for a single user, considering all trips made since the user was tracked for the first time. It consists of subsequent unique destinations that can be repeated (a single destination can appear several times in different trips). Moreover, it keeps expanding as new stays are identified.

The three definitions include procedures for their detection or generation, called in this document: stay detection (or travel segmentation), POI labeling, and travel chain generation.

4.2.2 Paper contribution

Since generation of travel histories depends on a correct detection of stays and POIs, and these concepts are strongly related to the calibration of the before-mentioned parameters, many attempts have been made in previous research to overcome these difficulties. In contrast to mobility data from surveys or certain apps where users need to specify locations and times explicitly, lifelogging allows tracking mobility behavior in a passive way taking advantage of the omnipresence of smartphones. One disadvantage is that continuous access to location services is required, also abundant sampling and high precision of sensors are desired. Additionally, GPS coordinates are only estimates the accuracy of which depends on different factors as cited in (49), making it more difficult to correctly retrieve the actual user's travel history.

This paper proposes a novel iterative algorithm to improve the detection of the upcoming POI's from a set of stay points by using the previously learned knowledge. This is achieved by combining spatial information with patterns found in earlier chain links of the travel history. Then, these patterns allow estimating the probabilities of the origin-destination pairs at certain departure times, so that POI labels are decided considering not only the proximity to a known visited location, but also the a priori likelihood of that location, inferred from temporal variables and correlations between the visited locations. In contrast to most approaches, this new approach incrementally improves POI detection and consistency of travel histories as new information is acquired. This supports producing better models for next-trip prediction or unsupervised learning. To illustrate the methodology, some tests are first performed on a single user's history to evaluate the classification performance. Then, several histories with heterogeneous spatial information are parsed, showing that our algorithm can detect the correct POI label in the trip chain and is much less sensitive to search distances than classical density-based techniques such as (50) or (51).

This document is organized as follows:

- An introduction stating the intended contribution of this paper is presented, together with a review of recent literature and the exhibited limitations.
- The proposed algorithm is introduced and tested on a fraction of a single user's history from a real-world dataset.
- Tests are performed on several users' travel histories with heterogeneous spatial information from the same dataset.
- The conclusions and recommendations for future work are given.

4.3 Literature Review

The current section provides of related literature on production of travel histories, including the aspects mentioned before, that is stay detection (called in many ways throughout the literature), POI labeling and travel chain generation. In some cases, they only cover part of the process or they do not give details about the big data collection techniques.

One of the first contributions can be found in (14). In this paper, Ashbrook and Starner clustered GPS data into meaningful locations at multiple scales to produce a Markov model that can be consulted by intelligent agents for a variety of context-aware applications. They used a threshold t of 10 minutes for stay detection and then a variant of the K-means family of algorithms (52) to find the POIs. At last the model is produced after computing the probabilities of the transitions. Later in (15), they extended their work by improving the stay-detection algorithm when a loss in

GPS signal (specially inside buildings) is detected to determine whether a user has reached a new location.

In (51), Zhou et al. replaced the K-means variants by density-based clustering, which allows clusters of arbitrary shapes and classifies unusual points into noise (since not every point may be considered a meaningful place). The introduced algorithm is called DJ-Cluster and is based on the classical DBSCAN. At last they added a few temporal constraints to reduce computation times. Later in (18), the authors extended their work when classifying a POI by its relevance based on the number of visits and frequency of each location. Other authors in (53), recycle DJ-Cluster for next-trip prediction by using the POIs transitions to produce a Markov Chain of order n, that is the next state (POI) depends on the past states. They called this model a n-MMC, which stands for Mobility Markov Chain considering the sequence of the n previously visited POIs.

In (54), Kang et al. presented one of the first approaches mentioning a source different than GPS. They used Wi-Fi access points to approximate a user's position. When a client device (e.g. smartphone) explores for nearby access points, it receives their mapped geographic coordinates and then estimates its position by computing the coordinates' centroid. At last, they used temporal features to cluster points into unique stays prior to detect POIs. In (15), authors also used raw GPS data. Once stays had been identified with the typical procedure, they used a relational Markov network to label (classify) the POIs (which they call activities such as: at home, at work, shopping) based on attributes such as: activity duration, time of the day, day of the week and so on. Finally, they used unsupervised learning techniques to learn transportation routines between pairs of POIs, such as: frequent trajectories. Later in (55), they used previous knowledge to learn only based on transport routines, locations such as bus stops and parking lots where the user frequently changes the mode of transportation. In (20), Nurmi et al. presented a non-parametric algorithm for place identification called Dirichlet Process Clustering (DPCluster) algorithm. Its main feature is guessing the correct number of POIs based on probability distributions. It consists of two phases: first they grouped some points into clusters and labeled the remaining data as noise; and then they re-estimated the labels by finding the cluster which has the largest likelihood to generate the point. At last, they pruned the results by dropping those clusters with high inter-cluster variance or short cumulative stay time.

In (21), Zheng et al. used a direct way to detect stay points. They looked for regions where GPS points are clustered together, that is where consecutive measures are close given a specified radius. Then they evaluated the "stay time" computed by the difference in time between the oldest point (that one with the lowest timestamp) and the newest. If this difference exceeds a given threshold, a stay is found, and trips are segmented. At last, for each user they extracted locations of interest, then produced clusters of stays that belong to several users into tree hierarchies at a region level before mining POI's sequences. Besides the temporal aspects, most approaches so far use DBSCAN extensions to rely on the detection of stops in trajectories assuming those regions are denser with respect to those when the user is moving. In (56), authors updated this algorithm by replacing the $minPoints$ parameter to estimate density, with a new measure (taken from data fields theory (57)) which quantifies the interaction with other points instead of looking for points in a neighborhood .

In (58), the authors identified stay points like in previous efforts but also including the instant speed of each GPS point, so that stays must only be found on very low speed traces. Later, they use OPTICS (50) and K-Means (52) to cluster neighboring points into POIs. Finally, they split or merged clusters by considering reverse geocoding information as well as temporal features. Another approach to merge neighboring clusters in case there is enough evidence of common patterns can be read in (59), where authors used various data sources such as: sleep periods,

battery charge sessions, and physical activity based on step counts via wearable devices. This drastically improves stay detection by increasing the clustering dimensionality.

In (60), Lee at al. notably improved stay detection using a super-state model to describe a user's transitions. Each super state is composed of proportions of 5 different representative states: staying indoors, moving indoors, staying outdoors, walking, and in-transport. Then, patterns are detected when transitions between two different (or equal states) occur. The algorithm decides whether a user has stopped moving to start an activity, allowing detecting a new stay. In (61), Bhattacharya et al. proposed a method called POI-ID to rank a set of POIs given some stay's data points. For this, they increased the accuracy of a POI's coordinates from a set of unreliable GPS points belonging to a stay. For each of these GPS points they sampled m synthetic points inside a confidence circle (assuming GPS measures are normally distributed), then they joined consecutive synthetic points with line segments. At last, they rank known POIs to snap a stay, based on how often the segments intersect each POI's geometry defined by a polygon. An alternative approach is given in (62), where stays' coverage areas based on the points they contain are transformed into polygons or geometries of interest, via a fixed grid for a specific region. Stays are finally matched to well-known POIs if geometries are similar.

With respect to travel chains which is relevant for next-trip prediction and for retrieving a travel history, in (32), the authors divided a region into cells so that they can mine patterns on inter-cell transitions. Then they inferred association rules to predict the next cell in the chain. In (63), Ying et al. extracted what they call semantic trajectories per user, defined as sequences of labeled locations (what in this document is called a POI) and then clustered trajectories from several users with similar geographic behavior. At last they illustrated a framework called *SemanPredict* that uses both the semantic and the geographic patterns for the next-location prediction. In (64), Pappalardo et al. exploited collective information and a gravity model to drive the movements of an individual when exploring new places in a region. They used the relevance of POIs (measured by the total visits made by other users and its distance to the user's origin) to decide about the next trip's destination. In (65), Feng et al. analyzed the next-location prediction problem as a multi-classification problem with a limited discrete-location list. They trained a recurrent neural network with sequences of spatiotemporal points (POI label and timestamps). A similar approach using Bayesian networks for location prediction when conditioned to previous states can be read in (66).

More literature on processing trajectory data for different purposes such as: POI detection, next-location prediction, or mining mobility patterns for a specific objective can be found in (67) and (68). The algorithm presented in this paper, improves the detection and labeling of POIs from a set of stay points produced on the initial stage of the travel history generation from life-logging data. The main difference is the iterative aspect which is regularly learning new patterns as new data is available. This allows errors in POI labels to be corrected to form accurate travel chains.

4.4 METHODOLOGY

This paper proposes a novel algorithm called *Iterative POI detection* (IPD) to label points-of-interest and retrieve an accurate travel chain by iteratively mining patterns in existing data in a travel history. The general process includes the following steps, which are summarized in the flow chart of Figure 8 and is now explained. It is assumed that the endpoints of each trip have already been detected by a segmentation procedure such as (21).

1. At a first stage, an initial subset T_0 of n trips from a non-processed travel history T is used to find a starting set of unique locations by exploring neighboring stays using some initial radius r, as will be described in the details of the initialization process.
2. Each POI relevance is estimated based on the trips it attracts, so that marginal probabilities where no other events are considered are computed. Then, origin-destination patterns, that is recurrent transitions between pairs of POI's are learned, and conditional probabilities under other events (arrival time, stay time, previous location, etc.) for a given POI are estimated too.
3. Next, as data comes available a new subset of trips T_n is included in the analysis
4. POI labels for these new trips are allocated and labels of previous stay points are re-assigned if necessary, according to a similarity function that considers the learned patterns and probabilities in previous steps. An iterative process begins looping between steps 2, 3 and 4 using a new radius increased by a factor g to explore further until no more trips in T are left and until re-labeling ends in the inner loop.
5. At last, the final trips chain T_f is retrieved after filtering out infrequent transitions, which differs from the initial trips set since it has POI labels instead of only stays' coordinates.

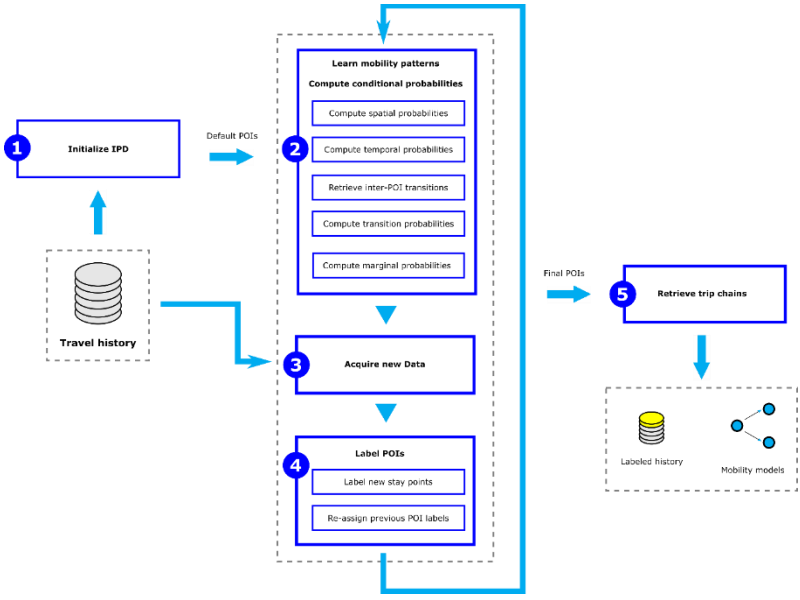

Figure 8. Workflow for Iterative Detection of Points-of-interest.

This process implemented at a high level in Algorithm 5, which includes the following instructions:

- In line 2, a first set of trips T_0 (sufficiently large to find some recurrent behavior) is fetched from history via function $getTrips()$.
- In line 3, the initial set of personal points of interest "$pois$" is retrieved via density-based clustering applying a search radius ϵ in function $ipdInitialize()$.
- In line 4, the trips are copied to travel history T_f

- In line 5, a new subset of trips arrives in T_n
- In line 6, the search radius is increased by a factor g
- In line 7, the new trips are added to history T_f
- In line 8, via $ipdIterate()$ every stay point found in T_f is labeled as the most probable POI in "$pois$". Internally the classifier is re-calibrated and data are re-labeled until convergence is achieved. Then, a new list of "$pois$" is returned together with the updated travel set.
- Lines 5 to 8 are repeated every time a new subset T_n arrives
- In line 10, trip chains are checked in T_f via function $ipdChain()$ to filter out inconsistencies in the travel history before returning it.

The details of implementation are given in further sections and illustrated with sample data. The dataset to be used is first described.

Algorithm 5. Iterative POI detection - main procedure

1: **procedure** IPDALGORITHM(T, n, g, ϵ)
2: $T_0 \leftarrow getTrips(T, n)$
3: $pois \leftarrow ipdInitialize(T_0, \epsilon)$
4: $T_f \leftarrow T_0$
5: **while** $T_n \leftarrow getTrips(T, n)$ **do**
6: $\epsilon \leftarrow g * \epsilon$
7: $T_f \leftarrow merge(T_f, T_n)$
8: $[pois, T_f] \leftarrow ipdIterate(T_f, pois, \epsilon)$
9: **end while**
10: $T_f \leftarrow ipdChain(T_f, pois)$
11: **return** T_f
12: **end procedure**

Algorithm 5 consists of two loops, the "outer" loop is about fetching new trips with $getTrips()$ as they are available. The inner loop implemented inside $ipdIterate()$ is labeling old and new stays then updating the probabilities so that labels are again assigned. At the end, stays converge to the expected labels and probabilities to the actual values.

4.4.1 Dataset description

To evaluate the algorithm's classification performance, a dataset consisting of tracking data with some previously labeled POIs must be used so accuracy performance can be measured. For this, data were collected for a period of 3 months, via a mobile app specifically developed for research purposes from a sample of 638 college students in the city of Cuenca in Ecuador.

The resulting dataset of the campaign contains around 50,000 trips. To illustrate the algorithm, a small sample called dataset A from the travel history of a single user who was asked to manually log her trips in a separate file is to be used for validation purposes.

The stay points of 20 consecutive days in this user's travel history are displayed in Figure 9, different colors state different POI's which can cover areas of different size. One can imagine that using only distance-based data mining can yield different clustering structures depending on the radius size for the agglomerative procedure. The two most relevant clusters that in fact include several smaller-size clusters of stay points are highlighted in the figure, where most labeling mistakes occur.

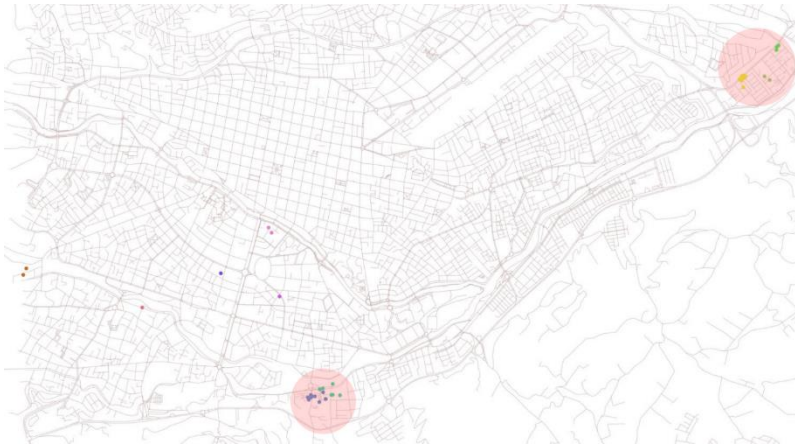

Figure 9. Stay points on a map for a unique user in the studied dataset. Circles denote the two most relevant clusters.

4.4.2 Step 1: Initialization of Iterative POIs detection

The unprocessed data taken from most mobile apps contain information about each stay point's coordinates captured by GPS and a timestamp. That is, the full trajectory has already been segmented into a chain of n stays, usually detected where the user decreased the speed considerably and indicates where a trip ends, and possibly the origin of the next trip via methods described in the literature review section. This first task then reduces to clustering the stays by distance and labeling them with the same POI unique identifier (POI-UI) to represent visits at different moments to a same location.

An initialization process is required to have a preliminary set of POIs which are assumed to be reliable and that will guide the iterative procedure of the algorithm. This trustworthy set should not be difficult to find given that at least home and work locations are expected to appear in most day tours.

4.4.2.1 Initialization: Radius-based clustering

The key aspect in this first step is parsing a first fraction of the travel history, so that the initial set of POIs is discovered, which allows finding some initial user patterns. The semi-supervised approach requires data mining to assign labels to some observations so that a model can be trained, which will need continuous calibration as new data is available. Since relevant locations per user can change over time as well as patterns, using only a fraction (e.g. a couple months) is sufficient and necessary to have a model with the most recent behavior.

Let a data point (stay) i that needs to be clustered have at least the following attributes:

$s_i = (x_i, y_i)$ location of stay point i,

t_i Timestamp of arrival to stay point i,

so that, if S is a subset of all stays in a user's travel history, then they will be parsed in the order they were captured, that is: $t_{i+1} > t_i \quad \forall i \in S$

The criterion to join a new stay point to an existing cluster in this first task, is solely based on spatial data but unlike most density-based methods, the distance to the cluster's centroid is used so that the coverage area remains relatively small. Let $m_k = (\bar{x}_k, \bar{y}_k)$ denote the centroid's location of existing cluster k, then the dissimilarity between this cluster and any stay point i is measured by the Euclidean distance to the cluster's centroid m_k:

$$d_{k,i} = euc.dist(m_k, s_i) = \sqrt{(\bar{x}_k - x_i)^2 + (\bar{y}_k - y_i)^2}$$

in which: $\bar{x}_k = 1/f_k \sum x_i \qquad \bar{y}_k = 1/f_k \sum y_i,$

where f_k is the number of stay points contained in cluster k, and $\{(x_i, y_i), (x_{i+1}, y_{i+1}), .. (x_{f_k}, y_{f_k}),\}$ are the coordinates of every stay point in k. Then, given a maximum search radius r, a stay point j is assigned the closest (most similar) in a set of C existing clusters, that is:

$$c^* = \underset{k \in C}{\operatorname{argmin}}(d_{k,i}) \text{ , subject to: } d_{k,i} \leq r$$

Subsequently, the assigned cluster's centroid is incrementally updated after including any new stay point. If no cluster is reachable from the current stay point i within radius r, a new cluster is created containing solely i. At the end of this process, as illustrated in Figure 10, a first clustering structure is obtained containing a set C of unique POIs identifiers equal to the number of clusters, where each cluster contains at least one element, that is: $f_k \geq 1, \qquad \forall k \in C,$

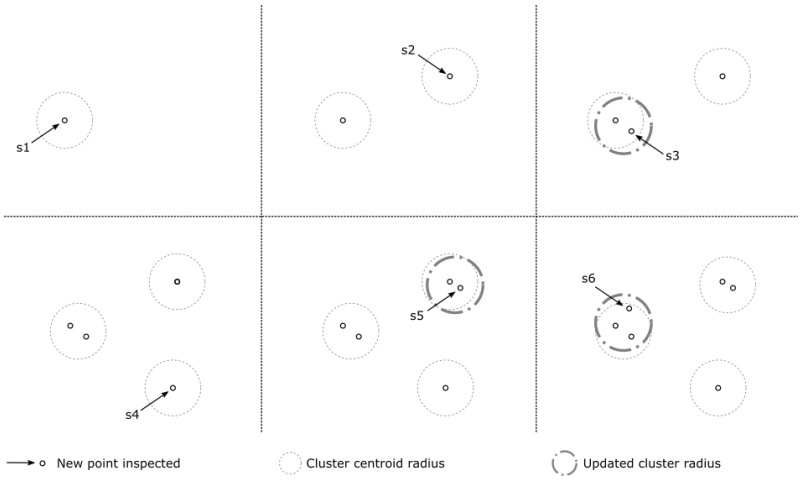

Figure 10. Task 1: Stay points are parsed in the order they appear, then clustered according to their similarity measured as the distance between each new point s_i and an existing cluster's centroid $m_k = (\bar{x}_k, \bar{y}_k)$.

Algorithm 6. Initialization of the Iterative POI detection

```
1: procedure IPDINITIALIZE(T_0, ε)
2:     pois ← ∅
3:     newLabel ← 1
4:     while s ← getNextStay(T_0) do
5:         found ← false
6:         minDist ← ∞
7:         if pois not empty then
8:             while p ← getNextPOI(pois) do
9:                 d ← distance(s, p)
10:                if d ≤ ε and d < minDist then
11:                    minDist ← d
12:                    s.label ← p.label
13:                    found ← true
14:                end if
15:            end while
16:        end if
17:        if found = false then
18:            s.label ← newLabel
19:            p ← createPOI(s)
20:            pois ← pois ∪ p
21:            newLabel ← newLabel + 1
22:        end if
23:    end while
24:    return pois
25: end procedure
```

This process (shown in Algorithm 6) allows all stay points to be labeled with a POI identifier for the location they represent. The rest of trips' stay points in the travel history that were not analyzed (given that new data can be available as it is collected) will be assigned a new POI identifier (one per trip) or merged to the existing clusters in posterior steps. The reason of using the cluster's centroid instead of a point in the border to measure the similarity with a given external stay point, is that search is constrained to minor areas producing small clusters that can be merged later if patterns are similar. Some partial results of the initialization process when using the first 60% of trips in dataset A are presented in Figure 11, where labels show the cluster ID to which the stay points belong.

Figure 11. Results of initialization process in stay points of dataset A. Numbers denote each cluster's labels.

4.4.2.2 Origin-based conditional probability

Let y_k be the number of all trips in a travel history to destination u_k; similarly let $y_{m,k}$ be the number of trips found from origin u_m to u_k, then the conditional probability of POI u_m given destination $u_k$is:

$$Pr(p_{i-1} = u_m | \, p_i = u_k) = y_{m,k}/y_k$$

So that, after a large amount of observations, the relative frequency of every pair of consecutive POI identifiers $p_i \rightarrow p_{i+1}$ in a travel history, can help estimating the probability of having seen a certain POI before the current destination identified as u_k.

4.4.2.3 Arrival time-based conditional probability

The list of registered arrival times to a certain POI can be retrieved from the timestamps of the matching stay points with the same POI identifier, that is: $A_{(k)} = \{ t_i \mid p_i = u_k \}$, where t_i is the timestamp of stay i, and p_i its POI identifier. A probability density function $f(t)$ can be approximated by smoothing the time stamps' frequencies distribution via kernel density estimation (69). Then the conditional probability of a certain arrival time given a subset of trips with destination u_k is:

$$\Pr\,(t = t_i | p_i = u_k) = \; f(t) = \frac{1}{nh} \sum_{j=1}^{n} K\left(\frac{t_i - t_j}{h}\right)$$

where $n = \left|A_{(k)}\right|$, that is the number of registered arrivals to destination u_k; K a kernel function (generally a Gaussian distribution) and h a bandwidth parameter, which the larger it is, the better the curve fits the frequencies of the distribution.

4.4.2.4 Distance-based conditional probability

At last, the likelihood of a given POI u_k should also decline as the distance to location s_i of stay point i increases, which can be estimated by a so-called distance decay function, like for instance the following expression (70):

$$z_{k,i} = 1/\left(d_{k,i}\right)^2$$

Then, the conditional probability of a stay-point to belong to a POI u_k at certain distance is proportional to this measure and can be approximated by:

$$Pr(s = s_i \mid p_i = u_k) = \frac{z_{k,i}}{\sum_{j \in C} z_{j,i}}$$

4.4.3 Step 3: Update stay point labels to best matching POI

After probabilities have been computed, the analyzed subset is extended by including a new fraction of the trips and then new stay points can be labeled, additionally also labels of the "old" ones can be updated. To decide the best matching label (POI), the joint probability is calculated by assuming events are exclusive, that is:

$$\Pr(u_k, u_m, t_i, s_i) =$$

$$Pr(\, p_{i-1} = u_m | p_i = u_k \,) \times \, Pr(t = t_i | p_i = u_k) \, \times Pr(s = s_i | p_i = u_k) \times \, Pr(p_i = u_k)$$

with $Pr(p_i = u_k) = r_k$, in other words, the prior probability equals the relative frequency of the POI.

Then, the best matching POI for stay point i, with arrival time t_i and location s_i at iteration j is:

$$p_i{}^{(j)} = \underset{k \in C}{\operatorname{argmax}}(\Pr(u_k, u_m, t_i, s_i))$$

where u_m, is the identifier of the previous stay point in the chain.

In this way, the POI sequence patterns can help finding "mistakes" in the travel chains by updating each stay point label to the best matching (most expected) POI. After labels have been updated, probabilities can be re-calculated and labels re-updated, this iterative process continues until no more trips are left in history and convergence is observed.

The final travel history must then match the actual sequence of POIs visited by the user, which possibly contains a reduced list of unique POIs compared to the initial list after the first task. The final clustering structure of the complete procedure implemented in Algorithm 7 is presented in Figure 12.

Figure 12. Clustering structure of IPD (complete procedure)

In contrast to other algorithms as DBSCAN, IPD can classify stay points by using multidimensional patterns instead of only using distance or temporal variables. For instance, the structure with DBSCAN with $\varepsilon = 180$ is shown in Figure 13.

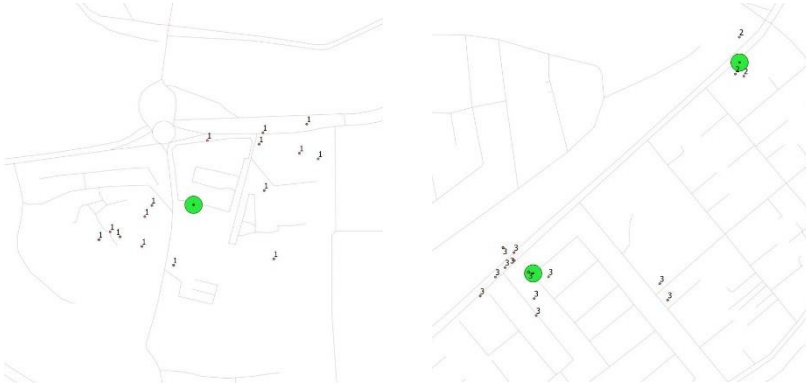

Figure 13. Clustering structure of DBSCAN when using a large radius, stays are clustered into three POI's.

The first snapshot (left figure) has merged all stay points; if a smaller $\varepsilon = 100$ is used, the structure shown in Figure 14 is obtained, where many clusters were produced splitting clusters into smaller parts in both figures. It must be noticed that only five POIs actually exist in both snapshots, but since the user used different parking slots they seem to be more as the covered area is larger.

For dataset A, the resulting POIs configurations of each method when snapping cluster centroids to the closest actual POI on the map of each method are displayed in Table 6. In the following tables, only the DBSCAN method with small ε is used since it provided better results than the former.

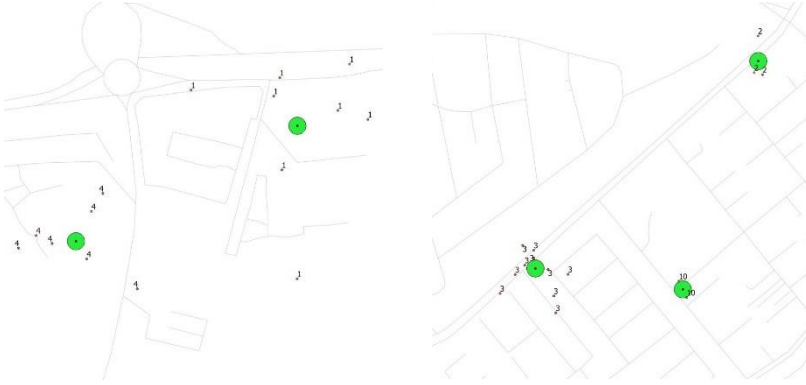

Figure 14. Clustering structure of DBSCAN when using a smaller radius, stays are clustered into five POI's.

Table 6. Comparing size of detected clusters with each method snapped to the closest actual POI.

ACTUAL POIs			IPD			DBSCAN	
LABEL	**SIZE**		**LABEL**	**SIZE**		**LABEL**	**SIZE**
A	14	←	1	15	←	1	15
B	2	←	2	2	←	2	2
C	3	←	3	3	←	3	3
E	2	←	4	2	←	4	2
F	1	←	5	1	←	5	1
G	6	←	6	6	←	6	8
H	14	←	7	13	←	7	11
I	1	←	8	1	←	8	1
K	1	←	9	1	←	9	1
L	2	←	10	2	←	10	2

Table 7. Confusion table that compares the classification performance of (left) DBSCAN and (right) IPD. Left margin labels are actual values and headers are predictions.

		DBSCAN										IPD									
		A	**B**	**C**	**E**	**F**	**G**	**H**	**I**	**K**	**L**		**A**	**B**	**C**	**E**	**F**	**G**	**H**	**I**	**K**	**L**
REFERENCE	**A**	14	0	0	0	0	0	0	0	0	0	**A**	14	0	0	0	0	0	0	0	0	0
	B	0	2	0	0	0	0	0	0	0	0	**B**	0	2	0	0	0	0	0	0	0	0
	C	0	0	3	0	0	0	0	0	0	0	**C**	0	0	3	0	0	0	0	0	0	0
	E	0	0	0	2	0	0	0	0	0	0	**E**	0	0	0	2	0	0	0	0	0	0
	F	0	0	0	0	1	0	0	0	0	0	**F**	0	0	0	0	1	0	0	0	0	0
	G	0	0	0	0	0	6	0	0	0	0	**G**	0	0	0	0	0	6	0	0	0	0
	H	1	0	0	0	0	2	11	0	0	0	**H**	1	0	0	0	0	0	13	0	0	0
	I	0	0	0	0	0	0	0	1	0	0	**I**	0	0	0	0	0	0	0	1	0	0
	K	0	0	0	0	0	0	0	0	1	0	**K**	0	0	0	0	0	0	0	0	1	0
	L	0	0	0	0	0	0	0	0	0	2	**L**	0	0	0	0	0	0	0	0	0	2
	TOTAL	15	2	3	2	1	8	11	1	1	2		15	2	3	2	1	6	13	1	1	2

One of the difficulties of tacking this task only as a clustering problem is that optimization is oriented to reducing the intra-cluster variance; however for instance detecting incorrectly two small clusters instead of one actual large cluster of stay points will yield a lower variance on average, and clusters of one single point will even have zero variance. Moreover, large clusters will naturally have large variances as POIs can have arbitrary shapes and sizes, so that variances do not necessarily reflect the real performance of a certain algorithm.

When seeing the plots, it seems that the DBSCAN with small ε produces a more natural clustering, however given inaccuracy in GPS measures, finding some points closer to the centroid of an incorrect cluster is frequent. As this is a classification problem, a better measure of performance is achieved by comparing the predicted against the reference POI labels.

The average recall (true positive rate) for DBSCAN was 90.95%, while for IPD was 97.14%, where the recall $TPR(k)$ for a specific POI identifier u_k can be computed by:

$$TPR(k) = \frac{|\{i \in S \mid p_i = u_k \wedge \widehat{p_i} = u_k\}|}{|\{i \in S \mid \widehat{p_i} = u_k\}|}$$

where, S in the set of all stay points in a user's history, p_i is the POI identifier of stay point i assigned by IPD and $\widehat{p_i}$ is the actual POI the point belongs to. That is, this measure provides the proportion of stay points correctly labeled as u_k which is expected to be close to 100% for a good performing classifier. Similarly, the false positive rate (expected to be small) can be computed by:

$$FPR(k) = \frac{|\{i \in S \mid p_i = u_k \wedge \widehat{p_i} \neq u_k\}|}{|\{i \in S \mid \widehat{p_i} \neq u_k\}|}$$

The average false positive rate for DBSCAN was 1.90%, while for IPD was 1.06%; in this small sample IPD is clearly more accurate. As a final remark about this step, at least two more conditional probabilities considering the attributes: day of the week and travel mode could be included in the previous formula.

4.4.4 Computation time

Since the proposed algorithm has an iterative nature, its complexity is affected by the dataset size (number of analyzed stay points as well as the number of features included in the model. In order to evaluate the computational complexity, the algorithm was run on a set of increasing-sizes portions of the dataset previously used. At last, the computational run time was fitted using models such as linear $O(n)$, quadratic $O(n^2)$, logarithmic $O(\log n)$ among others. The results are presented in Figure 15, where it suggest a quadratic to cubic form as the closest complexity type when models are compared via a Mean Squared Error measure. This fact states that computation efforts increase rapidly as travel histories become larger.

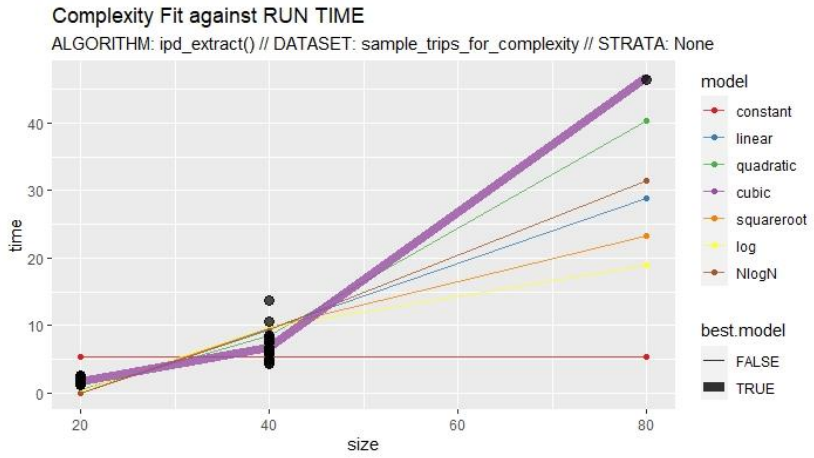

Figure 15. Computational time evaluation of IPD.

4.5 STEP 4: RETRIEVING CONSISTENT TRIPS CHAIN

From the travel chain in the previous step, final patterns consisting of subsets of POI sequences of different lengths and their characteristics can be obtained. This is an optional step; however, it makes the algorithm useful for most applications in real life.

Let H be a travel history written as the full sequence of POI identifiers $\{p_1, p_2, .., p_{n-1}, p_n\}$ for the visited stay points by a certain user; a simplified lagged matrix of sequences of length 2 is:

$$\begin{bmatrix} p_1 & p_2 \\ p_2 & p_3 \\ \vdots & \vdots \\ p_{n-2} & p_{n-1} \\ p_{n-1} & p_n \end{bmatrix}$$

For sequences of any length $L \in \mathbb{Z}^+$, where $1 < L < n$, the following matrix is obtained.

$$\begin{bmatrix} p_1 & \cdots & p_{L-1} & p_L \\ p_2 & \cdots & p_L & p_{L+1} \\ \vdots & \ddots & \vdots & \vdots \\ p_{n-L+1} & \cdots & p_{n-L} & p_n \end{bmatrix}$$

The matrix can then be rewritten as a set of POI sequences of length L:

$$H_{(L)} = \{< p_1, \dots, p_L >, < p_2, \dots, p_{L+1} >, \dots, < p_{n-L+1}, \dots, p_n >\}$$

Let X be a POI sequence of length L, the absolute frequency of X in the travel history is:

$$freq(X) = \left|\{h \in H_{(L)} \mid h = X\}\right|$$

and the relative frequency or support of X is:

$$supp(X) = \frac{\left|\{h \in H_{(L)} \mid h = X\}\right|}{\left|H_{(L)}\right|}$$

$$\hat{H} = \{X \in H_{(L)} \mid supp(X) > \varphi\}, \text{ with } 1 < L < n$$

Then, applying a threshold $\varphi > 0$, the POI sequence patterns of any length are retrieved.

Algorithm 7. Updating stay point labels according to maximum likelihood.

```
procedure IPDITERATE(T, pois, ε)
    T_f ← ∅
    P_max ← 0
    while s ← getNextStay(T) do
        p* ← s.label
        while p ← getNextPOI(pois) do
            d ← distance(s, p)
            if d ≤ ε then
                p_attracion ← sum(OD_matrix[, p.label])/sizeOf(T)
                decay ← 1/d²
                p_decay ← decay/totalDecay
                f ← gaussian_density(p.arrival_times)
                p_arrival ← f(s.arrival_time)
                if prev_s then
                    od_trips ← OD_matrix[prev_s.label, p.label]
                    p_OD ← od_trips/sum(OD_matrix[prev_s.label, ])
                end if
                p_joint ← p_attracion * p_decay * p_arrival * p_OD
                if p_joint > P_max then
                    P_max ← p_joint
                    p* ← p.label
                end if
            end if
        end while
        if p* ≠ s.label then
            s.label ← p*
        end if
        prev_s ← s
    end while
    return [pois, T]
end procedure
```

The generation of a correct and consistent travel chain is critical for modeling transitions of mobility behavior. This algorithm will be proven to converge to the actual chain compared to solely density-based techniques in the next section. Several applications take advantage of travel chains since this allows location prediction to provide different services, a recent review on techniques for location prediction based on trajectory data can be found in (67).

For dataset A, the sequence patterns of length two (origin-destination patterns) when using a minimum support of two trips per pattern are displayed in Table 8.

Table 8. Origin-destination patterns for dataset A, when applying a minimum frequency of 2 trips.

Origin	destination	frequency	relative frequency
H	G	5	0.10869565
H	A	3	0.06521739
B	A	2	0.04347826
G	G	3	0.06521739
G	C	3	0.06521739
G	A	3	0.06521739
C	A	3	0.06521739
A	H	8	0.17391304
A	B	2	0.04347826
A	L	2	0.04347826
L	A	2	0.04347826

Applying this minimum support, filters out some POIs as seen in the last table; allowing the actual personal points of interest to be identified instead of listing all unique visited destinations as so far. Nevertheless, it depends on the application of interest since the real travel history can only be retrieved after all locations have been used (even when they were visited just once). The same procedures illustrated in this section are to be applied to a larger dataset with trips from several users. It can be imagined that numerous apps can take advantage of the approach described in this paper to develop smart features.

4.6 FURTHER EXPERIMENTS AND DISCUSSION

In this section, the algorithm is used on a second dataset called B, with an entire travel history from life-logging data collected for an anonymous user during the campaign. This new data set covers an area much larger than data set A and contains 454 trips made in 3 months, after displacements of less than 200 meters in length were filtered out. A snapshot of the area containing most of these non-processed trips is shown in Figure 16, together with the convergence of the IPD algorithm that took to complete . It shows how the number of POIs is reduced after each iteration of the re-labeling process. In real life situations, permanent tracking is provided so that chunks of data are repeatedly collected; in particular for this example, 60% of data was used in the initialization process and all the remaining 40% in the next steps, attaining convergence in 24 iterations in the inner loop. After the initialization stage 201 early centroids were detected and reduced at the end of the relabeling process to 75 unique destinations.

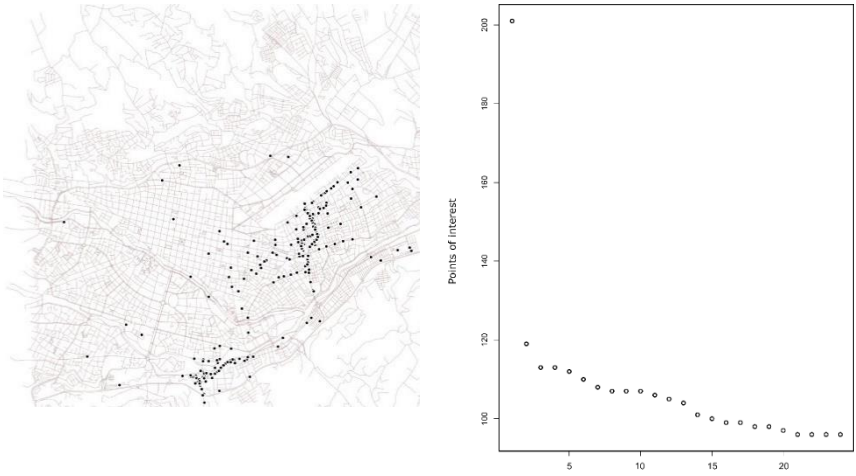

Figure 16. (left) Stay points in dataset B before using IPD and (right) convergence after using the entire dataset when relabeling.

At last, a personal point of interest must have a minimum support, that is a minimum number of visits (application-dependent), the higher the required support the lower the quantity of POIs. In Figure 17 this fact can be observed when applying different thresholds, then in the same figure (right) a fraction of the corresponding POIs on the map for a support of 15 visits.

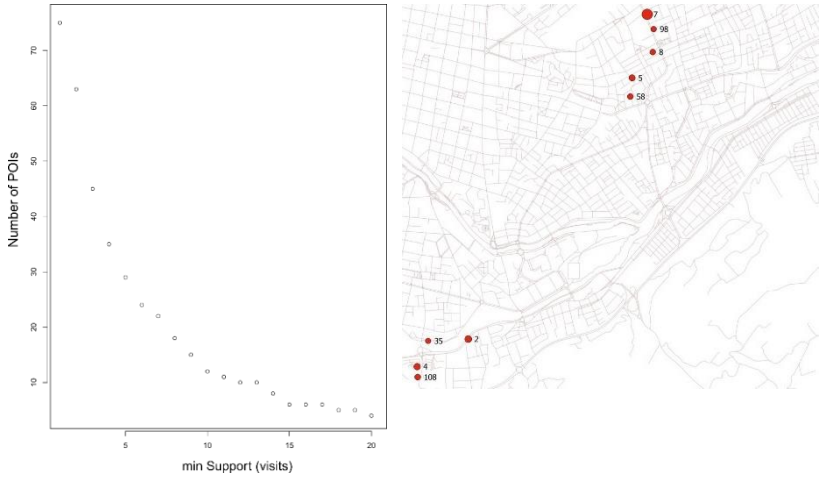

Figure 17. (left) Number of Personal Points of interest in travel history of dataset B for different support values and (right) the corresponding POIs for a support of 15 visits.

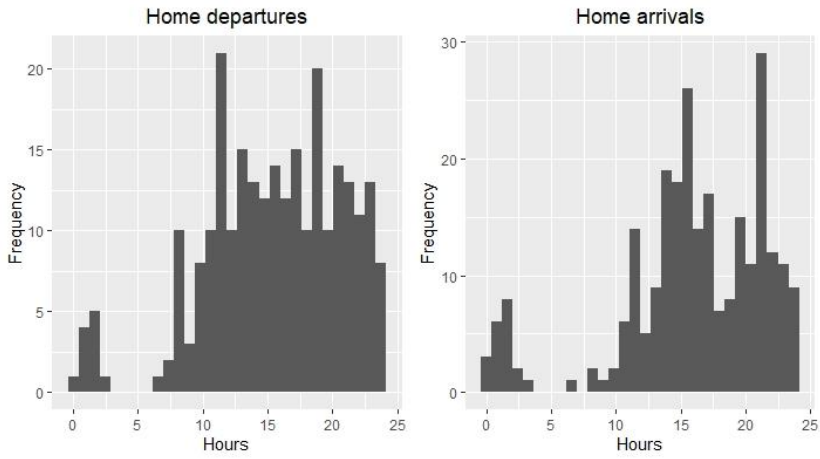

Figure 18. Temporal patterns of POI identified as "home".

The temporal patterns of the most visited POI (cluster 7) that is the assumed home location, are shown in Figure 18, where this user (student) mostly leaves this location around noon or at 7pm, and then returns around 3pm or 9pm. The origin-destination (OD) patterns in step 4, for sequence patterns of length $L = 2$ with a minimum support of 5% are displayed in Table 9.

Table 9. OD patterns for home location (cluster 7)

Origin	Destination	Frequency	Proportion
4	7	39	8.59%
42	7	30	6.61%
58	7	28	6.17%
7	14	29	6.39%
7	2	31	6.83%
7	42	40	8.81%
7	5	25	5.51%

4.7 CONCLUSIONS AND FURTHER WORK

Because of the mobility patterns found in early stages of travel histories, detection of personal points-of-interest (locations recurrently visited) can be greatly improved by adding more attributes in the clustering procedure besides the typical spatiotemporal components. If some mobility patterns can be observed in a history, such as: arrival times or the frequent POI's transitions, the classification of stay points into the best fitting POI label can be decided.

Compared to traditional techniques for POI detection, the iterative algorithm presented in this paper uses the knowledge regularly learned from a travel history to improve detection of the upcoming trip endpoints (origins or destinations). It can be expected that accuracy will increase as more information is collected, allowing a specific classification model to be generated per

user history, which is robust enough to adapt to upcoming changes in the user's mobility behavior, as it is automatically calibrated (and probabilities are updated) as new knowledge is learned. Moreover, calibration require old trips in history to be disregarded so that only recent behavior is included in the model; this can be achieved by training the classifier at a fixed time interval basis (e.g. daily) using only the n previous records in history (e.g. only the n trips made in the last months).

Some issues to be mentioned that could affect the performance of the algorithm can be identified. The results of the initial clustering carried out by the first task depend on the number of initial POIs. It must be large enough to identify some of the user's mobility patterns, but small enough so that stay points in the border of two or more clusters are not merged but instead their labels decided on further stages. Also, the computation time is always longer than most of DBSCAN and K-means variants because clustering and re-labeling is repeated until achieving convergence; the complexity in run time fits a quadratic shape as shown earlier in Figure 15. It can be expected that in larger travel histories, the number of required iterations and computation time will rapidly increase, so that again mobile devices should collect and consume data, but the presented algorithm should run on the server side possibly one time a day to process new daily trip chains.

Two main improvements can be suggested for future work. First, including new trip characteristics to the classification inputs such as day of the week or travel mode, allowing new correlations to be discovered with the visited location. At last, a general framework to develop specialized mobility apps can be defined making it possible to trigger location- or pattern-based services.

4.8 ACKNOWLEDGEMENTS

This research was supported by the Secretary of Science and Technology of the Ecuadorian government.

5 PRODUCING MULTI-DAY SYNTHETIC POPULATIONS FOR SHARED-MOBILITY SIMULATIONS FROM LIFELOGGING DATASETS.

Ivan Mendoza [a, b, *], Chris M. J. Tampere [a]

[a] KU Leuven, L-Mob, Leuven Mobility Research Center, CIB, Leuven, Belgium

[b] Faculty of Science and Technology, Universidad del Azuay, Cuenca, Ecuador

Submitted to Journal of Transport Geography

5.1 ABSTRACT

Evaluating ridesharing potential is a trend in current research efforts because it provides additional mobility alternatives without extra vehicles on the road. Nevertheless, in most studied scenarios, the demand produced by surveys and demographic information does not include multi-day characteristics of a trip such as its frequency. Yet, this is important for estimating the supply of rides, as the recurrence/regularity of a trip may affect the likelihood for a driver taking the effort of registering the trip as being available for sharing. Likewise, if automated apps are used to recognize patterns in one's trips and pro-actively offer them for sharing, the successful anticipation of such apps may again depend on the regularity of the trip. Multi-day data are however complex to produce. In this paper, a data-driven procedure is proposed to generate an enriched synthetic demand for more realistic assessments. This can be achieved by combining standard single-day datasets and travel behavior patterns, which can be extracted after mining lifelogging data collected by most existing mobile apps. The enriched datasets, produced after transferring information from one dataset to a receptor via statistical matching techniques, will constrain matching trips by multi-day characteristics. This approach allows simulations of complex scenarios, enhancing the evaluation of shared mobility systems for planning better strategies.

***Keywords**:* Ridesharing, lifelogging data, datamining, travel behavior, hot-deck imputation.

Credit author statement

Iván Mendoza: Conceptualization, methodology, software, writing original draft preparation, and visualization / data presentation.

Chris Tampere: Supervision, validation, Writing- Reviewing and Editing.

5.2 INTRODUCTION

The widespread deployment of current mobile technology has increased the interest in shared mobility systems. One such systems is dynamic or real-time ridesharing. We discuss here the form of ridesharing, where citizens who make a car trip decide to offer the available free seats

* Corresponding author, email address: imendoza@uazuay.edu.ec

in their vehicle to potential passengers, with origins and/or destinations on (or close to) their route. The supply of rides in dynamic ridesharing thus depends on the car trips that are being made, and the probability that the driver (or an automated app on his behalf) decides to offer the ride for sharing. As dynamic ride sharing creates mobility supply without adding additional vehicles on the road, it has attracted researchers and policy makers as a potentially sustainable form of transport. Many studies have evaluated its potential, as a stand-alone system, or in combination with public transport (e.g. as a feeder service)(71,72).

Most studied scenarios in microscopic simulations make use of demand produced by surveys and demographic information, which does not include multi-day characteristics. Yet, such characteristics may be important for estimating the supply of rides, as the recurrence/regularity of a trip may affect the likelihood that a driver takes the effort of registering the trip as being available for sharing. For instance, drivers may have the habit of sharing their regular commute trips, but not irregular ones like doing groceries. Or on the contrary, they may only share the less recurring trips for which they already consult their phones for traffic info or navigation. Traffic info and navigation apps may even be adapted to encourage sharing these trips with a click of a button. Likewise, if automated apps would be used to recognize patterns in one's trips and pro-actively offer them for sharing, the successful anticipation by such apps may again depend on the regularity of the trip. These are just examples showing that multi-day travel demand patterns may be relevant for creating alternative potential ridesharing scenarios.

Because producing this type of multi-day demand can be a complex task, this paper provides a rather direct procedure to generate enriched synthetic demand with multi-day characteristics for more diverse, realistic assessments. This is achieved by combining synthetic populations containing typical average-day datasets, with mobility patterns extracted from lifelogging data which can easily be collected by existing mobile apps.

The information extracted from these big data, consisting of several heterogeneous sources, constantly collected by lifelogging apps; allows patterns of travel behavior to be identified so that multi-day characteristics produce augmented datasets. This information can further be exported to a synthetic population, via statistical matching techniques (73–75), to generate a dataset suitable for multi-day simulations. In the context of ridesharing applications, these resulting datasets let matching trips to be constrained by the multi-day features, allowing novel complex scenarios not included in most previous research efforts to be evaluated by simulators.

The contribution of this paper yields on the generation of this multi-day synthetic demand, from information already available on most lifelogging apps and public online data sources, providing a framework that combines different techniques for general purposes. Examples of multi-day characteristics identified in the mobility patterns extracted from the lifelogging datasets are: a weekday rate, a trip's average daily occurrence, the trip's frequency relative to other trips in the travel history (called regularity in this document), allowing different ridesharing scenarios such as supply consisting of regular weekday trips in certain time window and location (8).

This document is organized as follows:

- A literature section that aims at these two aspects: generation of synthetic demand from lifelogging datasets and evaluation of shared-mobility systems containing multi-day characteristics.
- The proposed methodology is explained in depth, which provides a framework combining different techniques to produce the multi-day synthetic population.

- As a proof of concept, a few uncommon scenarios that can be evaluated via the enriched dataset are presented.
- A discussion of the results and future research directions.

5.3 LITERATURE REVIEW

The following literature is studied considering two aspects. Firstly, we review the generation of activity-based demand for traffic simulations from data sources of different forms. We state how this paper's approach improves the detection of multi-day characteristics by using lifelogging data. Secondly, we review the evaluation of shared-mobility systems (mainly ridesharing), paying attention to the applied constraints. We state how the generated demand in the previous step allows assessing ridesharing in new ways. Matching optimization is not discussed in this document, however for recent review on algorithms for shared mobility scenarios can be found in (76).

5.3.1 Generation of synthetic demand from lifelogging datasets

These contributions mainly focus on creating activity-based demand for traffic simulations from data sources of different forms (a.k.a. population synthesis). In this context, one of the most relevant and earliest efforts came from (77) by using iterative proportional fitting procedures (IPFP) to solve the population synthesis problem. The method completes a contingency table for the joint probabilities of several variables, where the fixed marginals are taken from some census summary tables. Then the probabilities are transformed into absolute numbers, and lastly, individual observations from a separate disaggregate sample (e.g. surveys) are drawn to create the synthetic population.

In (78) the authors updated the procedure in (77) to avoid zero-cell values in the contingency tables, then combined two public datasets. One data set is the "Public Use Microdata Sample" (PUMS), which contains disaggregate records about individual people and housing units; the other set are the U.S. Census Records from where the joint distributions of most sociodemographic variables were obtained.

In recent years, tools such as URBANSIM (79) have been used to model urban systems, trusting on its detailed agent-level interactions to avoid the assumed demand homogeneity of previous models. This way, demand is produced in a disaggregate form where each agent's destination and travel mode for a single day is decided based on the demographic data. One of the first attempts to review the state of the art of population synthesis for the recent type of agent-based microsimulations is presented in (80), where other tools such as PopGen (81) are mentioned. All of them follow a similar approach, which can be summarized in two steps: a fitting stage where the contingency table is created by aggregating statistics from a population sample, then an allocation stage where individual agents are sampled from the previous proportions by adding some heterogeneity in the process. More recently in (82), the authors developed a disaggregate tour-based mobility demand model, including a departure time choice model to detect the effects of increasing congestion; they called the synthesizer PopSim. In the last years, the Flemish government developed a tour-based synthetic population for the region of Flanders, Belgium that can be used in several simulators, called the strategic persons models of Flanders v4.1.0 (83).

The previously mentioned contributions are mostly based on traditional data collection methods; however, the uprising of big data allows demand datasets to be produced by lifelogging-related procedures. Some relevant literature is mentioned. For instance, in (84), authors studied human mobility patterns by analyzing mobile phone data to conclude that

despite the diversity of each user's travel history, humans follow simple reproducible patterns that could be used for urban planning and agent-based models. Later in (85), they used similar data to construct origin-destination matrices, allowing the aggregate demand to be automatically generated instead of using surveys. More recently in (86), they used data mining techniques to define a framework that can be used to evaluate patterns in a region of interest, based on the extrapolation of mobility patterns found in (Call Detail Records) CDR data and then tested for the Singapore metropolitan area. Lastly in (87), authors used a similar approach to create the OD matrices for Senegal after aggregating the mobile phone data into inter-district trips and then extrapolating based on census data.

Even though population synthesizers in the papers discussed so far can generate heterogeneous demand for multiple days, the approach described in this paper enhances those efforts by adding unique multi-day characteristics per trip, such as the trip's frequency in a specific day of the week, which can only be inferred after finding a user's mobility patterns. More about demand modelling for transport research via big data can be found in (88) and (89).

5.3.2 Evaluation of shared-mobility systems containing multi-day data

Some research efforts related to evaluation of shared-mobility systems (mainly ridesharing) devote special attention to the use of disaggregate demand data and multi-day characteristics from mobility patterns. For instance, in (39), authors assessed the potential of ridesharing within de cities by combining CDR with social networks data from Twitter and Foursquare, in order to match users with similar spatiotemporal patterns. They added a constraint called social distance, quantifying the number of friends that users have in common. They hypothesize that the larger the distance, the smaller the chance to share a ride. In (5), authors used CDR data to infer an average daily origin-destination (OD) matrix. Then, by assuming an adoption rate, they proposed ridesharing matches to finally assess the impact on congestion. A different point of view can be found in (90), (7) and (8), where big data was used to suggest ridesharing hotspots based on the regularity of mobility patterns: the stronger the pattern, the more relevant the hotspot will be for taking passengers to similar destinations.

A different use of collected disaggregate data to evaluate ridesharing is taken in (91) and (92). There, authors proposed a method to improve ride-matching rates, by increasing the destination choice set with new alternative destinations. The set consisted of locations for certain types of activity, taken from public databases of points-of-interest. Later in (93), they extended the matching process by adding social-network links for joining people. They called this approach a collaborative activity-based ridesharing, where detour tolerances and willingness to share rides with friends are assumed.

These approaches use mobile phone records to improve or evaluate ridesharing scenarios, using mobility patterns when available, so that users are matched across typical variables such as spatio-temporal or social distances. The approach of the current paper enhances the previous research efforts, by learning some multi-day characteristics from a separate sample and then transferring this knowledge to a synthetic demand to be able to extrapolate to a population level. Finally, these characteristics containing information about the frequency and regularity of each trip are used to match the trips. This approach allows evaluating ridesharing in complex scenarios, where the likelihood of trips being offered for sharing may depend in one way or another on the multi-day attributes.

5.4 METHODOLOGY

The main idea for evaluating ridesharing multi-day scenarios, is to match trips from a dataset that contains the extra characteristics and using this information to add new constraints. To obtain this type of dataset, two or more heterogeneous sources must be combined, assuming that they were sampled from the same population and that they are statistically similar with respect to the shared variables. The entire procedure is introduced in Figure 19. At least two datasets are required, a first "D" that has multi-day characteristics but does not have enough observations to draw faithful conclusions for the population it represents; and a second "R" that is large and/or representative enough to characterize the study population and faithfully represents an average weekday of the case study, nevertheless it does not have multi-day characteristics. The contribution of this paper is to define a formal procedure to produce a third enriched dataset "$R+$" by merging D and R; this demand will contain both desirable properties, that is, multi-day characteristics and enough observations. Each step is described in the following sections. Since R is the largest dataset, it is suitable for simulating ridesharing experiments and from this point on this will be called the "Receptor". Similarly, dataset D containing the desired multi-day characteristics coming from lifelogging data collected by mobile devices, will be called in this context the "Donor". The class of methods used to transfer information from D to R, is known in literature as statistical matching (74).

Let X be the list of characteristics (variables) describing each observation (trip) in a travel history, including but not limited to travel time, travel distance, travel mode or trip frequency. Also, let X_M be the set of characteristics that can be found in both datasets, Y those that are unique to R and Z those unique to D. The observations on each dataset we have the following characteristics:

- $X_R = X_M \cup Y$, trip characteristics for observations in dataset R
- $X_D = X_M \cup Z$, trip characteristics for observations in dataset D

Where $X = X_M \cup Y \cup Z$ are all the possible characteristics for a trip if both datasets were merged so that the intersection $X_M = X_R \cap X_D$ the matching (common) variables. This new enriched dataset will be called R+.

In general terms, the steps illustrated in Figure 19 involve the following tasks:

- Datasets D and R are collected via lifelogging apps and other sources. Both having some matching variables X_M in common.
- Even though some variables in both datasets could be matched to populate X_M, by only performing unit conversions or simple operations, some others are initially "hidden" and can only be discovered after data mining procedures. Therefore, an extended set $X_M +$ of matching variables must be found to increase chances of finding similarities in the datasets.
- The multi-day information in D to be transferred is mined from the lifelogging data via discovery of mobility patterns. At this point the original D and R have been extended producing D' and R'.
- In order to transfer the multi-day information to R', regression models are learned from information contained in the matching variables in D'.
- Alternatively, to accelerate transferability, trips in D' are first classified into donor classes so that specific models per class can be produced.
- Receptor dataset is enriched by predicting the multi-day characteristics via models trained in the previous step producing $R+$.

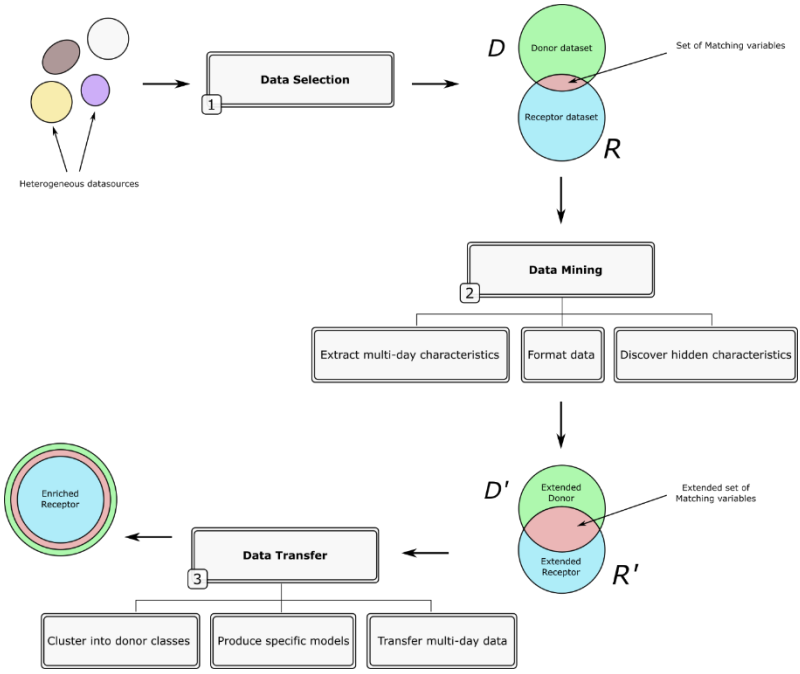

Figure 19. Combining datasets to get an enriched multi-day dataset

5.4.1 Datasets Description

For this research, two sets of sample data are available. D was obtained by an app specially developed for a campaign called Transmob (94) to track multi-modal trips with around 85,000 observations in the region of Antwerp. In this dataset, trips are automatically registered; moreover, personal information is hidden. Furthermore, no socio-demographic data is provided. Similarly, R is a synthetic population provided by the Flemish Government (83) and contains around 1 million single-day home-based tours in different departure periods, following a procedure similar to (82). The latter also includes a large number of socio-demographic and economic variables such as age, household size, income, among others. Both datasets cover the study area of Antwerp in Belgium so that they belong to the same population.

Since R lacks multi-day information, as mentioned earlier, the task is to import these characteristics from D through methods described in Figure 19 and elaborated in the following sections. The most relevant attributes found in each dataset are described below, after raw data was aggregated into trip records by some trip segmentation algorithm such as (21) and (56).

Table 10. Trip characteristics in both datasets.

	Name	Description	Dataset
o_i	Origin location of trip T_i	Coordinates (x, y) of location where trip starts	D, R
e_i	Destination location of trip T_i	Coordinates (x, y) of location where trip ends	D, R
a_i	Arrival time to destination of trip T_i	Time of the day in minutes $a_i \in [0, 1440)$	D
s_i	Starting time from origin of trip T_i	Time of the day in minutes $d_i \in [0, 1440)$	D, R
m_i	Travel mode of trip T_i	$m_i \in \{\text{motorized, bicycle, walking}\}$	D
u_i	User unique ID	A user ID of the person that is being tracked.	D
ad_i	Arrival date of trip T_i	day, month and year	D
δ_i	Purpose of trip T_i	$\delta_i \in \{\text{work, education, recreational, shopping}\}$	R

That is, each record in the dataset is a unimodal trip (single trip leg). In our donor and receptor datasets, only the origin, destination, and departure time are common. However, it is preferable to have multiple matching variables to increase the chances of finding good, specific "twins" between datasets. We will do so by inferring hidden characteristics as a key step, which is elaborated in the further section.

5.4.2 Discovering hidden characteristics

This section explores some options to increase the number of characteristics $x \in X_M$ that are common to both datasets (see Figure 20). A first relevant characteristic is inferred by paying attention to each trip's direction, so that trips can be classified into inbound, tangential, and outbound with respect to some region's centroid c.

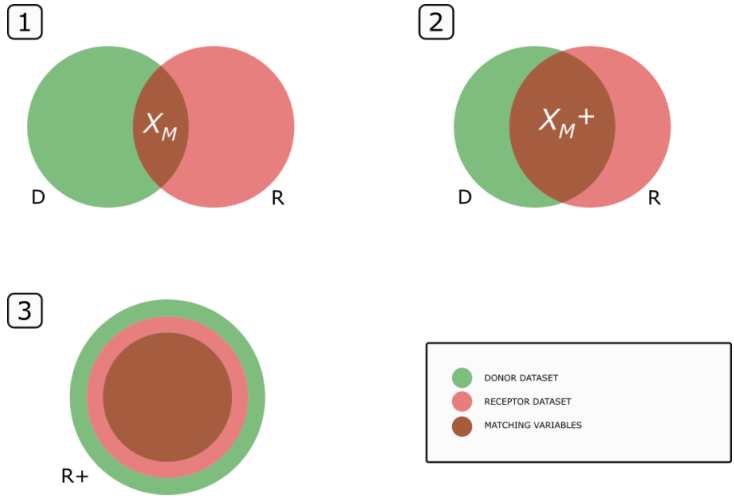

Figure 20. Extending matching variables to increase chances of finding similarities between both datasets.

5.4.2.1 Radial-tangential movements

For this, c is located by clustering all destinations' coordinates for trips on weekdays and morning peak hours, so that city's downtown area is assumed to occupy the location of the largest cluster, attracting the majority of commuting trips in the morning. The clustering algorithm for this task must be agglomerative using a dissimilarity measure like this one:

$$d_{k,i} = \min_{p_j \in K} \{ d(p_j, p_i) \}$$

Where $d_{K,i}$ is the dissimilarity between cluster K and some external point p_i with coordinates $[p_i.x\ , p_i.y]$, computed as the distance from p_i to the closest point p_j already contained in K. Likewise, function $d(p_j, p_i)$ provides the euclidean distance between two points after conversion of the spherical coordinates. The most attractive (largest) cluster of the resulting clustering structure C, where c will reside is identified by:

$$C^* = \underset{C_k \in C}{\operatorname{argmax}} \{ |C_k| \}$$

Lastly, the assumed region's centroid denoted by $c = [c_x\ ,\ c_y]$ is the centroid of this cluster, where:

$$c_x = \frac{\sum_{p_j \in C^*} p_j.x}{|C^*|} \qquad c_y = \frac{\sum_{p_j \in C^*} p_j.y}{|C^*|}$$

For instance in Figure 21-1, for a sample drawn from dataset D that mostly contains trips around the Antwerp region, the relative size of each found cluster is given in Table 11.

Table 11. Size of clusters of trip endpoints.

Cluster ID	Size (%)	Cluster ID	Size (%)
1	7.7%	5	0.3%
2	1.3%	6	1%
3	87.8%	7	1.6%
4	0.3%		

The densest area includes clusters 1 to 3 (see Figure 21-2), where the largest cluster (with ID 3) is used to extract the region's attraction centroid (Figure 21-3), that is the center of the "hottest" spot in the heatmap. This point on the map is displayed in Figure 21-4. The direction of any trip (radial or tangential) relative to c can be measured in the following way. Let $\overrightarrow{u_i}$ be the trip vector, with o_i and e_i as the corresponding initial and terminal points. Similarly, let $\overrightarrow{v_i}$ be the trip-to-centroid vector, with o_i and c as the corresponding initial and terminal points, then the trip direction r_i is the cosine of the angle between both vectors defined as:

$$r_i = \frac{\overrightarrow{u_i} \cdot \overrightarrow{v_i}}{\|\overrightarrow{u_i}\| \|\overrightarrow{v_i}\|}$$

The values r_i can take describe the following behavior:

close to +1 Trip movement is radial going towards c (= inbound)

close to -1 Trip movement is radial going in the opposite direction of c (= outbound)

close to 0 Trip movement is tangential

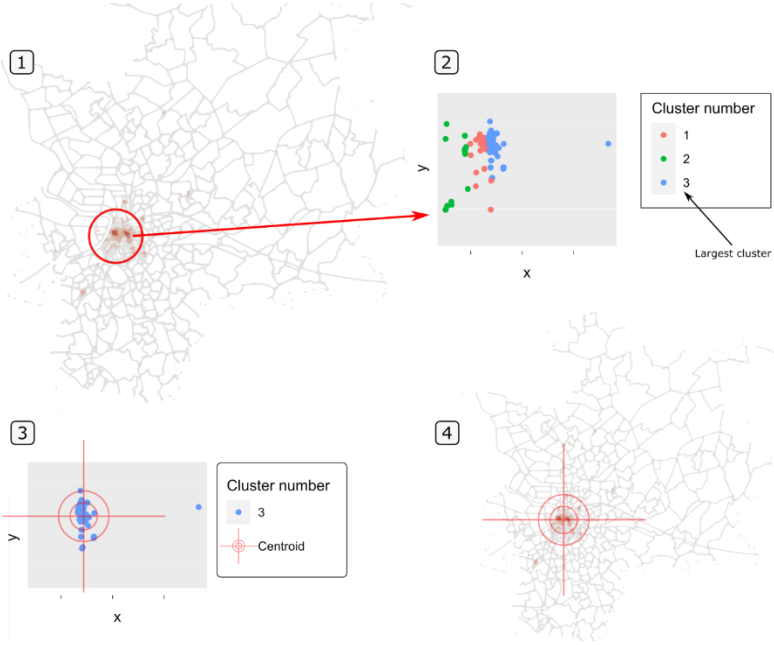

Figure 21. Finding a region's attraction centroid, (1) trip destinations in D, (2) densest area containing three clusters, (3) largest cluster and its centroid and (4) inferred region's centroid.

For the same sample dataset D, the radial movement is displayed in Figure 22, where most trips have radial movements (approaching and moving away from the centroid of the assumed region).

5.4.2.2 Hubs of commercial activity

Another variation of the region's center concept is to consider a set of centers (hubs) of commercial activity specially visited in rush hours, that is all the "hot" spots of minimum temperature in the heatmap instead of one single point. To obtain this information, a different approach can be taken and a dataset of public points of interest categorized by activity type is required.

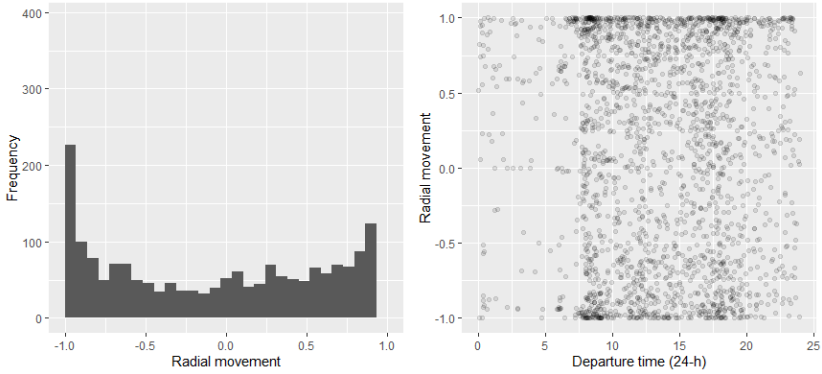

Figure 22. (left) Histogram of radial movement in dataset D, (right) behavior at different times of the day.

Fortunately these data sources are available online via open data projects such as (95) and (96). Let $p_i = (p_i{}^x, p_i{}^y, p_i{}^c)$ be a public POI i found in some public database, and its main attributes $p_i{}^x, p_i{}^y$ the location's cartesian coordinates and $p_i{}^c$ a given category. Then consider only those POIs that indicate some commercial activity in the category attribute such as:

$$p_i{}^c \in \{\text{"restaurant", "architect office", "pub", "cafe", "bank",...}\}.$$

Then a set of clusters $I = I_1, I_2, .. I_n$ is obtained, where the dissimilarity measure between two neighboring POIs p_i and p_j is its projected distance according to coordinates $p_i{}^x, p_i{}^y$ via a modified version of DBSCAN. A measure of the intra-cluster heterogeneity can be the average square distance of each POI p_i to cluster I_k's centroid c^k, defined as:

$$\sigma(I) = \frac{1}{|I_k|}\sum_i d(c^k, p_i)^2, \qquad p_i \in I_k$$

At last, hubs (clusters) of commercial activity $\hat{I} = \hat{I}_1, \hat{I}_2, .. \hat{I}_m$ should be found at zones with a dense concentration of POIs of certain categories, namely, those clusters for which the number of POIs exceeds n_{min} and the intra-cluster heterogeneity does not exceed σ_{max}.

$$\hat{I} = \{I_k \in I \mid \sigma(I) \leq \sigma_{max} \ \wedge \ |I_k| \geq n_{min}\}$$

A measure β_i to establish whether a trip i was made to one of these centers could be the distance to the closest one. That is:

$$\beta_i = \min_{j \ \in \ \hat{I}} \{d(e_i, j)\}$$

Where $d(e_i, j)$ is the distance from trip i's destination to the closest POI found in the border of activity center $j \in \hat{I}$. This measure is expected to be smaller for regular trips in the morning peak, since trips to work locations are expected to happen during this time interval. For the previous sample in dataset D, the results are displayed in Figure 23.

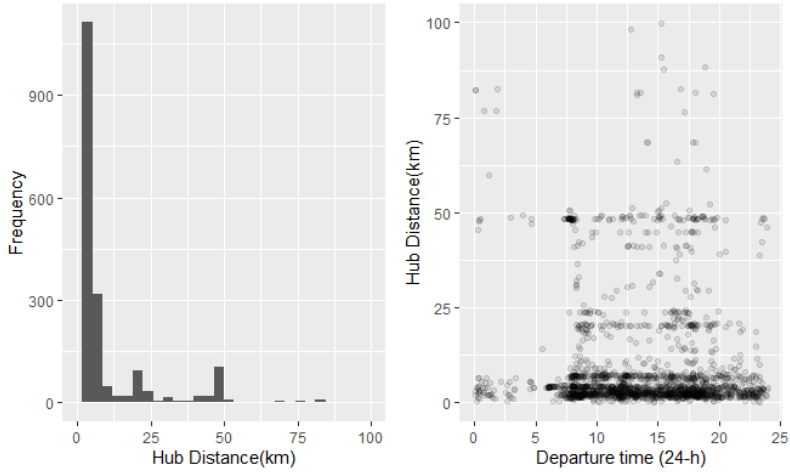

Figure 23. (left) Histogram of distance to activity hubs for destinations in dataset D, (right) behavior at different times of the day.

5.4.3 Acquiring multi-day knowledge

The next step is to define which multi-day attributes are intended to be transferred. For this, we need to define the data structure in D, consisting of travel histories generated from lifelogging data. The donor dataset must allow retrieving special information regarding a user's mobility behavior during several days through the detected routines. This is possible via lifelogging apps that collect spatiotemporal information about a user's activity. The trip segmentation of these spatiotemporal data between the detected origins and destinations produces a travel history between unique destinations (97).

Assuming a fraction of consistent chains can be retrieved from a travel history, the complete sequence of unique destinations, where an origin or destination can appear multiple times is:

$$p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_1 \rightarrow p_3 \rightarrow p_4 \rightarrow \cdots$$

Every subsequence of length two produces a new trip. Let ad_i and e_i, be the arrival date and the destination respectively of any trip T_i in history T, then the number of trips in a single date t to a unique destination p is:

$$y_{[day=t,\ p]} = |\{\, T_i \in T \mid ad_i = t \ \wedge \ e_i = p\}|$$

For the entire list of dates Φ in a travel history T, the average daily frequency of p is:

$$f(p) = \frac{\sum_{t \in \Phi} y_{[day=t,\ p]}}{|\Phi|}$$

Moreover, the **average daily frequency** of an OD pair (any "origin-destination"combination) is:

$$f(q,p) = \frac{\sum_{t \in \Phi} y_{[day=t,\ q,p]}}{|\Phi|}$$

where, $y_{[day=t,\ q,p]}$ is the number of trips in a single date t from origin q to destination p, that is:

$$y_{[day=t,\ q,p]} = |\{ T_i \in T \mid ad_i = t \wedge o_i = q \wedge e_i = p\}|$$

Likewise, the relevance of a unique destination p can be measured as the number of trips it attracts relative to the total number of trips in the history (how relevant the location is for the user).

$$r(p) = \frac{|\{ T_i \in T \mid e_i = p\}|}{|T|}$$

Then, the relevance of any OD pair called **OD regularity** (8) in this paper for a non-empty history $|T| > 0$, would be:

$$r(q,p) = \frac{|\{ T_i \in T \mid o_i = q \wedge e_i = p\}|}{|T|}$$

Now let a_dow_i be the day of the week trip T_i arrives to destination, then the probability of OD (q,p) on certain day of the week k can be measured by:

$$w_{[a\_dow=k,\ q,p]} = \frac{|\{ T_i \in T \mid o_i = q \wedge e_i = p \wedge a\_dow_i = k\}|}{r(q,p) \times |T|}$$

Which is only possible to compute if there are trips from $q\ to\ p$, so that $r(q,p) \neq 0$. If $k = 0$ for "Sundays", then the **weekday rate** for OD (q,p) (where values close to 0 entails higher frequencies on weekends and close to 1 on weekdays) is:

$$w(q,p) = \sum_{k \in \{1,..,5\}} w_{[dow=k,\ q,p]}$$

The distribution of each OD multi-day characteristics in D is presented in Figure 24.

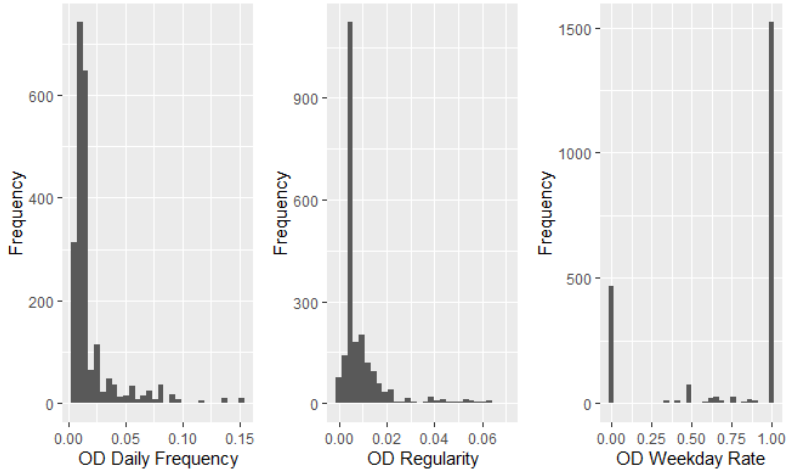

Figure 24. Histograms of multi-day characteristics in dataset D.

So far, with techniques in the methodology, matching variables in both datasets can been extended with the inferred trips, increasing matching rates and discovering new behavior that correlates with the multi-day information in the donor; the way to transfer it to the receptor is now explained in the following section.

5.4.4 Creating specific models

After sufficient matching characteristics are found in both datasets, the next step is to find pairs of comparable observations (twins) regarding these variables, so that all or part of the remaining characteristics can be transferred. That is, given any observation d_i in the donor dataset:

$$\boldsymbol{d_i} = [{x_1}^{(i)}, {x_2}^{(i)}, \ldots, {z_1}^{(i)}, {z_2}^{(i)}, \ldots], \qquad {x_k}^{(i)} \in {\boldsymbol{x_M}}^{(i)}, \quad {z_k}^{(i)} \in \boldsymbol{z}^{(i)}, \quad \forall \boldsymbol{d_i} \in D$$

Another observation r_j in the receptor can be matched, so that some multi-day characteristics of interest in $\boldsymbol{z}^{(j)}$ that are missing can be transferred, producing an enriched observation ${r_j}^{+}$.

$$\boldsymbol{r_j} = [{x_1}^{(j)}, {x_2}^{(j)}, \ldots, {y_1}^{(j)}, {y_2}^{(j)}, \ldots], \qquad {y_k}^{(j)} \in \boldsymbol{y}^{(j)}, \quad \forall \boldsymbol{r_j} \in R$$

$$\boldsymbol{r_j}^{+} = [{x_1}^{(j)}, {x_2}^{(j)}, \ldots, {y_1}^{(j)}, {y_2}^{(j)}, \ldots, {z_1}'^{(i)}, {z_2}'^{(i)}, \ldots], \qquad {z_k}'^{(i)} \in \boldsymbol{z}'^{(i)}, \quad \boldsymbol{z}'^{(i)} \subseteq \boldsymbol{z}^{(i)}, \quad \forall \boldsymbol{r_j} \in R$$

Basically, there are two ways to achieve the "transfer" process, a first attempt is to enrich an observation $\boldsymbol{r_j}$ in R by finding the nearest neighbor $\boldsymbol{d_i}$ in D that is statistically similar with respect to the common characteristics X_M. Another approach is to approximate a set of functions $f: \mathbb{R}^m \to \mathbb{R}$ through training data in D to estimate each of the missing characteristics $\boldsymbol{z}' = [{z_1}', {z_2}', \ldots {z_n}']^T$, based on values in $\boldsymbol{x_M} = [x_1, x_2, \ldots x_m]^T$, that is:

$${z_k}'^{(j)} = f({\boldsymbol{x_M}}^{(j)}), \qquad \forall \boldsymbol{r_j} \in R$$

For the following steps, the second alternative of learning functions is used. This way, it is not necessary to have as many observations in the donor as in the receptor; also, trip characteristics are not just copied from the donor, but instead values are approximated for similar observations. In order to generate the functions, the principal components that correlate with the multi-day variables are identified in the next section.

5.4.4.1 Identifying correlated components

Since not all the matching variables are required to learn the family of functions, studying those that correlate with the multi-day characteristics is a relevant step. For instance, by using k-means (52) with $k = 3$ with respect to OD regularity $r(q,p)$, three clusters of trips classified into: regular, barely regular and non-regular were produced. The differences between regular and non-regular trips for some characteristics are presented in Figure 25, where some correlation can already be seen, the red curves being the regular trips and blue curves the non-regular ones.

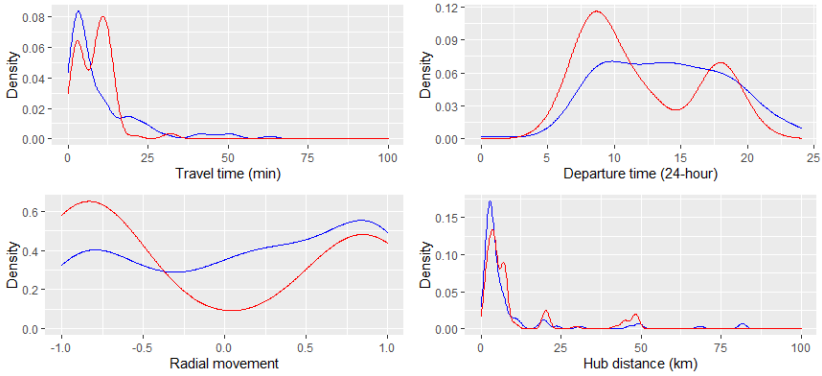

Figure 25. Correlations between regularity and other trip characteristics. Red curves are regular, blue curves are non-regular.

In the case of radial movement, correlation with regularity can be noticed if it is evaluated together with departure time. For instance, Figure 26 shows departure time in the horizontal axis and the radial movement in the vertical axis. It can be observed that most regular trips (large black spots) have radial movements and occur around rush hours. Similarly, most regular trips' destinations are close to commercial hubs and occur during rush hours, as seen in Figure 27. One possible approach to exploit these correlations is to produce a decision tree whose splits are decided by measuring the amount of randomness contained in the subsets for each attribute (characteristic) via entropies (98), so that datasets with clear patterns will produce small values. For a given multi-day attribute $z_k{}' \in \boldsymbol{z}'$, which possible values can be grouped in n classes with corresponding probabilities $p_1, p_2, .., p_n$, the entropy h_k can be defined as:

$$h_k = -\sum_{c=1}^{n} {p_c}^{(k)} \log_2 {p_c}^{(k)}$$

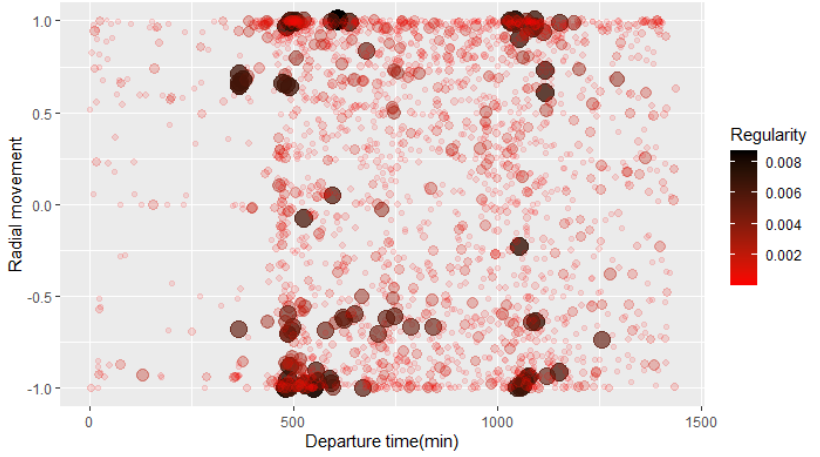

Figure 26. Correlation of regularity with departures and radial movement. Black large dots are regular trips.

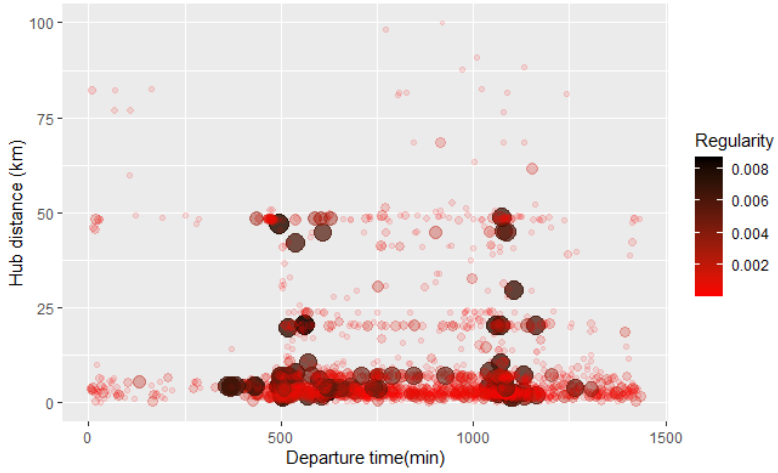

Figure 27. Correlation of regularity with departure time and hubs proximity. Black large dots are regular trips.

The highest value of h_k will occur when all classes of characteristic ${z_k}'$ have the same probability, meaning high randomness, and the lowest when there is a predominant class. The probability ${p_c}^{(k)}$ of an observation having class c in attribute k can be computed by:

$$ {p_c}^{(k)} = \frac{\left| \{ {x_k}^{(i)} = c \mid \boldsymbol{d_i} \in D \} \right|}{|D|} $$

If ${D_1}^{(q)}, {D_2}^{(q)}, .., {D_N}^{(q)} \subseteq D$ are the subsets produced when splitting dataset D by a matching variable $x_q \in \boldsymbol{x_M}$, then the new average entropy ${\bar{h}_k}^{(q)}$ of these N subsets is expected to change, so that the information gain ${I_k}^{(q)}$ is: ${I_k}^{(q)} = h_k - {\bar{h}_k}^{(q)}$

where

$$ {\bar{h}_k}^{(q)} = \frac{-\sum_{j=1}^{N} \sum_{c=1}^{n} {p_{c,j}}^{(k,q)} \log_2 {p_{c,j}}^{(k,q)}}{N} $$

The probability ${p_{c,j}}^{(k,q)}$ of an observation in the subset ${D_j}^{(q)}$ having class c in attribute k is:

$$ {p_{c,j}}^{(k,q)} = \frac{\left| \{ {x_k}^{(i)} = c \mid \boldsymbol{d_i} \in {D_j}^{(q)} \} \right|}{\left| {D_j}^{(q)} \right|} $$

The best attribute for a split is selected by:

$$ \underset{x_q \in \boldsymbol{x_M}}{\operatorname{argmax}} {I_k}^{(q)} $$

By taking not only the best attribute, but all those ${x_k}' \in {\boldsymbol{x_M}}'$ with a minimum information gain φ and ${\boldsymbol{x_M}}' \subseteq \boldsymbol{x_M}$, the best matching variables that describe the multi-day characteristic ${z_k}'$ are chosen. The function is updated to:

$$ {z_k}'^{(j)} = f({\boldsymbol{x_M}}'^{(j)}), \quad \text{where } {\boldsymbol{x_M}}' = \{ x_q \in \boldsymbol{x_M} \mid {I_k}^{(q)} \geq \varphi \} \qquad \forall \boldsymbol{r_j} \in R $$

A reduced version of a decision tree for a sample in dataset D is shown in Figure 28, so that characteristics correlated with a trip's regularity are learned, after transforming the task into a classification problem where 0 denote the non-regular trips and 1 the regular ones (see again Figure 25). In this plot, three characteristics were used for the splits, then at least these ones should become part of the matching variables in both datasets in order to transfer regularity. The same approach is used to identify correlations with the other multi-day attributes.

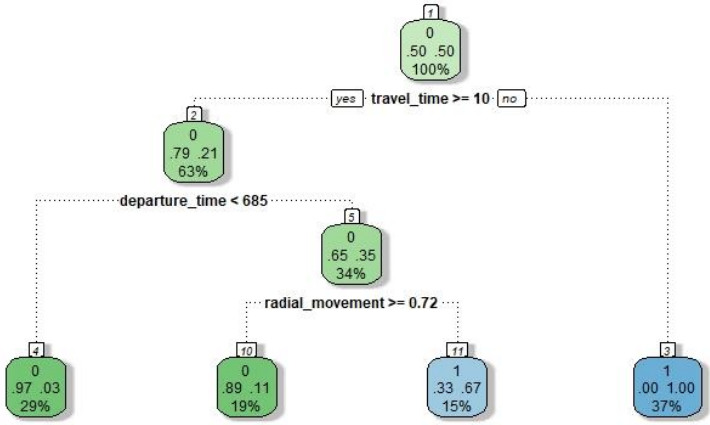

Figure 28. Decision tree to separate regular trips from non-regular to identify characteristics correlated with regularity.

5.4.4.2 Creating donation classes

In order to reduce effort in generating the regression models with the entire donor, an interesting approach (99) is to have clusters of trips with related characteristics and then a family of regression functions for each cluster. That is, at first instance an observation in the receptor is assigned a cluster of similar trips by using the matching variables found in the donor, and then a function is used to transfer specific information. A clustering procedure on trips in D will help identify groups of trips sharing the same characteristics. A variation of the original k-means algorithm (52) is used and the optimal number of clusters or donation classes k^* is evaluated by minimizing the intra-cluster variance.

$$k^* = \arg\min_{k \in K} \sum_{i=1}^{k} \sum_{j}^{n_i} \|c_i - x_j\|^2$$

$$\forall i: n_i \geq \delta$$

Where k, is tested for a finite number of clusters $K = \{1,2,..,k_{max}\}$, x_j refers to each observation, n_i is the number of observations in cluster i and c_i its centroid. For each donation class we have a separate set of functions to estimate the missing characteristics. For the same sample of D, when $k_{max} = 20$, the optimum $k^* equals\ 2$; characteristics of the two donation classes are presented in Figure 29 and Figure 30.

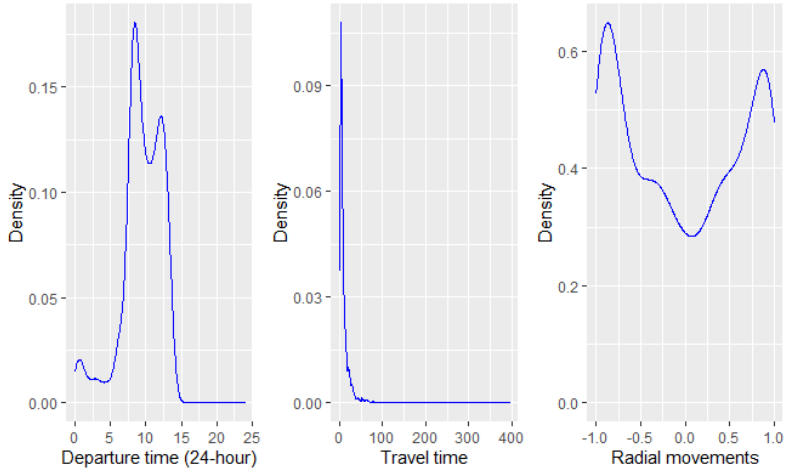

Figure 29. Distributions for three matching variables in donation class A.

In these figures, the two most prominent classes (morning and afternoon trips) are presented. Since trips in R must be assigned to one of the donation classes, a classifier is required. Using a decision tree, the performance on a validation set of 197 trips is displayed in Table 12, providing an average accuracy of above 97%.

Table 12. Confusion matrix to evaluate performance of Binary Classification of donor classes by random forests.

Reference / prediction	**Donor class A**	**Donor class B**	
Donor class A	78	2	80
Donor class B	3	114	117

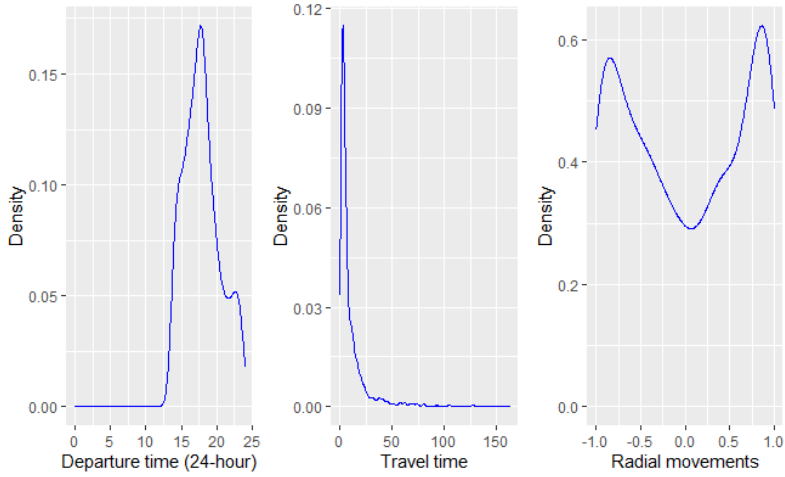

Figure 30. Distributions for three matching variables in donation class B.

5.4.5 Transferring information

In previous steps, a regression model for the OD regularity has been produced with each donation class in D by using as inputs the three relevant components identified in the previous stages (i.e. travel time, radial movement, and departure time). The same approach is taken for the other two multi-day characteristics: weekday rate and OD daily frequency; but including the already predicted characteristics for every new regression. So that after assigning trips to the donation classes, the regression performance (on a separate validation set) of the different models when 80% of data is used for training set and 20% for the validation set is presented in Figure 31 and Figure 32, where random forests (100) were used due to their high performance as reported in (101).

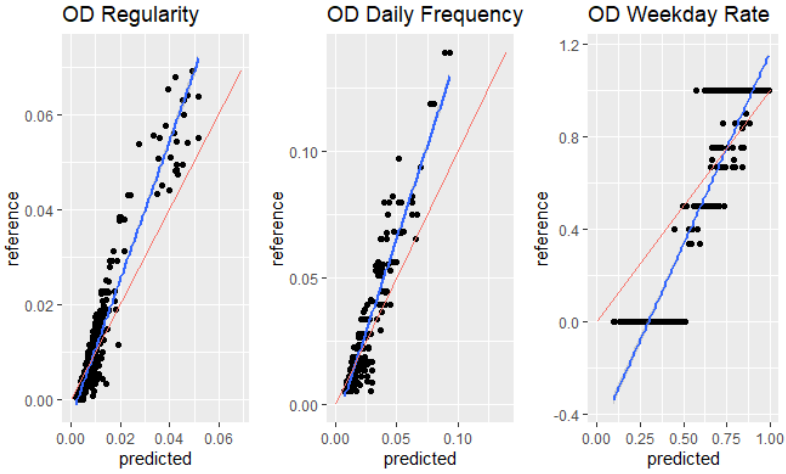

Figure 31. Performance on the validation set of the models for three multi-day characteristics in class A.

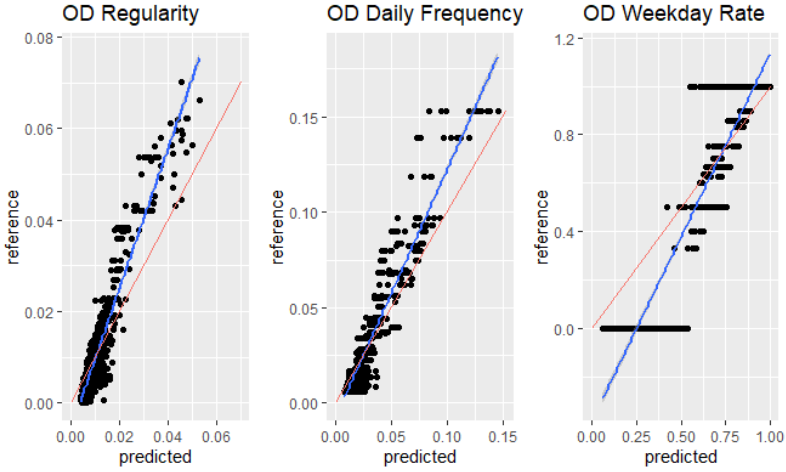

Figure 32. Performance on the validation set of the models for the three multi-day characteristics in class B.

In these plots, the blue lines are regression lines fitting the points and the red lines are references for perfect predictions. The estimates for regularity and frequency are slightly underestimated, predicting trips to be less regular than they actually are. This bias may or may not affect the simulations depending on the kind of scenarios. In order to predict each multi-day feature, the following inputs were used when training each model (see Table 13).

Table 13. Inputs for each specific model.

Predicted feature	**Inputs**
OD regularity	Travel time, radial movement, departure time
OD daily frequency	OD regularity, Travel time, radial movement, departure time
OD weekday rate	OD daily frequency, OD regularity, Travel time, radial movement, departure time

5.4.5.1 Populating a receptor with multi-day data

The next step is classifying trips in R as one of the donation classes and then using the corresponding functions created with the regression models. Since both datasets are known to belong to the same population, no further validation for compatibility is required. Moreover, the binary classifier allows knowing the probability of a record to belong to certain class, then if this probability is low for several observations, that is, if uncertainty is high, increasing the number of donation classes is an option.

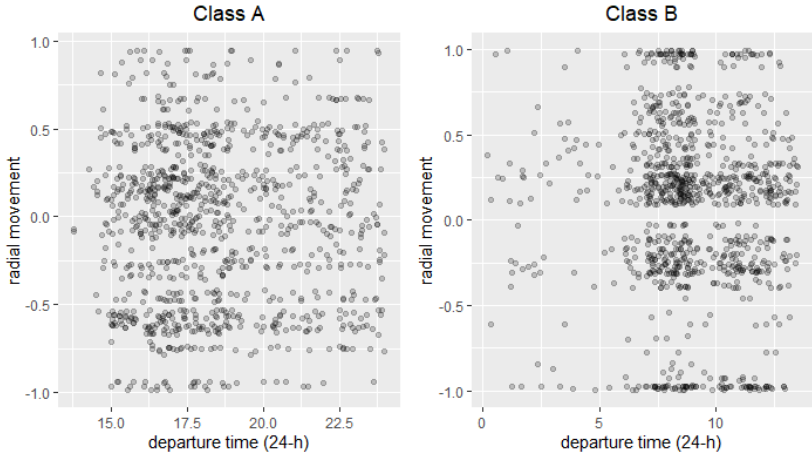

Figure 33. Trips in dataset R, after classification into donation classes.

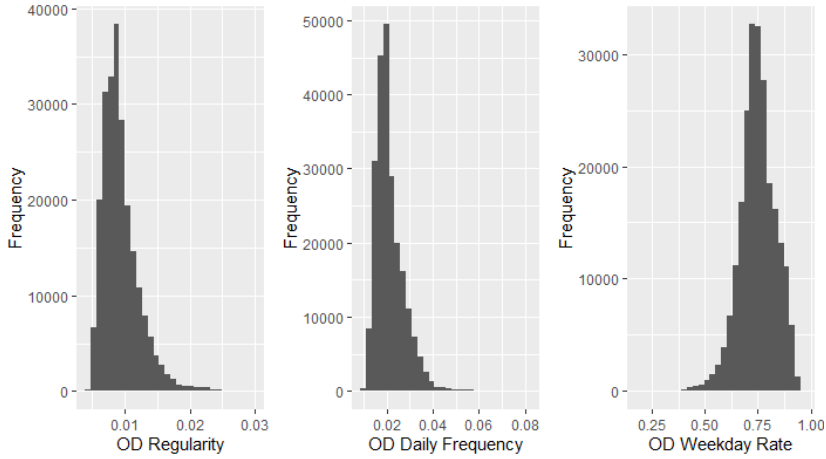

Figure 34. Distributions of the transferred multi-day in R+.

The results of transferring the three multi-day characteristics (OD regularity, OD daily frequency and weekday rate) from dataset D to a sample in R consisting of 230,000 trips, via models for the corresponding variables and donation classes, producing the enriched dataset $R+$ are displayed in Figure 34. The third plot confirms that $R+$ contains mainly weekday trips, which was expected given that demand datasets of this type normally represent an average weekday. Moreover, $R+$ should exhibit more regular trips than D as confirmed in Figure 35. This enriched dataset now allows simulations to be constrained to these multi-day characteristics, so that various transport and traffic-related services can be modelled for different scenarios. Next section provides some examples.

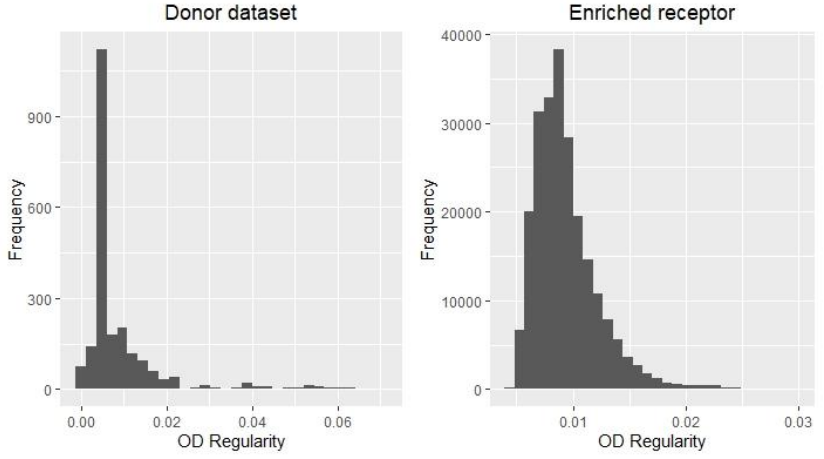

Figure 35. Comparing regularity on trips in dataset D and R+.

5.4.5.2 Shared mobility use cases

Having multi-day data in simulations is advantageous for evaluating complex scenarios, where OD regularity (probability) express the chance of predicting the trip by a software agent, and OD daily frequency the chance of finding that trip on the road. For instance, by adding assumed trajectories through shortest path algorithms (102) for illustration purposes only, it is possible to find trips on specific routes.

A first example considers all trips an app could predict for a certain OD, that is starting or going through an origin on the way to a known destination. Since originally, trip endpoints in dataset R are zone-based, the assumed "exact" locations for origins and destinations were sampled within each zone's polygon. Moreover, trajectories were added via Dijkstra's algorithm (103). For instance, the plot in Figure 36 shows the expected trips for overlapping a destination (blue circle), when daily frequency is set above 0.025 and departures occur after 12h00.

It may seem like there are many options to reach this destination by sharing rides. However, suppose that the ridesharing application is not capable of recognizing trips for sharing, unless they are sufficiently regular so that the agent has had ample opportunity to learn and predict this trip in a user's pattern. Then, in the same figure, the trips have been divided into predictable (regular) and unpredictable when the required minimum regularity of any OD is set above 0.01. The green trips in the right plot would thus be done, but they would be missed for ridesharing for lack of regularity. In addition, in this plot, the roads where the trips that can potentially be shared can be identified, suggesting the locations of the access points to pick up passengers. In the same example, each trip was considered shareable if any part of the traveled path was not more than 300 m (maximum walking distance) from the destination, and departures occurred between 8:00 and 11:00.

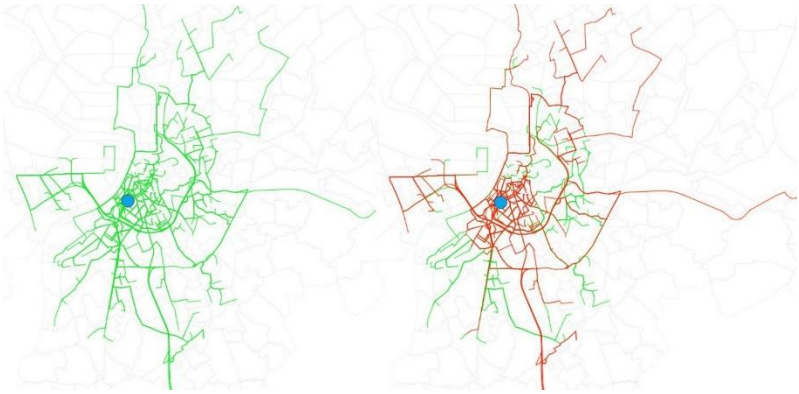

Figure 36. (left) Expected frequent trips to the marked blue circle, (right) predictable trips in red and non-predictable in green.

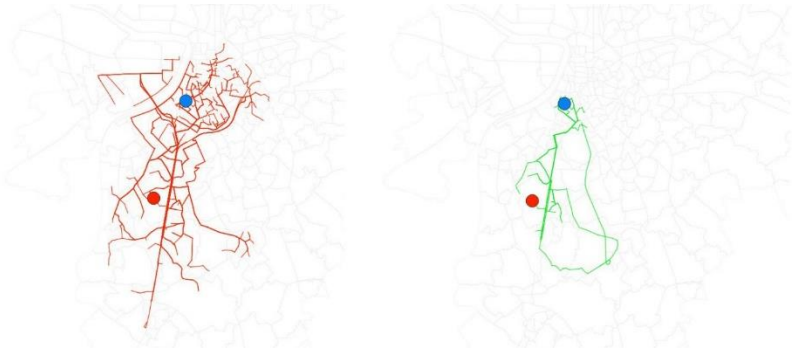

Figure 37.Expected non-regular trips between marked places, (left) weekday trips and (right) weekend trips. Origin denoted by red circle and destination by blue circle.

Another example shows a scenario where the ridesharing app would not be designed to recognize, predict and share regular trips, but rather shares trips for which the user first consults a navigation application. It can be expected that these are the less recurrent trips, so this ridesharing app design would capture more non-regular trips that can potentially be shared. The plot in Figure 37 shows the expected trips overlapping an origin (red circle) and destination (blue circle), when the daily frequency and regularity of the OD is set bellow 0.025 and 0.01 respectively. In the plot, the trips have been divided into weekday and weekend trips. It can be imagined that more detailed information can be extracted when adding temporal constraints.

5.5 CONCLUSIONS AND RECOMMENDATIONS

This paper has shown how multi-day information discovered in the logs of lifelogging apps can be transferred to another dataset, as long as they belong to the same population. Additionally, a subset of correlated matching variables must exist or can be inferred via data mining techniques, so that matching rates between datasets are boosted. Learning regression models on the donor, allows the receptor to be populated with additional trip characteristics; making it possible to generate a multi-day synthetic population with several characteristics. Potential uses for this enriched receptor include traffic simulations with multi-day constraints, as well as complex matching procedures for shared-mobility applications. Besides the multi-day characteristics, the socio-demographic and economic information already found in the receptor, will permit adding constraints to trips proposed as supply such as car ownership or income levels.

Since the probability of finding supply is approximated by the trip's daily frequency and predictability via its regularity, it is possible to simulate exceptional scenarios where an app makes predictions about the upcoming trips. Several applications of shared mobility can benefit from this approach because lifelogging data is continuously produced by existing mobile apps, allowing the automated generation of multi-day demand for simulation.

Data quality is important in R+ as simulations for several scenarios of the region of interest should lead to accurate predictions, and then to better decision-making for authorities and transport planners. This decreases risk of wrong judgments of implemented strategies in the study region and leads to steady improvements in mobility.

It can be imagined that if the actual supply could be extracted for a region (e.g. ridesharing campaigns where users register as drivers and include a list of the actual trips to share), so that it can be compared with the predicted trips by the app, some issues can be expected such as:

- Incorrect predictions of paths or departure times (false positives)
- Regular trips not being offered because these patterns have not yet been observed in the travel histories (false negatives)
- Vehicles' occupancy already reaching capacity (i.e. no more empty seats).
- drivers are simply not willing to share their rides with strangers

Nevertheless, risks get smaller as apps becomes smarter; such difficulties could be overcome by improving next trip prediction via better modelling, apps can confirm with drivers their willingness to participate in ridesharing, and occupancy can be measured or even predicted by smart sensors.

6 OVERALL CONCLUSIONS

This section aims to give answers to the research questions stated in the introduction of the document, by looking at the discussion of results after each publication. At the end of the document some future directions are also provided.

Is it possible to retrieve a consistent travel history from tracking data by inferring trips between locations from the spatio temporal information?

It could be evidenced that with data mining techniques and a proper calibration of the spatial parameters, a good approximation of the actual location of the stay points (and points-of-interest) can be achieved, together with a close estimate of the arrival and departures times. Nevertheless, there is space for improvement by adding more variables to the clustering procedure since mobile devices can collect lots of contextual information. Based on this principle it was found that mobility patterns in early stages of travel histories, greatly improve detection of personal points-of-interest (locations recurrently visited). For instance, known transitions between POIs help deciding whether a stay point must be labelled as a new unique location or as a place already seen in travel history. This way, POI detection becomes an iterative process, since chains of POIs allow extracting mobility patterns, but then these patterns improve detection of future POIs.

This approach allows customized POI classifiers for each user and the acquired knowledge constantly expanding, that not only includes spatio-temporal characteristics as in most research efforts, re-calibrates the existing models to improve performance. Additional variables can be explored such as day of the week and travel mode; these dimensions will provide more types of mobility patterns that can possibly improve classification performance.

Is it possible to mine multiday mobility patterns from each user's travel history in order to build trip prediction models conditioned to context information?

A methodology to learn these multi-dimensional mobility patterns from smartphone data at a user level has been provided, together with different measures to evaluate the multi-day characteristics of a trip. As soon as recurrent behavior is observed in a travel history, patterns can be learned, including but not limited to personal points of interest, typical arrival times or transitions between them. The new characteristics based on these patterns include:

- weekday rates, that is, how often the trip occurs on weekdays or weekends
- average daily frequency, how often the trip occurs during an average day
- OD regularity, how likely it is that the trip occurs when some prior information is known

These characteristics can be conditioned to knowledge previously observed by a software agent such as: an origin (current location), certain day of the week, time of the day, or any other sensed information. In the case of regularity, this measure provides the conditional probability of inferring the next trip and can be used later for simulating special scenarios. It has been shown that the multi-day information can be extracted via data mining techniques, although heterogeneity in travel histories is high (some users are more predictable than others). The number of patterns and their frequency have been found to decrease when more trip characteristics are considered, although regularity and hence prediction accuracy with respect to a set of conditions will increase.

Is it possible to transfer these multiday characteristics to synthetic populations to represent an entire region's behavior in order to simulate different shared-mobility scenarios of interest?

In this research, an approach fusing two datasets was studied. The first set consists of multi-day tracks from lifelogging data and the second one, synthetic home tours validated for a specific region. It has already been demonstrated that the extraction of multi-day characteristics is possible, as long as patterns can be found in the tracks. These characteristics have been proven to correlate in a certain degree with others such as: travel time, trip distance or departure times. There are some other characteristics not directly measured but that can also be computed, such as proximity to hubs of commercial activity or radial-tangential displacements. Then, the multi-day information can be transferred to a statistically compatible dataset, if a subset of the correlated matching variables can be found between both datasets, so that machine learning techniques produce models trained in the donor dataset to predict the missing data in the receptor.

This enriched receptor continues to be a representative sample of the population of the region, since the same probability distributions of each variable are kept; however, dimensionality has been expanded by discovering the new "hidden" travel features, which was only possible after data mining of the lifelogging data.

Is it possible to create simulations that mimic context-aware software agents, predicting ride matches at short term to evaluate different ridesharing scenarios?

The enriched receptor allows traffic simulations to include multi-day constraints, which is very useful for evaluation of shared mobility services such as ridesharing. For instance, scenarios where an app makes predictions (via conditional regularity) about the upcoming trips. Since lifelogging data is being produced all the time by existing mobile apps, this approach is convenient.

The simplest case is OD regularity, that is, the probability of a trip to certain destination when only the origin (current location) is known. Then, this value indicates how likely it is to predict the upcoming trip by an agent at short term, that is as soon as the location of the current trip has been identified. Moreover, this predictability measure can be constrained to other variables besides origin, so that conditional probabilities are used. For instance, this allows simulating the prediction of a trip conditioned to a certain time window and travel mode; so that predictions for pairs of users allow providing ride matches at short term in simulations. When trip trajectories are added, via simulation or via mining frequent routes between specific ODs complex scenarios are obtained.

Examples of these unique ridesharing scenarios could include:

- Ride matches predicted by agents during a morning peak between pairs of ODs.
- Ridesharing supply predicted by agents to reach a train station in certain time window from a given origin.
- Predicted hotspots of ridesharing (carpooling) supply to certain destination for certain day of the week.

7 FUTURE WORK

Some improvements are suggested on each stage of the research, moreover new steps are also proposed to continue with the research of new instances.

Data collection and aggregation

In general terms, the accuracy of the classifiers used for next-trip prediction and labelling personal points-of-interest can be increased by having more and more diverse contextual information, coming from sensors, remote servers and other agents. More experiments comparing different state-of-the-art techniques for next-trip destination using as input the enhanced context are required. At last, designing novel enriched context-aware systems for transport-oriented applications, taking advantage of the collected big data and continuous development of mobile technology. Examples of such context may include all kind of real time information, as well as proximity and interaction with other agents. Moreover, socio-demographic, and economic information explicitly entered by users (e.g. car ownership, household size), could become strong predictor of shareable trips.

Mining mobility patterns

Exploring additional "hidden" information by data mining techniques, allows mobility behavior of different class to be extracted. An enhancement of this research will be retrieving novel patterns that are not only based on spatio-temporal characteristics, but from specialized sensors, and even from unseen relations between existing variables. Based on these patterns, more multi-day characteristics could be obtained such as frequent paths of a trip, at user level or for a group of users with common behavior.

Demand generation

From this point of view, creating an enriched synthetic population with multi-day characteristics will benefit from extending the number of matching variables, as well as adding new multi-day variables to allow simulations with new constraints for more complex scenarios. The receptor dataset used in this research had zone-based home tours starting at fixed time windows; a nice addition would be to produce an activity-based population, where each agent has plans of sequential trip legs. Then, trips' endpoints will have specific coordinates with precise departure times. However, the best enhancement would be to make multi-day plans by using the probabilities of the expected trips for individual days of the week.

Multi-day simulations to test shared mobility

About this aspect, simulations in a multi-day format for several scenarios are possible. Shared mobility includes bikesharing, ridesharing, carsharing and other forms. Using the approach in this research to produce this class of demand, allows evaluating the potential of a service assuming some penetration rate, as well as planning strategies for their implementation and others. The multi-day characteristics allow splitting regular and non-regular trips, pretending when trips can be predicted by apps; moreover, demand for macro and microscopic simulations consisting of several days permit evaluating implementation effects in a long term.

REFERENCES

1. Furuhata M, Dessouky M, Ordóñez F, Brunet M, Wang X, Koenig S. Ridesharing: The state-of-the-art and future directions. 2013;(57):28–46.

2. Wang H, Yang H. Ridesourcing systems: A framework and review. Transp Res Part B Methodol. 2019;129:122–155.

3. Di Febbraro A, Gattorna E, Sacco N. Optimization of dynamic ridesharing systems. Transp Res Rec. 2013;(2359):44–50.

4. Simonetto A, Monteil J, Gambella C. Real-time city-scale ridesharing via linear assignment problems. Transp Res Part C Emerg Technol. 2019;101:208–232.

5. Alexander LP, González MC. Assessing the impact of real-time ridesharing on urban traffic using mobile phone data. Proc UrbComp. 2015;1–9.

6. Ma S, Wolfson O. Analysis and Evaluation of the Slugging Form of Ridesharing. In: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems [Internet]. New York, NY, USA: ACM; 2013. p. 64–73. (SIGSPATIAL'13). Available from: http://doi.acm.org/10.1145/2525314.2525365

7. Mendoza I, Tampère C, Mekerlé P. Anticipatory Assistance for Real Time Ride-Sharing in Environments of Pervasive Computing. In: 95th Annual Meeting of the Transportation Research Board. Washington, D.C.; 2016.

8. Mendoza I, Rydergren C, Tampère CMJ. Discovering Regularity in Mobility Patterns to Identify Predictable Aggregate Supply for Ridesharing. Transp Res Rec. 2018;0(0):0361198118798720.

9. Adomavicius G, Tuzhilin A. Context-aware recommender systems. In: Recommender systems handbook. Springer; 2011. p. 217–253.

10. Zickuhr K. Location-based services. Pew Res. 2013;

11. Agatz N, Erera A, Savelsvergh M, Wang Xin. Sustainable Passenger Transportation: Dynamic Ride-Sharing. Erasmus Research Institute of Management (ERIM); 2010.

12. Agatz N, Erera A, Savelsbergh M, Wang X. Dynamic Ride-Sharing: a Simulation Study in Metro Atlanta. 2011;

13. Pelekis N, Theodoridis Y. Mobility Data Mining and Knowledge Discovery. In: Mobility Data Management and Exploration7. New York: Springer; 2014. p. 143–67.

14. Ashbrook D, Starner T. Learning significant locations and predicting user movement with GPS. In: Wearable Computers, 2002(ISWC 2002) Proceedings Sixth International Symposium on. IEEE; 2002. p. 101–108.

15. Ashbrook D, Starner T. Using GPS to learn significant locations and predict movement across multiple users. Pers Ubiquitous Comput. 2003;7(5):275–286.

16. Samaan N, Karmouch A. A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. Mob Comput IEEE Trans On. 2005;4(6):537–551.

17. Chen Y, Mahmassani HS, Hong Z. Improving Online Network Traffic Prediction Through Data Mining and Pattern Matching for Dynamic Origin-Destination Demand Estimation. In: Transportation Research Board 94th Annual Meeting. 2015.

18. Zhou C, Bhatnagar N, Shekhar S, Terveen L. Mining personally important places from GPS tracks. In: Data Engineering Workshop, 2007 IEEE 23rd International Conference on. IEEE; 2007. p. 517–526.

19. Ester M, Kriegel H-P, Sander J, Xu X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In 1996.

20. Nurmi P, Bhattacharya S. Identifying Meaningful Places: The Non-parametric Way. In: Indulska J, Patterson DonaldJ, Rodden T, Ott M, editors. Pervasive Computing [Internet]. Springer Berlin Heidelberg; 2008. p. 111–27. Available from: http://dx.doi.org/10.1007/978-3-540-79576-6_7

21. Zheng Y, Zhang L, Xie X, Ma W-Y. Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th international conference on World wide web. ACM; 2009. p. 791–800.

22. Zheng Y, Liu L, Wang L, Xie X. Learning transportation mode from raw gps data for geographic applications on the web. In: Proceedings of the 17th international conference on World Wide Web. ACM; 2008. p. 247–256.

23. Gidófalvi G, Pedersen TB. Mining long, sharable patterns in trajectories of moving objects. Geoinformatica. 2009;13(1):27–55.

24. Lin M, Hsu W-J. Mining {GPS} data for mobility patterns: A survey. Pervasive Mob Comput. 2014;12(0):1–16.

25. Chon J, Cha H. Lifemap: A smartphone-based context provider for location-based services. IEEE Pervasive Comput. 2011;(2):58–67.

26. Mekerlé P, Verhaegen P, Tampere C, Himpe W, Plevka V. The prediction of commutes using data from consumer elecrtonics. In 2013.

27. Huang J, Tsai C-H. Improve GPS positioning accuracy with context awareness. In: Ubi-Media Computing, 2008 First IEEE International Conference on. IEEE; 2008. p. 94–99.

28. Antoniou C, Gikas V, Papathanasopoulou V, Danezis C, Panagopoulos AD, Markou I, et al. Localization and driving behavior classification using smartphone sensors in the direct absence of GNSS. In: Transportation Research Board 94th Annual Meeting. 2015.

29. Shen L, Stopher PR. Review of GPS Travel Survey and GPS Data-Processing Methods. Transp Rev. 2014;34(3):316–334.

30. White CE, Bernstein D, Kornhauser AL. Some map matching algorithms for personal navigation assistants. Transp Res Part C Emerg Technol. 2000;8(1–6):91–108.

31. Liao L, Patterson DJ, Fox D, Kautz H. Building personal maps from GPS data. Ann N Y Acad Sci. 2006;1093(1):249–265.

32. Yavaş G, Katsaros D, Ulusoy Ö, Manolopoulos Y. A data mining approach for location prediction in mobile environments. 2005;54(2):121–46.

33. Reddy S, Mun M, Burke J, Estrin D, Hansen M, Srivastava M. Using Mobile Phones to Determine Transportation Modes. ACM Trans Sen Netw. 2010 Mar;6(2):13:1–13:27.

34. Agatz N, Erera A, Savelsbergh M, Wang X. Optimization for dynamic ride-sharing: A review. N-Holl. 2012;

35. TEODOROVIC´ D, DELL' ORCO M. Mitigating Traffic Congestion: Solving the Ride-Matching Problem by Bee Colony Optimization. Transp Plan Technol. 2008;

36. CIARI F, BALMER M, AXHAUSEN K. Concepts for a large scale car-sharing system: Modeling and evaluation with an agent-based approach. In 2008.

37. Bicocchi N, Mamei M. Investigating ride sharing opportunities through mobility data analysis. Pervasive Mob Comput. 2014;14:83–94.

38. Bicocchi N, Mamei M, Sassi A, Zambonelli F. On Recommending Opportunistic Rides. IEEE Trans Intell Transp Syst. 2017;

39. Cici B, Markopoulou A, Frias-Martinez E, Laoutaris N. Assessing the Potential of Ride-Sharing Using Mobile and Social Data: A Tale of Four Cities. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing. 2014. p. 201-211).

40. Stiglic M, Agatz N, Savelsbergh M, Gradisar M. The benefits of meeting points in ride-sharing systems. Transp Res Part B Methodol. 2015;82:36–53.

41. Stiglic M, Agatz N, Savelsbergh M, Gradisar M. Making dynamic ride-sharing work: The impact of driver and rider flexibility. Transp Res Part E Logist Transp Rev. 2016;91:190–207.

42. Goel P, Kulik L, Ramamohanarao K. Optimal pick up point selection for effective ride sharing. IEEE Trans Big Data. 2017;3(2):154–168.

43. Williams MJ, Whitaker RM, Allen SM. Measuring individual regularity in human visiting patterns. In: Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom). IEEE; 2012. p. 117–122.

44. Wang Y, Yuan NJ, Lian D, Xu L, Xie X, Chen E, et al. Regularity and Conformity: Location Prediction Using Heterogeneous Mobility Data. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining [Internet]. New York, NY, USA: ACM; 2015. p. 1275–1284. (KDD '15). Available from: http://doi.acm.org/10.1145/2783258.2783350

45. Zhong C, Batty M, Manley E, Wang J, Wang Z, Chen F, et al. Variability in Regularity: Mining Temporal Mobility Patterns in London, Singapore and Beijing Using Smart-Card Data. PLOS ONE. 2016;11(2):1–17.

46. Caruana R, Freitag D. Greedy Attribute Selection. In: Cohen WW, Hirsh H, editors. Machine Learning Proceedings 1994 [Internet]. San Francisco (CA): Morgan Kaufmann; 1994. p. 28–36. Available from: https://www.sciencedirect.com/science/article/pii/B978155860335650012X

47. Butler H, Schmid C, Springmeyer D, Livni J. EPSG projection 31370 - belgian lambert 72. Spatial Reference. 2008.

48. Tukey JW. Exploratory data analysis. Addison-Wesley Publishing Company; 1977.

49. Bajaj R, Ranaweera SL, Agrawal DP. GPS: location-tracking technology. Computer. 2002;35(4):92–94.

50. Ankerst M, Breunig MM, Kriegel H-P, Sander J. OPTICS: ordering points to identify the clustering structure. In: ACM Sigmod Record. ACM; 1999. p. 49–60.

51. Zhou C, Frankowski D, Ludford P, Shekhar S, Terveen L. Discovering personal gazetteers: an interactive clustering approach. In: Proceedings of the 12th annual ACM international workshop on Geographic information systems. ACM; 2004. p. 266–273.

52. Jin X, Han J. K-means clustering. Encycl Mach Learn Data Min. 2017;695–697.

53. Gambs S, Killijian M-O, del Prado Cortez MN. Next place prediction using mobility markov chains. In: Proceedings of the First Workshop on Measurement, Privacy, and Mobility. ACM; 2012. p. 3.

54. Kang JH, Welbourne W, Stewart B, Borriello G. Extracting places from traces of locations. ACM SIGMOBILE Mob Comput Commun Rev. 2005;9(3):58–68.

55. Liao L, Patterson DJ, Fox D, Kautz H. Learning and inferring transportation routines. Artif Intell. 2007;171(5–6):311–331.

56. Luo T, Zheng X, Xu G, Fu K, Ren W. An improved DBSCAN algorithm to detect stops in individual trajectories. ISPRS Int J Geo-Inf. 2017;6(3):63.

57. Li D, Du Y. Artificial intelligence with uncertainty. CRC press; 2017.

58. Cao X, Cong G, Jensen CS. Mining significant semantic locations from GPS data. Proc VLDB Endow. 2010;3(1–2):1009–1020.

59. Vhaduri S, Poellabauer C. Opportunistic Discovery of Personal Places Using Multi-source Sensor Data. IEEE Trans Big Data. 2018;

60. Lee C, Yoon G, Han D. A probabilistic place extraction algorithm based on a superstate model. IEEE Trans Mob Comput. 2013;12(5):945–956.

61. Bhattacharya T, Kulik L, Bailey J. Automatically recognizing places of interest from unreliable GPS data using spatio-temporal density estimation and line intersections. Pervasive Mob Comput. 2015;19:86–107.

62. Mousavi SM, Harwood A, Karunasekera S, Maghrebi M. Geometry of interest (GOI): spatio-temporal destination extraction and partitioning in GPS trajectory data. J Ambient Intell Humaniz Comput. 2017;8(3):419–434.

63. Ying JJ-C, Lee W-C, Weng T-C, Tseng VS. Semantic trajectory mining for location prediction. In: Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM; 2011. p. 34–43.

64. Pappalardo L, Rinzivillo S, Simini F. Human mobility modelling: exploration and preferential return meet the gravity model. Procedia Comput Sci. 2016;83:934–939.

65. Feng J, Li Y, Zhang C, Sun F, Meng F, Guo A, et al. Deepmove: Predicting human mobility with attentional recurrent networks. In: Proceedings of the 2018 World Wide Web Conference. International World Wide Web Conferences Steering Committee; 2018. p. 1459–1468.

66. Zhang Y, Hu J, Dong J, Yuan Y, Zhou J, Shi J. Location prediction model based on Bayesian network theory. In: GLOBECOM 2009-2009 IEEE Global Telecommunications Conference. IEEE; 2009. p. 1–6.

67. Wu R, Luo G, Shao J, Tian L, Peng C. Location prediction on trajectory data: A review. Big Data Min Anal. 2018;1(2):108–127.

68. Zhang H, Dai L. Mobility Prediction: A Survey on State-of-the-Art Schemes and Future Applications. IEEE Access. 2018;7:802–822.

69. Silverman BW. Density estimation for statistics and data analysis. Routledge; 2018.

70. Taylor PJ. Distance transformation and distance decay functions. Geogr Anal. 1971;3(3):221–238.

71. Stiglic M, Agatz N, Savelsbergh M, Gradisar M. Enhancing urban mobility: Integrating ride-sharing and public transit. Comput Oper Res. 2018;90:12–21.

72. Yan X, Levine J, Zhao X. Integrating ridesourcing services with public transit: An evaluation of traveler responses combining revealed and stated preference data. Transp Res Part C Emerg Technol. 2019;105:683–696.

73. Andridge RR, Little RJ. A review of hot deck imputation for survey non-response. Int Stat Rev. 2010;78(1):40–64.

74. D'Orazio M. INTRODUCTION TO STATISTICAL MATCHING. :192.

75. Plevka V. STRATEGIC MOBILITY RESOURCE OWNERSHIP DECISIONS.

76. Mourad A, Puchinger J, Chu C. A survey of models and algorithms for optimizing shared mobility. Transp Res Part B Methodol. 2019;123:323–346.

77. Beckman RJ, Baggerly KA, McKay MD. Creating synthetic baseline populations. Transp Res Part Policy Pract. 1996;30(6):415–429.

78. Guo JY, Bhat CR. Population synthesis for microsimulating travel behavior. Transp Res Rec. 2007;2014(1):92–101.

79. Urbansim [Internet]. Open Platform for Urban Simulation. 2010 [cited 2020 Jan 1]. Available from: urbansim.org

80. Müller K, Axhausen KW. Population synthesis for microsimulation: State of the art. Arbeitsberichte Verk- Raumplan. 2010;638.

81. Ye X, Konduri K, Pendyala RM, Sana B, Waddell P. A methodology to match distributions of both household and person attributes in the generation of synthetic populations. In: 88th Annual Meeting of the Transportation Research Board, Washington, DC. 2009.

82. Bok M de, Jong G de, Baak J, Helder E, Puttemans C, Verlinden K, et al. A Population Simulator and Disaggregate Transport Demand Models for Flanders. Transp Res Procedia. 2015;8:168–80.

83. Verkeersmodellen afdeling Beleid. Strategisch Personenmodel Vlaanderen versie 4.1.0. In.

84. González MC, Hidalgo CA, Barabási A-L. Understanding individual human mobility patterns. Nature. 2008 Jun 5;453:779.

85. Alexander L, Jiang S, Murga M, González MC. Origin–destination trips by purpose and time of day inferred from mobile phone data. Transp Res Part C Emerg Technol. 2015;58:240–250.

86. Jiang S, Ferreira J, Gonzalez MC. Activity-based human mobility patterns inferred from mobile phone data: A case study of Singapore. IEEE Trans Big Data. 2017;3(2):208–219.

87. Demissie MG, Antunes F, Bento C, Phithakkitnukoon S, Sukhvibul T. Inferring origin-destination flows using mobile phone data: A case study of Senegal. In: 2016 13th International conference on electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON). IEEE; 2016. p. 1–6.

88. Anda C, Erath A, Fourie PJ. Transport modelling in the age of big data. Int J Urban Sci. 2017;21(sup1):19–42.

89. Croce AI, Musolino G, Rindone C, Vitetta A. Transport system models and big data: Zoning and graph building with traditional surveys, FCD and GIS. ISPRS Int J Geo-Inf. 2019;8(4):187.

90. Correa O, Ramamohanarao K, Tanin E, Kulik L. From ride-sourcing to ride-sharing through hot-spots. In: Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services. 2017. p. 136–145.

91. Wang Y, Kutadinata R, Winter S. Activity-based Ridesharing: Increasing Flexibility by Time Geography. In: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems [Internet]. New York, NY, USA: ACM; 2016. p. 1:1–1:10. (SIGSPACIAL '16). Available from: http://doi.acm.org/10.1145/2996913.2997002

92. Wang S, Gao S, Feng X, Murray AT, Zeng Y. A context-based geoprocessing framework for optimizing meetup location of multiple moving objects along road networks. Int J Geogr Inf Sci. 2018;32(7):1368–1390.

93. Wang Y, Winter S, Tomko M. Collaborative activity-based ridesharing. J Transp Geogr. 2018;72:131–8.

94. Transmob Project [Internet]. Institute for Mobility of Flanders. 2017 [cited 2017 Oct 10]. Available from: http://www.vim.be/projecten/transmob

95. Open Geodata stad Antwerpen [Internet]. Portaal Stad Antwerpen. 2020. Available from: https://portaal-stadantwerpen.opendata.arcgis.com/

96. OpenStreetMap Antwerp [Internet]. Wiki OpenStreetMap. 2020 [cited 2020 Aug 2]. Available from: https://wiki.openstreetmap.org/wiki/Antwerpen

97. Nurmi P, Koolwaaij J. Identifying meaningful locations. In: Mobile and Ubiquitous Systems: Networking & Services, 2006 Third Annual International Conference on. IEEE; 2006. p. 1–8.

98. Hu Q, Che X, Zhang L, Zhang D, Guo M, Yu D. Rank entropy-based decision trees for monotonic classification. IEEE Trans Knowl Data Eng. 2011;24(11):2052–2064.

99. D'Orazio M, Di Zio M, Scanu M. Statistical matching of data from complex sample surveys. In: Proceedings of the European Conference on Quality in Official Statistics-Q2012. 2012.

100. Goel E, Abhilasha E, Goel E, Abhilasha E. Random forest: A review. Int J Adv Res Comput Sci Softw Eng. 2017;7(1).

101. Aljahdali S, Hussain SN, others. Comparative prediction performance with support vector machine and random forest classification techniques. Int J Comput Appl. 2013;69(11).

102. Zhang L, He X. Route Search Base on pgRouting. In: Software Engineering and Knowledge Engineering: Theory and Practice. Springer; 2012. p. 1003–1007.

103. Dijkstra EW, others. A note on two problems in connexion with graphs. Numer Math. 1959;1(1):269–271.