Knowledge Discovery through Mobility
Data Integration

Paolo Cintia

Supervisor
Prof. Mirco Nanni

Supervisor
Prof. Dino Pedreschi
Abstract

In the era of Big Data a huge amount of information are available from every single citizen of our hyper-connected world. A simple smartphone can collect data with different kinds of information: a big part of these are related to mobility. A smartphone is connected to networks, such as GSM, GPS, Internet (and then social networks): each of them can provide us information about where, how and why the user is moving across space and time. Data integration has a key role in this understanding process: the combination of different data sources increases the value of the extracted knowledge, even though such integration task is often not trivial. This thesis aim to represent a step toward a reliable Mobility Analysis framework, capable to exploit the richness of the spatio-temporal data nowadays available. The work done is an exploration of meaningful open challenges, from an efficient Map Matching of low sampling GPS data to Inferring Human Activities from GPS tracks. A further experimentation has been performed over GSM and Twitter data, in order to detect and recognize significant events in terms of people presence and related tweets. Another promising perspective is the use of such extracted knowledge to enrich actual geospatial Datasets with a ‘Wisdom of the crowd’ dimension to derive, for instance, routing policies over road networks: most chosen paths among usual drivers are more meaningful than simple shortest paths.
Contents

I Background 15

1 Related Works 17
  1.1 The Map Matching Problem 18
  1.2 Human Activities Recognition 21
  1.3 Mobile Phone Data Knowledge Extraction 23

II Data integration: boosting the power of data 27

2 Exploiting GPS Data to assess Road Network Travel times 31
  2.1 Problem definition 32
    2.1.1 The gravity model 32
    2.1.2 Candidate selection 34
    2.1.3 Buffer method 34
    2.1.4 K-Neighbor method 35
  2.2 From speed estimation to travel time 35
  2.3 Validation 36
    2.3.1 Generation of synthetic observations dataset 37
  2.4 Performances study 37
  2.5 Real case study 39
  2.6 Summary 41

3 A Time Aware Map Matching Method to enrich Raw Spatio-Temporal Trajectories 43
  3.1 Time-Aware Map Matching 47
  3.2 Point-to-segment matching 47
  3.3 Path reconstruction heuristics 49
  3.4 Analysis of the algorithm proposed 52
  3.5 Implementation details 54
  3.6 Experimental results 55
3.6.1 ACM SigSpatial Cup dataset ........................................ 55
3.7 Experiments on large datasets ........................................ 58
  3.7.1 Datasets .................................................................. 60
  3.7.2 Temporal alignment ................................................. 60
  3.7.3 Middle-point Test ................................................... 61
  3.7.4 Applications .......................................................... 61
3.8 Conclusions ............................................................... 63

4 Inferring Human Activities from GPS Track .................. 65
  4.1 Concepts .................................................................. 66
  4.2 Methodology ............................................................. 70
  4.3 The ACTIVE process .................................................. 73
    4.3.1 ACTIVE Inference algorithm .................................... 75
  4.4 Experiments ............................................................. 78
    4.4.1 Trajectories Datasets .............................................. 78
    4.4.2 Stops, POIs and Activities ...................................... 79
    4.4.3 Evaluation .......................................................... 82
  4.5 Exploiting ACTIVE to infer the activity dynamics of a city .. 86
  4.6 Summary ................................................................. 87

5 Integrating Domain Knowledge and Mobility Data Mining to boost Routing Planners 89
  5.1 Proposed Analytic Model ............................................. 91
  5.2 Experiments ............................................................. 92
  5.3 Deviations Analysis .................................................... 95
  5.4 Towards an Experienced Route Planner .......................... 97
  5.5 Summary ................................................................. 101

III Conclusions, ongoing and future works ...................... 103

6 Detect and recognize events by combining CDR and Twitter data ..... 107
  6.1 Case Study ............................................................. 109

7 Extract and exploit Mobility Routing Pattern .................. 115
  7.1 Leveraging Road Network Hierarchy through Big Data ........ 115
  7.2 Evaluate Routing Policies according to the Wisdom of the Crowd .. 117

8 A particular type of Mobility Data: sports analytics .......... 121
  8.1 Analyzing Cyclists’ performance with Data Mining techniques .. 121
Introduction

In the world of Big Data, mobility is one of the most interesting phenomena to study, both for technological and philosophical reasons. Human beings base their life on moving from a place to another one to satisfy their needs. This era of Data provides a new, giant, tool: a global microscope to observe how people move across time and space and, thus, how they live. Mobility is in fact one of the main aspect of our life. An average estimation on hours spent driving [51] is about 100 minutes per day: moving activities cover the 7% of a lifetime. By moving we contribute to write a microscopic part of world’s evolution. In his formalization of the Science of the Cities, Batty [7] explains how the cities where we live evolve mainly from the bottom up as the product of millions of individual and group decisions. A city, a region, a country, are more like biological environments than mechanical systems: they admit innovation, generate surprise and display catastrophes. Big Data are what it was missing to prove and exploit theories such as the Science of the Cities. Nowadays we have the proper lens to observe this complex, huge, multiform environment. Mobility Big Data together with the evolving tools of Data Science represent a reliable framework to understand all the dynamics of human mobility: unlike past years, today every person is producing different analyzable tracks of his/her mobility. As an instance, data gathered from the most popular devices among us, cellphones, are already used to assess the characteristics of a given territory in terms of residents, commuters, and visitors [25]. Approximately 97% of world population owns a GSM cellphone, this means that by acquiring and analyzing data log of GSM Antennas –anonymized, of course– we have the best possible view on human mobility, and, thus, on all the underlying dynamics such as urban flows, migrations, tourism and so on. However, GSM Data represent only a dimension of mobility, since they reveal people presences in a certain area at a certain time but not how people reach that area nor what was the purpose of those trips. But Mobility Data are not only composed by our cellphone logs. A simple smartphone has various sensors, supporting tons of mobility apps such as Navigation Assistants and Social networks.

The ensemble of all the possible kind of data produced by billions of everywhere-
connected users represents the big mosaic of mobility data. Every single dowel of such mosaic encloses a dimension of human mobility, every right combination of two or more dowels is a step less toward the understanding of big picture. So far, main results in Mobility Data Analysis have been the comprehension of (almost) all the single dowels; the next challenge is to compose the whole mosaic.

The aim of this thesis is to address and accept the challenge of Data Integration: combining different sources of Mobility Data lead to significant knowledge gain in terms of semantics, thus increasing the value of performed analysis. As an example, while GSM data encloses information about where people are and what mobility profile they belong to, GPS data express how people move across time and space. Social networks such as Twitter, instead, can suggest an association between a text string (the tweet) and a particular place. Furthermore, a semantic enrichment process can also benefit from the integration of such user-generated mobility data with a knowledge base. In particular, the integration between GPS data and Open Data such as OpenStreetMap and Google Places is the main focus of the thesis, while attempts to explore other directions (i.e. GSM and Social Media data integration) are still on going.

The availability of huge datasets of GPS data have introduced plethora of opportunities for Mobility Data analysts, hence opening up different kind of problems. GPS datasets are commonly collected by devices installed into cars for insurance purposes: for each monitored car a device uses to store a point recording its features, such as longitude, latitude, timestamp, speed, heading. These datasets have been already analyzed and exploited in many different ways, capturing the attention of the scientific community. In 1998, Abraham and Roddick [2] were the first ones who perceived the potential of spatio-temporal databases. The need of data mining techniques able to deal with spatio-temporal data arose and produced first results starting from the immediately next decade. Works such as [30] and [27], where authors provided methods to extract pattern from GPS data, represents the foundations of Mobility Data Mining. The contribution of this thesis is a climb over the semantic levels of GPS data analysis; although state-of-the-art works already figured out how to efficiently extract knowledge from raw mobility data, there is room for improvement when the goal is to enrich such pattern with contextual data, e.g. associated roads actually traversed to GPS points. The main achievements of the work done over this three years represent an improvement of data integration techniques, focused on exploiting existing knowledge bases such as OpenStreetMap [32] – to enrich different datasets of Mobility Data.

The workflow of the thesis is composed by the following contributions [and the respective publications]:
• **Road Network Travel Times Estimation** [18]: in this line of work we tackle the problem of generating traffic information in time-dependent networks using GPS data. A GPS point recorded by widely-adopted cars insurance devices store instant speed of vehicles, together with usual spatio-temporal data. To exploit such information, every GPS point needs to be matched to the road segment the car was traveling in. This task is not trivial: several works in literature make strong assumptions on the error distribution that we want to drop, proposing a gravitational model method to compute road segment average speed from trajectory data. Furthermore we show how to generate travel-time functions from the computed average speeds useful for time-dependent networks routing systems. This approach allows creating an accurate picture of the traffic conditions in time and space.

• **Time Aware Map Matching** [14]: The process of associating a segment of the underlying road network to a GPS point gives us the chance to enrich raw data with the semantic layer provided by the road-map, with all contextual information associated to it, e.g. the presence of speed limits, attraction points, changes in elevation, etc. Most state-of-art solutions for this classical problem simply look for the shortest or fastest path connecting any pair of consecutive points in a trip. While in some contexts that is reasonable, in this work we argue that the shortest/fastest path assumption can be in general erroneous. Indeed, we show that such approaches can yield travel times that are significantly incoherent with the real ones, and propose a Time-Aware Map matching process that tries to improve the state-of-art by taking into account also such temporal aspect. Our algorithm results to be very efficient, effective on low-sampling data and to outperform existing solutions, as proved by experiments on large datasets of real GPS trajectories. Moreover, our algorithm is parameter-free and does not depend on specific characteristics of the GPS localization error and of the road network (e.g. density of roads, road network topology, etc.).

• **Semantic Enrichment of GPS Track** [24],[16]: spatio-temporal datasets proved to be very useful to analyze and understand mobility behaviors of citizens but, at the same time, poor in terms of semantics: we can infer where and when people move, but not the purposes of their movements. This work is a step in the direction of semantic enrichment of mobility data. We define a method, called ACTIVE, to associate the trajectories stops to the most probable activity performed, analyze the Points of Interests in the stop neighborhood, and exploit the gravity model. Experiments done show the good accuracy of the algorithm when compared to a ground truth.
• **Route Planning exploiting Wisdom of the Crowd** [31]: the data integration process could also be viewed as a cycle. The information gain of the first loop (e.g. map matched GPS data) becomes a data source itself, to be combined with further mobility data. Map matched trajectories represent the most precise proxy of road network usage, providing the possibility to learn how drivers route across the network to reach a destination. Modern route planners – such as personal navigation assistants – generally return routes that minimize either the distance covered or the time traveled. However, these routes are rarely considered by people who move in a certain area systematically. Indeed, due to their expertise, they very often prefer different solutions. In this paper we provide an analytic model to study the deviations of the systematic movements from the paths proposed by a route planner. We extract the systematic movements from a set of drivers and translate them into sequences of road segments, in order to compare such sequences with the shortest/fastest path relative to the respective systematic movement. Our results show that about 30-35% of the systematic movements follow the shortest paths, while the others follow routes which are on average 7 km longer.

Fig. 1 shows a general schema describing the contributions provided by this thesis and the relations among them. As depicted, the point-to-segment matching method and travel time estimation algorithm is the starting point for all the subsequent contributions. In fact, a point-matching task is involved in all the subsequent works. Furthermore, results from the application of Time-Aware map matching algorithm are also part of the input of data for the analysis of users’ route planning policies.

This thesis is organized as follows: in Part I there is a review of state-of-the-art works regarding the challenges faced. Part II describes and explains the main contributions of this thesis. Part. III summarizes some preliminary results obtained on new challenging analytical problems that, in part, follow the above mentioned achievements up, and in part explore different directions in the domain of Mobility Data analysis.
Figure 1: Workflow of the thesis. The first result has been the base to explore Mobility Data Integration research field, thus starting an integration process where the result of a first step is itself a further data source.
Part I

Background
1

Related Works

In this era of Big Data a huge amount of information are available from every single citizen of our strong connected world. A simple smartphone can collect data enclosing different kinds of informations: a big part of these are related to mobility. A smartphone is connected to networks, such as GSM, GPS, Internet (and then social networks): each of them can give us information about where, what and why the user is moving across space and time. Understanding human mobility is an important challenge, and the comprehension of such phenomena have conquered the attention of many scientific communities: physicists, sociologists, are working together with computer scientists to take advantage of the chance given by the increasing amount of easy-collected Mobility Data. The knowledge on human mobility - fundamental for application such as urban planning, traffic forecasting or spread of biological and mobile viruses - is getting bigger, helping decision makers on their job. To give a quick overview, in a recent and popular study regarding the diffusion of Ebola virus ([29]), authors developed a statistical model by integrating daily airline passenger traffic worldwide and the disease model obtained combining the community, hospital, and burial transmission dynamics. Mobility data analysis have been involved in the understanding of a case that gained the attention of all the main institutions all over the world, and the solution proposed has been built through the aggregation of multiple types of data.

Data integration has a key role indeed. The combination of different data sources increases the value of the extracted knowledge, even though the integration process is often not trivial. In the following sections there is a review of state-of-the-art methods regarding the Data Integration challenges faced in this thesis.

First contributions in the field of Mobility Data Analysis regarded the process of Trajectory Mining. When the possibility to analyze huge dataset of spatio-temporal data arose, many researchers started to develop methods and algorithms to find pat-
tern and common behaviors among those datasets. A spatio-temporal trajectory can be represented as a sequence of points \((x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n)\), where \((x_i, y_i)\) is the spatial location, consisting in longitude and latitude, and \(t_i\) represents the timestamp, i.e. the time when the location \((x_i, y_i)\) is recorded. Trajectory mining have been defined by extending traditional Data Mining techniques to such spatio-temporal data: clustering, join, classification, are now operation available also over trajectory data. One of the first paper in this field is [27], where Trajectory Pattern Mining has been proposed as a model able to provide a concise description of frequent behaviors in term of both space and time. Pattern discovery among trajectories was also the topic of [30], where authors addressed the problem of modeling subgroups of trajectories that exhibit similar movement and proximity for a certain amount of time. Once introduced the concept of spatio-temporal pattern, the next challenge has been to go up through the possible semantic layers of Mobility Data. This has lead to the ipothesis of a framework able to extend the concept of mobility pattern: mobility data sources are nowadays manifold, describing different meanings of human mobility: it is not only where and when we go, but also why we go, and what we are going to do. The wide field of Trajectory Data Mining can benefit from the evolution of Data Integration processes. In [50] authors propose a method to estimate traffic conditions based on GPS trajectories, historical data, and contextual informations such as the Point of Interests associated to each road segment. The complex task of Trajectory Classification (see [52], [36]), that is the process of recognize and asses the characteristics of a trajectory (e.g. still/moving, transportation modes, activities performed) is already taking advantage of POIs to train the machine learning models –i.e. Conditional Random Fields, Hidden Markov Model- involved in the classification process.

The work done in this thesis aim to face the challenge of enriching the semantic informations extracted from mobility data through the integration of multiple data sources. The focus has been posed on some particular open problems, which are defined in the following subsections.

1.1 The Map Matching Problem

Map matching is the process of associating a sequence of spatio-temporal points to a connected sequence of road segments, thus enriching raw data with the semantic layer provided by the roadmap and all the contextual information associated to it. Although it is a classical and well known task in GIS literature, the map matching problem still represents an important and a valuable challenge. The map matching
problem can be treated at two different scales, depending on the characteristics of input data, which can be made of either high-frequency or low-frequency samples of the real position of the recording device. The former is mainly treated in the field of Personal Navigation Assistants, where the device is able to identify in real time the road where the user is traveling. The latter is common for applications dealing with smartphones or GPS-equipped black boxes installed vehicles for security or insurance purposes. This kind of devices sample and store their location at a lower frequency to limit the battery consumption or to reduce data exchanging between the device and the server that stores the information. The result is a coarse-grained GPS data, harder to deal with but still with high value: this data represents the most reliable proxy for road network mobility. One important issue introduced with low-frequency samples is path reconstruction. After mapping single points to the road network, between two consecutive locations there might be a significant gap, therefore requiring strategies to reconstruct the path traversed by the vehicle or the individual.

Map matching algorithms are classified according to three categories: global, incremental and probabilistic algorithms. The focus of this review is on global and incremental algorithms, since the probabilistic approach (e.g. Kalman Filters) is used to tackle the high-sampling rate map matching problem.

Global algorithms solve the problem by considering the entire trajectory, the solution is obtained by searching the closest path in the map w.r.t. the input trajectory. In [4] there is a first example of global matching algorithm: map-matching is the result of a spatial query, the resulting road network path has the minimum Fréchet distance w.r.t to the input trajectory. Fréchet distance is a mathematical model used to compare two curves: in its more common and easier illustration, it is viewed as the minimal length of a leash between a dog and his owner, whom are walking on different curves. The complexity of this approach is quite high: $O(nm \log^2 nm)$, with $n$ as the number of trajectory points and $m$ the number of road network edges. In [12] a more efficient version of Fréchet distance computing algorithm has been provided.

The main issue for global algorithms is the purely geometric approach. All the characteristics of the road network are ignored, the matching is only based on the research of a similar curve. It is obvious to notice that there will never be a low-sampled trajectory completely equal to a path in the network: this means that there is not a precise definition of the optimum to reach. In [10] Fréchet distance is even used for a quality evaluation of the results obtained. The incremental approach for low-sampling map matching is based on joining optimal local searches. The local optimum is represented by the most probable path between two consecutive matched GPS points. IVMM algorithm ([63]) is one of the state-of-the-art incremental map
matching algorithm we used to compare the results of our work. The matching process is done through consecutive steps; first of all, a preliminary refinement is done by dropping vertex according to a spatial range query. Then, the matching probability is computed assuming the GPS error with a normal distribution; this position probability is combined with a transition probability, that is the ratio between the euclidean distance of two candidates and their shortest path distance on the road network. Furthermore, a temporal analysis is also considered: the cosine similarity between the travel-time (according to road speed limits) of the shortest path analyzed and the real time difference between the two GPS points. The last step of IVMM is a mutual influence modeling, used to decide the path between each consecutive points by considering also at the global trajectory. In this approach, there are several not verified assumptions: first of all a driver should always follow the shortest path. Moreover, the radius of the range query is arbitrary and the GPS error is assumed as Gaussian, with fixed parameters. Furthermore, the travel-time of road edges is obtained according to road speed constraints. These constraints, especially on a city road network, could be considered as arbitrary. In [18] a method to get rid of this assumption has been proposed: a gravity model is used to associate a GPS point, with his speed, to a road segment, choosing between the k-nearest neighbor.

A particular case of probabilistic approach used to deal with low-sampling rate data is [40], where authors developed a map matching algorithm based on the well known Hidden Markov Model. This algorithm has a weak point on its highly dependance on two parameters, both of them obtained from the ground truth, i.e. from the road segments actually traversed from the vehicle; this kind of data are not available in a real application scenario of a map matching algorithm. Another factor of weakness is its complexity: for each trajectory, Viterbi algorithm takes $O(|C| * |S|^2)$ to find a solution, where $C$ is the set of transitions between segments and $S$ the set of all the segments candidates to be matched with a point of the input trajectory. It is worth to point out that for all the transitions in $C$ the shortest path is computed, thus adding the complexity of this computation, that is $\sum_{p \in P} |C_p|^2 * (|E| + |V| log |V|)$. Here $C_p$ is the subset of candidates segments associate to the single GPS point $p$. In the next part of this thesis there is a comparison between a new approach for solving the map-matching problem, namely ”Time-Aware map matching algorithm”, and the state-of-the-art methods reviewed above. The comparison has been made on three different datasets, either with a ground truth (ACM sigspatial cup 2012 []) and gathered from on board GPS devices. The validation of a map-matching algorithm designed to process a real GPS dataset is not trivial indeed. Datasets actually available (either public or not) are usually stored without the roads traversed
by vehicles that have generated the GPS tracks; in this scenario, checking whether a map-matching is correct or not is not possible. Hence, every method based on a machine learning approach can not be exploited on such GPS datasets, where a ground truth is not provided. However, the validation of Time-Aware algorithm consider both -ground truth available/not available cases - in order to provide a detailed overview of the goodness of its approach.

1.2 Human Activities Recognition

The importance of Mobility Big Data is represented by the information they encapsulate: the traces of people’s activities. All of our movements are done with a specific purpose. The understanding of why a person moves is the last frontier in the semantic enrichment of Mobility Data. In other words, inferring the activity carried out by a moving individual from the raw mobility data, in absence of any metadata about the intention of user, is a challenging task that can bring highly innovative contribution to the study of human mobility behavior. Nowadays, several application areas would benefit from an extensive study on people’s activities such as traffic management, public transportation, commercials and advertising, security and police, hazard evacuation management, location based services and so on. Despite the fact that data collected from mobile devices is increasing its location accuracy, it is not improving in the same way their quality in terms of semantic richness. This means that there is a semantic gap between raw data collected from mobile devices and the personal activity that generated the traces. As a consequence, techniques to semantically enrich the collected data are necessary to (semi-) automatically infer the person’s activity given her/his location traces.

Inside this scenario, it is possible to address two different research trends: (1) the detection of stops in trajectory data and the relative association of a place and (2) the inference of the activity performed during the stops.

Regarding the first trend, it is worth to recall the pioneering work of [53] where authors propose a conceptual model for semantic trajectories. While trajectories are defined as a time-space function that records the changing of the positions of an object moving in space during a given time interval, semantic trajectories are defined as sequences of stops (where the moving object stays still during a time interval) and moves (the part of a trajectory where the position of the object changes). The basic assumption behind the notion of stop is: the place where a person stops is of some interest for her/him. Therefore, each stop is associated to an activity. The inference of the kind of activity performed requires the definition of a relation between Mobility Data and domain knowledge base. Such informations have two main sources:
(i) user direct annotation and (ii) Point-Of-Interests (POI) dataset. Many researches are based on volunteer’s tracked data, where users provide a complete survey of the activities associated to tracking data, while a completely different approaches relies on the identification of the POIs visited by the user during each stop.

The association between a POI and a trajectory stop is the objective of several approaches, ranging from the simplest (associating the closest like in [43]) to more sophisticated proposals, reported in [46]. However, most of the approaches do not explicitly consider the temporal validity of the association (i.e. if the POI exists or it is accessible during the actual stop), neither the probability value associated to each stop-POI pair, nor the concept of activity.

The identification of mobile activities from trajectories of people is not new in the literature ([64]). A trend of research is devoted to the identification of transportation means like the work [65]. Using speed, acceleration and speed change rate, authors first detect the positions where the movement switches between walking and non-walking. In a second step, they refine the non-walking segments into segments characterized by the other transportation modes: bicycle, bus, and driving. They use a combination of techniques, from supervised learning to decision tree inference, and add a post-processing step to improve the accuracy of the segmentation. The post-processing step relies on a graph that contains commonsense constraints about the real world and typical human behaviors.

Another trend is concentrating on the identification of the activity during a stop. In [59] authors present a method to automatically extract sequences of activities from large set of trajectory data. The assumption is that activities may be carried out at a POI during a stop in the user trajectory. The association between a stop and a POI - as in our case - is crucial and may depend on several factors. One is the distance between the POI and the trajectory and the other is the duration of the event. They base their approach on the concept of influence distance for associations among trajectories, POIs, and activities. Influence is a distance based measure, such that a trajectory $T$ can only be associated with a POI if there exists at least one point on $T$ that is influenced by the POI. They use the Voronoi diagram as a division of the area where each cell represents the influence area of the POI. They test their algorithms using synthetically generated trajectories dataset with the POIs collected in a specific area in California. Naturally, the drawback of this testing is that there is no real validation of the method since there is no proof of the correctness of the inferred POIs.

The work of [33] is again in the direction of inferring activities from users trajectories. This paper presents an approach using spatial temporal attractiveness of POIs to identify activity-locations and durations from raw GPS trajectory. The algorithm they propose finds the intersections of trajectories and spatial-temporal attractive-
ness prisms to indicate the potential possibilities for activities. The experiments use one month's GPS trajectories from 10 volunteers where they show an high accuracy of the method.

[35] propose a method for user intention recognition in the mobile case. They propose a framework where movement information through GPS data is used by a system of production rules and classification techniques for the intention recognition process. This approach mainly focuses on movement features such as speed, angles etc. to segment a trajectory, whereas approach we present in Chapter 4 relies on the identification of a stop where no GPS signal have been detected, then it infers the visited POI and consequently derives the user activity.

A different approach is [61]. Here the focus is not on the analysis of the single user but an aggregate vision of regions. Trajectories, POIs and road networks are combined to define functional regions. The result is a set of regions represented by a distribution of topics (or functions), where a topic is a POI category. With this work authors aim to help people to easily understand the complexity of a metropolitan area. The results are applied to different fields, such as urban planning, location choosing for business, advertisement casting and social recommendations.

In the same direction there is the work of [60], where authors propose an activity discovery method for moving objects based on collaborative filtering. After having generated the objects activity sequence and abstract the corresponding features, they build a three dimensional matrix (object identification, hot region, and sequence). On the basis of this matrix they compute objects interest degree to each hot region and generate an object-region interest degree matrix. Combined with collaborative filtering, similar objects can be queried and their common interesting activities can be found. Furthermore, they propose a recommendation system based on the K-nearest neighbor algorithm by which a series of potential interesting activities can be recommended to the moving objects.

1.3 Mobile Phone Data Knowledge Extraction

Mobile phone is the main character of this Big Data era. The penetration of such device have continously increased during the past decade, delineating a trend that is still running: by 2020, 90 percent of the world’s population over 6 years old will have a mobile phone. During every moment of people’s life, a mobile phone is present. Mobility is not the only feature of human behavior that can be derived from mobile phone data, even though is the most fascinating one.

Information about users phone activity is a side product of GSM protocol and can
Figure 1.1: Comparison between an user actual trajectory (dashed line) and CDR generated by his/her calling activity. It is only possible to derive the presence of the user over the spatial region A,B,C, i.e. the area covered by the tower the phone was connected to during the call and SMS he/her performed. Sometimes also a LAC Update is available: it represents a Location Area Change, i.e. phone is connected to a tower that is in a different area w.r.t. the previous tower it was connected to.

be collected in two modalities: on Network-side modality the provider records such data for specific billing purposes; in the Handset-side modality the data are collected by phone sensors (GPS, WiFi, Bluetooth,...). So far, research works in this field use to identify Mobile Phone Data as the Network-side Data. The main element of this data type is the Call Detail Record (CDR): it represents an user event that must be billed, i.e. voice calls, SMS, MMS and so on. Mobility knowledge contained in CDR Data has some particular weaknesses to deal with: the completeness in terms of space and time is not guaranteed. While GPS data can be properly resumed by means of trajectories, this is not possible with CDR Data. The position of an user over space and time is known only when he/her is calling or sending SMS. This lead to a coarse mobility data, not representable in terms of user trajectories. Moreover, the spatial granularity is defined at a GSM-tower level: for each event, a CDR contains the GSM tower where the phone was connected to. Since the coverage of a tower is in the order of square kilometers, even the concept of georeferenced point related to a billing event is not proper. Even though these lacks of quality (see Fig. 1.1), by means of Call Detail Records (CDR) Mobile operators are collecting the most precious source of mobility data, since it is covering the globality of world’s population.

Although the different spatio-temporal quality of CDR, there are plenty of works about mobility based on Mobile Phone Data. In [34], CDR regarding an one year long (October 2008-September 2009) user activities have been used to observe traffic pattern on the road connecting the two main city of Estonia. In this context,
authors built a traffic monitor based on mobile phone activities from half a billion clients: the potential of this research field relies exactly on the high number of users involved. Furthermore, the availability of data from a wide time range lead to a precise users profiling. This improve the quality of traffic monitoring by highlighting, for instance, the composition of traffic in terms of users type: residents, commuters or visitors. Another approach in this direction is the one from [11]; here, authors provide techniques to extract mobility patterns from mobile phone traces. Moreover, a valuable comparison between mobile-phone-based and odometer-based mobility measures has been performed, in order to validate and analyze the quality of the results obtained. Both of the works introduced rely on users profiling, that is a task useful to classify users according to their visited location. The purpose of such process is to further enrich the mobility analyses by extending the semantic of detected patterns, i.e. to distinct between patterns of residents and commuters w.r.t. to a selected geographical area. Users profiling is one of the task accomplished in [39]: CDR are used to compute the two most visited location for each user, inferring those locations as Home and Work. This result is further extendend and refined in [25], where a framework to detect residents, commuters and visitors is provided.
Part II

Data integration: boosting the power of data
Every kind of mobility data has its type and its meaning in terms of knowledge. Each one of them express a different dimension, enclosing a particular characteristic of every-day life activity; the idea leading this thesis is to exploit this multi-dimensionality to better understand the big picture of human mobility. The combination of such dimensions is still a demanding challenge, either for methodological and structural reason. The importance of such challenge is clear if we consider how many kinds of Mobility Data we produce with our most used device; in fact, a simple smartphone has lot of sensors: GPS, accelerometer, Wifi, and, of course, GSM antenna. All of them are proxies for a specific type of mobility data.

When a GPS device records a point (this process is called fix), it is able to get data with features such as \((latitude, longitude, heading, timestamp, \text{instant speed})\). When a call starts from a phone, the event is recorded by the GSM tower the phone was connected to, thus identifying a wide area the user visited at a certain timestamp. While GPS points are the most precise representation of a movement across space and time, GSM data have a higher pervasiveness but a rougher space precision. However, all of these data types do not provide any other semantic layer. A geotagged tweet, conversely, contain a text associated to the place the user was while writing. On Fourquare, users check in and specify the place they are both in terms of spatial reference and Place of Interest visited, and so on. Manifold are the example of semantic associated to spatio-temporal data; the data revolution happening nowadays is owing to spatio-temporal data the possibility to localize in space and time many events that represent different aspects of our lives.

Following, contributions to the data integration challenge - developed during the course of this Ph.D. - are described.
Exploiting GPS Data to assess Road Network Travel times

The assessment and consideration of traffic conditions are the keys for developing intelligent transportation services, such as, route planning, traffic flow analysis, route path discovery, to name a few. In this context, the time taken by a vehicle to traverse a road segment can vary depending on the time of the day and, more critically, road segments can even be unavailable during certain time intervals. For example, to compute the fastest route between two points within a time-dependent network, we take into account the departure time, since road traversal time may vary along time. Thus, key methods for intelligent transportation service, such as routing, shortest path, $K$NN, should be revisited and adapted according to this new constraint. The proliferation of GPS-enabled devices is allowing the production of a huge amount of location trajectory data. However, trajectory data collected from GPS devices suffers from two problems: (i) low sampling rate data (due to aggregations executed by the device to save communications with the base station), and (ii) error in GPS observations, which imply that most detailed information about the exact movement of the object is lost and great uncertainty arises in their routes. Clearly, this kind of uncertainty severely affects the effectiveness and efficiency of underlying methods such as, indexing, querying and mining. The proposed method allows to compute average speed for each segment of the road network in distinct time periods, providing a comprehensive view of the traffic conditions.
2. Exploiting GPS data to assess road network travel times

2.1 Problem definition

The aim of this work is to find a methodology to compute the real speed of each segment in a given road network, using the observation (i.e. points) coming from GPS devices. By definition, such observations are affected by errors. Then, from the speed computed, generate travel-time functions to compose a time-dependent network. Several methods in literature consider a Gaussian distribution of the GPS error; this assumption could be not true and therefore an innovative way to consider the problem is needed.

Below we provide a formal description regarding the speed estimation for a road segment.

Definition 1. Given a set of observations $O = \{o_1 \ldots o_m\}$ where $o_i = (p_i, d_i, s_i)$ has its spatial position $p_i$, its direction $d_i$ and its speed $s_i$, a set of road segments $R = \{r_1 \ldots r_n\}$ and having an function $\sigma(o_i, R) = (w(o_i, r_j), r_j)$ assigning the observation $o_i$ to the segment $r_j \in R$ with a confidence value $w(o_i, r_j)$, it is possible to estimate the speed over the segment as:

$$\text{Speed}(O, R, r_j) = \frac{\sum_{o_i \in O_j} w(o_i, r_j) \cdot o_i \cdot s}{\sum_{o_i \in O_j} w(o_i, r_j)}$$

Where $O_j = \{o_i | \sigma(o_i, R) = (w(o_i, r_j), r_j)\}$.

Following, we define a method to estimate $\sigma(o_i, R)$ function without any assumption on error distribution.

2.1.1 The gravity model

Having the set of observations $O$ and the set of segments $R$ we define the attraction of a segment $j$ for a point $i$ as: $w^{d}(o_i, r_j) = w^{\theta}(o_i, r_j)$ where

- $w^{d}(o_i, r_j) = 1 - \frac{\text{dist}(o_i, r_j)}{\sum_{r_k \in R} \text{dist}(o_i, r_k)}$;
- $w^{\theta}(o_i, r_j) = 1 - \frac{\text{ang}(o_i, r_j)}{\sum_{r_k \in R} \text{ang}(o_i, r_k)}$;

Furthermore, $\text{dist}$ is the Euclidean distance between a point and a segment and $\text{ang}$ is the absolute difference between the direction of the point and the direction of the segment. The direction is measured in degrees, where 0 degrees indicates north direction and 180 degrees is south directions. For the every observation, this value comes directly from GPS device. Therefore the force of attraction of a segment over a point is defined by the combination of these two dimensions as:
Definition 2. Having an observation $o_i \in O$ and a segments $r_j \in R$ the gravity force function $GF(o_i, r_j) = w^d_{(o_i,r_j)} \times w^\theta_{(o_i,r_j)}$. Finally the point is assigned to segment with the most powerful force: $\sigma(o_i, R) = \text{argmax}_{r_j \in R}(GF(o_i, r_j))$.

This methodology will be referred as Gravity Model and we consider to have a function $GMassign_{O,R}(o_i) = \langle o_i, \sigma(o_i, R), w(o_i, \sigma(o_i, R)) \rangle$ which returns, in other words, the triple $t \in \text{assignments}$ where $t.o = o_i$.

To better understand the idea under the definitions, we present a complete example (Figure 2.1) of how the forces are computed and how the points are attached to segments. In Figure 2.1 a sample of points and segments is shown, the points are attracted by all the segments with different forces and suddenly they fall over one of them. For example the point $P_1$ undergoes the following forces:

- $GF^d_{(o_1,r_{ab})} = (1 - 10/100)(1 - 39/200) = 0.9 \times 0.305 = 0.2745$
- $GF^d_{(o_1,r_{bc})} = (1 - 10/100)(1 - 4/200) = 0.9 \times 0.98 = 0.882$
- $GF^d_{(o_1,r_{cd})} = (1 - 28/100)(1 - 66/200) = 0.62 \times 0.67 = 0.4154$
- $GF^d_{(o_1,r_{ce})} = (1 - 28/100)(1 - 2/200) = 0.62 \times 0.99 = 0.6138$
- $GF^d_{(o_1,r_{hc})} = (1 - 8/100)(1 - 46/200) = 0.92 \times 0.77 = 0.7084$
- $GF^d_{(o_1,r_{hf})} = (1 - 9/100)(1 - 2/200) = 0.92 \times 0.99 = 0.9108$
- $GF^d_{(o_1,r_{gh})} = (1 - 9/100)(1 - 41/200) = 0.99 \times 0.795 = 0.78705$

Therefore the point is attracted mostly by the segment $hf$ or more formally $\sigma(o_i, R) = r_{hf}$.
2. EXPLOITING GPS DATA TO ASSESS ROAD NETWORK TRAVEL TIMES

2.1.2 Candidate selection

The Gravity Model is applicable over the whole road network, but in real applications the set of segments to be considered is too large. For this reason we present two different methods to select the set of candidate segments. The first one, already used in previous works in literature, uses a buffer around the points selecting only the segments with a distance which is lower than a certain threshold. Instead, the second method uses the concept of nearest neighbor, taking only the first k segments near the point.

2.1.3 Buffer method

All the most used approaches for the assignment of a segment to a GPS fix are depending from a search buffer: the segment, or the candidates, are initially chosen from all the segments contained in a circumference of radius $\epsilon$. This parameter $\epsilon$ is very application dependent and usually can not be derived from the real observations, because the ground truth is not available, and affect greatly the result of the point-segment process. Moreover using a buffer the error distribution is assumed to be normal which, as already said, is a strong assumption. For example the presence of buildings affects the precision of the GPS and therefore all the observation can be affected to a bias. In Figure 2.2 (on the left) two examples of road network is shown, here the buffer method selects a number of candidates which can greatly vary in different context (e.g. urban or sub-urban areas).

Figure 2.2: Two examples of road network and the candidate selection using the two different method: a buffer of 30m (left) and the 3 nearest neighbors (left).

Another effect of this method for the candidate selection is the fact that some points which has no segments inside the buffer are discarded, this leads to a reduction of the set of observation used.
2.2 FROM SPEED ESTIMATION TO TRAVEL TIME

2.1.4 K-Neighbor method

Instead of considering a fixed buffer with the limitation described before, we defined an innovative way to select the candidates choosing the $k$ nearest neighbors of a point. Here the only parameter needed is $k$ and we will show in the validation section how this affects the results increasing the accuracy compared to the classical buffer method. In Figure 2.2 it is possible to see another advantage of this method: in fact $k$ is the exact number of segments considered for every point-segment matching leading to a controlled complexity in time and space of the algorithm.

2.2 From speed estimation to travel time

The structure of a time-dependent network can be modeled by a TDG (a time-dependent graph) where the vertices represent the network junctions, starting and ending points of a road segment (e.g. a street or an avenue) and the edges connect vertices (depending on the application, additional points can represent a change in curvature or in maximum speed of a segment). The cost (time) to traverse an edge is a function of the departure time. A TDG $G = (V, E, C)$ is a graph where: (i) $V = \{v_1, \ldots, v_n\}$ is a set of vertices; (ii) $E = \{(v_i, v_j) \mid v_i, v_j \in V, i \neq j\}$ is a set of edges; (iii) $C = \{c_{(v_i, v_j)}(\cdot) \mid (v_i, v_j) \in E\}$, where $c_{(v_i, v_j)} : [0, T] \rightarrow R^+$ is a function which attributes a positive weight for $(v_i, v_j)$ depending on a time instant $t \in [0, T]$ and where $T$ is a domain-dependent time length. In other words, a TDG represents a road network with the traffic information about it including as the weight of the edges. In many cases the traffic information is not directly available or not really reliable, but the structure of the road network and a trajectory dataset are available. In following, we show how the speed estimation, using the gravitational model, gives us the support to build the functions in the set $C$ of a TDG from the available trajectory points. More specifically, we describe how to convert the average speed of a segment of sequence of time intervals, in a travel-time function that is piece-wise linear. We choose piece-wise linear functions because it is easy to check if these functions attend the FIFO property. We call this algorithm GenerateTravelTime: given a road segment and a timestamp, it returns the travel time of the segment, calculated using the average speed of that segment at that time. Furthermore, if we use the average travel-time for each time partition, we create a piecewise constant travel-time function, that means the function does not attend the FIFO property in all decreasing pieces. Figure 2.3 (left) shows a function obtained by two steps: the first (left) approximating the various pieces and the second (right) which ”smoothes” the function using a linear interpolation for the middle points. Having the function for each edge, methods as [42] are able to find the best routes considering the
dynamic travel time. We call a time-dependent network a road network which has a time-dependent function as the weight of its segments, instead of a constant value. To exemplify the fastest path problem in a TDG, consider that someone wants to go from $a$ to $d$ at $16h$. The best choice in this case, i.e. the path that has the lowest cost, is $<a,b,c,d>$. Travel from $a$ to $b$ takes 15 minutes, to go from $b$ to $c$ at $16h15$ takes 15 minutes and to go from $c$ to $d$ at $16h30$ takes 20 minutes, then the path $<a,b,c,d>$ takes 50 minutes. Consider now a departure time $t_s = 10h$, a path $<a,b,c,d>$ takes 70 minutes and a path $<a,c,d>$ takes 45 minutes. Thus, the fastest path at $10h$ is $<a,b,c,d>$, but at $16h$ the best choice is to follow the path $<a,c,d>$.

![Figure 2.3](image1.png)

**Figure 2.3:** On the left, an example of a function build in the first step with the travel-time averages. On the right, a result of the transformation applied in the second step.

![Figure 2.4](image2.png)

**Figure 2.4:** The result of the speed estimation for a the same portion of the road network at different time of the day: 4am, 6am, 10am and 1pm. The light green represents high speed and the red represents a low speed.

### 2.3 Validation

In this section we will validate and evaluate the proposed Gravity Model with the two candidate selection methods in order to study how the accuracy changes. As
already said the ground truth is not available in real data, for this reason we created a synthetic dataset using an up-to-date simulator and perturbing the resulting dataset. In this way for each simulated observation we have the real segment which generated it. In the following we first describe the way we produced the data and then a complete study of the methods results will be presented.

### 2.3.1 Generation of synthetic observations dataset

In order to validate our method we generated a synthetic dataset of observations. We used the SUMO simulator [8] for generating a synthetic dataset of trajectories. We used, as input, the road network of Pisa and surrounding area extracted from Open Street Map. In order to replicate the same condition we have in the real dataset (as we show in Section 2.5), we have used a set of parameters extracted from previous studies published on the same dataset [26] such as the sampling rate between 30 and 90 seconds. We generated the observations of 2000 vehicles moving on the network for a day obtaining 314785 points perfectly positioned on the road network. To simulate the error of the GPS devices, we added some noise at each observation: (i) the spatial component of the observation is moved by a random value considering a normal error distribution with a variance of 45 meters; (ii) the orientation is changed by a random value considering a normal error distribution with a variance of 90 degree. It is important to notice that these values are extracted by an empirical study over the real dataset which does not contains any information about that, therefore heuristics were applied in order to estimate these errors. Moreover, even if our methodology does not need to consider a normal distribution of the error, we decided to apply it to the synthetic data because we do not have information about buildings or other conditions interfering with the GPS signal.

### 2.4 Performances study

The simulated observations allow us to validate our methodology knowing for each point the segment which generated it. We used two measures in order to evaluate the methodology: precision and accuracy:

\[
\text{precision}_{O,R} = \frac{|\{o_i|\text{GAssign}_{O,R}(o_i) = \text{Belong}(o_i)\}|}{|O|}
\]

where \(\text{Belong}\) is the real information coming from the simulated data.

\[
\text{error rate}_{O,R}(r_j) = \frac{\text{Speed}(O, R, r_j)}{\text{Real Speed}(r_j)}
\]
where \( \text{Speed}(O, R, r_j) \) is estimated using the speed function obtained from the \texttt{GenerateTravelTime} algorithm and \( \text{RealSpeed}(r_j) \) is the speed used in the simulation for the segment \( r_j \).

Analyzing how the precision varies using different parameters for the two candidate selection methods, we obtained the results shown in the following table:

<table>
<thead>
<tr>
<th>Range buffer</th>
<th>precision</th>
<th>Nearest k</th>
<th>precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>10mt</td>
<td>15.2%</td>
<td>k 1</td>
<td>69.0%</td>
</tr>
<tr>
<td>20mt</td>
<td>29.7%</td>
<td>k 2</td>
<td>75.6%</td>
</tr>
<tr>
<td>30mt</td>
<td>42.4%</td>
<td>k 3</td>
<td>77.5%</td>
</tr>
<tr>
<td>40mt</td>
<td>72.0%</td>
<td>k 4</td>
<td>78.6%</td>
</tr>
<tr>
<td>50mt</td>
<td>74.8%</td>
<td>k 5</td>
<td>79.1%</td>
</tr>
<tr>
<td>60mt</td>
<td>75.2%</td>
<td>k 6</td>
<td>79.3%</td>
</tr>
<tr>
<td>70mt</td>
<td>75.9%</td>
<td>k 7</td>
<td>79.4%</td>
</tr>
<tr>
<td>80mt</td>
<td>76.0%</td>
<td>k 8</td>
<td>79.4%</td>
</tr>
</tbody>
</table>

Table 2.1: Precision comparison

Table 2.2: Comparison between precision obtained with range-buffer search and Nearest k search.

As expected the buffer method converge to a good result approaching the variance used in the simulation (45mt) but this put in evidence the fact that without a good estimation of the possible error it is difficult to obtain such results, and as described before this information is usually neither reliable or unverifiable. On the other hand the KNN method got better result since the beginning due the fact that it is more adaptive to different context, here the method obtains a precision which converges rapidly to a value of 79.4% highlighting the fact that it adapts better to every context in the road network. Even though this is not the most important result to us, it is a value useful for a comparison with other work in this field; the online algorithms like [40] have a really high precision (over 90%) when used with sampling rate 1 Hz and the measurement error is low, but with low sampling data the precision decrease significantly: with our same values of sampling and error variance, the precision of the algorithm mentioned is around 50%. Moreover selecting a value of \( k \) we select exactly that number of candidates without losing the control of the performances as it happens using the buffer method. We can see in Figure 2.5 the cumulative distribution of the KNN method error percentage, varying the parameter \( k \). This plot shows how 77% (8500) of the segments has an error in the speed estimation under 20%. As we saw in the precision comparison (2.2), the error rate
2.5. REAL CASE STUDY

In this section we present the results of our method applied to a real dataset of GPS observations of 30000 real car users in Tuscany in a time period of 5 weeks covering different kind of territories such as urban and suburban areas. This is a sample of data obtained by a private company employed specifically as a service for insurance companies and other clients called Octo Telematics. We processed this dataset of observations (Fig.2.8(left)) using the presented algorithms and in the following we will show the results obtained. As described above the GMatch is applied using the

Figure 2.5: Performance of Gmatch for different value of k

converge giving strenght to the approach. Another interesting aspect is that the error is usually on the low speed roads, therefore the average speed estimation error in absolute terms is 6.1km/h, and considering the average speed of all the network segments of 58.32 km/h, it is only a an average error of 10.4%: this is the main result of our work, since our goal is to estimate the speed of each road segment. Comparing the error_rate of the buffer method using 80mt and the KNN method with k=7 we can see in Figure 2.6 that the results obtained are better also in the evaluation of the speed.

Therefore we can conclude that the Gravity Model not only is better using the KNN candidates selection but it is very simple to use overcoming the problem of knowing or estimating the error variance of the application scenario.

2.5 Real case study

In this section we present the results of our method applied to a real dataset of GPS observations of 30000 real car users in Tuscany in a time period of 5 weeks covering different kind of territories such as urban and suburban areas. This is a sample of data obtained by a private company employed specifically as a service for insurance companies and other clients called Octo Telematics. We processed this dataset of observations (Fig.2.8(left)) using the presented algorithms and in the following we will show the results obtained. As described above the GMatch is applied using the
2. EXPLOITING GPS DATA TO ASSESS ROAD NETWORK TRAVEL TIMES

Figure 2.6: Comparison between precisions obtained by using the KNN method with \( k = 7 \) and the buffer method using a threshold of 80mt.

KNN candidate selections setting \( k = 7 \) and dividing the dataset into sub-datasets considered one hour time intervals. Then, applying the GenerateTravelTime, we build a function for each segment extrapolating the speed discovered in each interval. The result for each road is shown in Figure 2.7 where the travel time changes in time, in this case looking to a specific road Strada del Monte Serra the time needed to traverse it is lower in the morning and higher in evening, especially if the departure time is around 8:00pm.

In Figure 2.4 we show another real example of how the speed changes in time on a set of road segments in order to show how the right assignment lead to a consistent situation: here during the early morning at 4:00am the speeds are higher (light green) especially on the highway located at the bottom of each figure, than going on with the time, the speed over the segments becomes slower and slower (from green to red) reaching the 1:00pm where the traffic decrease the speed greatly on the main roads: the highway and the roads directly connected with its entrance/exit. This represent our best result in fact it can be done thanks to the functions computed over the real observation which are attached to the right segments as shown in the validation section.

Moreover in Figure 2.8(right), in order to assure the correctness of the result, we compared the distribution in time of the number of cars with the average speed on the network. It is important to see how the average speed increase when the number
2.6 Summary

In this chapter we proposed a gravity model method for computing road segment average speed from trajectory data. In sequel, we show how to generate travel-time functions from the computed average speeds. Our approach allows extracting a set of speed functions, which can represent precisely the traffic conditions in time and space, without considering strict assumptions on error distribution. The proposed method accuracy is validated using a synthetic and a real dataset showing that it leads to a more realistic results. We shown also how the results obtained give the possibility to improve existing methods for routing, KNN, and shortest-path algorithms which take advantage of the time-dependent network.

Figure 2.7: An example of real travel-time functions. On the left, travel time generated using speed estimation from Strada del Monte Serra, Pisa, in the morning. On the right, travel time from speed estimation from the same road at night.

Figure 2.8: The dataset used (left) and the average speed of the network compared to the number of active cars during the day (right).
2. EXPLOITING GPS DATA TO ASSESS ROAD NETWORK TRAVEL TIMES
A Time Aware Map Matching Method to enrich Raw Spatio-Temporal Trajectories

The widespread diffusion of location devices for personal usage, from GPS navigators to location-based services on smartphones, are making this decade the era of Geo-Spatial Data. Coupled with the novel technologies for storing and processing large streams of data, this phenomenon is leading to the collection of massive datasets of GPS (or GPS-like) traces describing the movement of people and vehicles, as well as to the development of analysis methods and applications that use such information to extract useful knowledge. Some examples are provided by the current studies on knowledge discovery from spatio-temporal data, based on methods like trajectory pattern mining [27] or flock mining [41]. These approaches rely only on spatio-temporal features of raw data without considering any geographical characteristic, such as the features of road network.

In this context, map matching, i.e. the process of associating a sequence of GPS points to a connected sequence of road segments, gives us the chance to enrich raw data with the semantic layer provided by the road map and all contextual information associated to it, e.g. the presence of speed limits, attraction points, changes in elevation, etc.

Although it is a classical and well known task in GIS literature, the map matching problem still represents an important and a valuable challenge. The map matching problem can be treated at two different scales, depending on the characteristics of input data, which can be made of either high-frequency or low-frequency samples of the real position and movement of the device. The former is mainly treated in the field of Personal Navigation Assistants, where the device is able to identify in real time the road where the user is traveling. The latter is common for applications
dealing with smartphones or GPS-equipped black boxes installed vehicles for security or insurance purposes. This kind of devices sample and store their location at a lower frequency to limit the battery consumption (e.g., with smartphones) or to reduce the traffic of data between the device and the server that stores the information. The result is a coarse-grained GPS data, harder to deal with but still with high value: this data represents the most reliable proxy for road network mobility. One important issue introduced with low-frequency samples is path reconstruction. After mapping single points to the road network, between two consecutive locations there might be a significant gap, therefore requiring strategies to reconstruct the path traversed by the vehicle or the individual.

With this work we present a significant improvement on the state-of-art of map matching for low-frequency samples, by considering two aspects that were neglected in previous literature: first, a data-driven estimation of traversal times of road segments is introduced and exploited in the evaluation of map matching alternatives; second, we perform a shift of perspective in the path reconstruction phase and remove the most common assumption adopted in literature: the shortest/fastest the better.

**Inferring and exploiting segment traversal times.** Surprisingly enough, virtually all the literature on map matching reasons in terms of length of the alternative paths, and not in terms of time requested to traverse them – which instead is the obvious target of personal navigation systems, for instance. Part of this phenomenon can be explained by the general lack of reliable information about travel times on road networks, which greatly compromises the applicability of traversal time-based methods in practice. In this work, we propose to fill in the gap by exploiting the information we can infer from the same GPS data we want to match: either the instantaneous speed, where available, or estimates derived from trip length and the timestamps attached to the points. Thus, the path reconstruction heuristics can exploit such estimates to provide an evaluation of traversal time for each alternative path.

**Shortest/fastest path: a questionable assumption.** Most part of the literature on map matching assumes that the most likely path connecting two consecutive points in a trajectory is also the shortest or the fastest. Clearly, that is inspired by the fact that real trips are just means to reach a destination B from a starting location A, without any objective other than reaching the destination in the most efficient way. What if this seemingly obvious assumption could be violated in practice? That would mean that, for some reason, there are trips that last longer than the minimum possible, and therefore any map matching method that looks for the shortest or fastest path would return shorter times than reality. Since typical GPS traces also contain an accurate temporal information – most often neglected by map
matching methods – we can actually check whether this happens or not. Figure 3.2 reports such an experiment; a shortest path method is applied to a real dataset of trajectories described in Section 3.7, and the travel time according to the algorithm is compared against the real one, obtained from GPS timestamps. It is clear that when the travel times become significant, larger than one minute, the reconstructed trips tend to be much faster than the real ones.

We propose an effective Time-Aware Map matching process for low-sampling rate GPS data based on the reduction of temporal mismatch introduced above. Fig. 3.1 provides the general workflow of Time-Aware map matching. With the initial and independent point-to-matching task we obtain a road network enriched with a precise time-dependent estimation of travel times (see upper part of the figure), based on the method proposed in 2. The core of our work is the second phase (lower part): a Time-Aware map matching algorithm that uses travel times to transform an input raw GPS trajectory -with travel time $t$- into a sequence of road segments with a travel time $t' \sim t$.

In particular, we focus on finding the path between consecutive points that better fit the real travel time. Fig.3.3 shows the idea that guided us towards the development of this new approach. The raw GPS trajectory composed by points $a$ and $b$ has a travel time of 78 seconds. Once matched $a$ and $b$ to the corresponding road segments, thus obtaining source and destination of the path, there are two options: shortest path is also the fastest, with a travel time 60 seconds. An alternative path, there called Time-Aware, would be more reasonable to select, since it has a more similar travel times (72 seconds). The main contributions of this chapter can be
summarized as follows:

- a methodology for inferring speeds and traversal times of road segments is applied, based on the principles introduced in [18];

- a novel time-aware map matching method is proposed, that takes into consideration the real traversal times as described in the raw GPS data. A proof of the complexity of method is also provided, showing its higher scalability compared with existing competitors;

- a new methodology for evaluating the performances of map matching over large datasets, named *middle point test*, is introduced and adopted;

- a wide comparison against the state-of-art competitors is performed, based on three real datasets: a small one from SigSpatial Cup 2012 and two large ones describing, respectively, the movements of taxis in San Francisco and private vehicles in Tuscany, Italy.

The outline of the chapter is as follows. Section 2 presents a survey of related works in the field of low-sampling rate map matching, while section 3 contains the definition of our proposed algorithm. Section 4 is dedicated to the validation of the algorithm, while in section 5 all the experiments on our dataset are reported. Section 6 gives the conclusion and introduces real applications and ideas for future works.

![Figure 3.2: Average travel times of reconstructed path according to the GPS travel time of the original points. The area highlight the relative standard deviation. $\overline{\Delta}$ indicates the average difference between Path and GPS travel times.](image-url)
3.1 Time-Aware Map Matching

A map matching process is based on two main steps: a point-to-segment matching process and an heuristic process to choose the path between the possible candidates. These tasks are modeled according to the data type the map matching is designed for. The two steps are well separated, as highlighted in Fig. 3.1: first of all, we match the gps points to a road segment, then we reconstruct the path between every two consecutive matched segments with a Time-Aware heuristic.

All the state-of-the-art methods rely on what we called the “shortest path” assumption. As described before, in literature there are different approaches to map matching founded on a large variety of heuristics. The common point for all of them is the use of the shortest path to connect two consecutive matched GPS points. The underlying assumption is: a driver is always choosing the shortest path. This strong assumption could not be true. We released this assumption by introducing our Time-Aware heuristic: the path between two matched GPS points is the most compatible with the real travel time. In Fig. 3.2 the difference between real GPS time and path time of shortest-path reconstructed trajectories is highlighted. The curve represents the average shortest path time $y$ for each pair of consecutive GPS points with GPS time $x$. The line indicates the optimal case, when the shortest path travel time is equal to real GPS time. The error bars provide an estimation of the error that affect this heuristic. For reading purpose, error bars are related to aggregated bins. As showed, for our 1M+ trajectories dataset the average difference between real time and shortest path time is more than 120%. The goal of our work is to develop a new map-matching algorithm able to minimize the difference between selected path travel time and real time, then we show how this minimization improves the results in terms of accuracy.

3.2 Point-to-segment matching

The first phase of our map-matching algorithm considers each point separately, matching it to a segment in the road network. For this task, we recall the method introduced in Sec. 2.

**Definition 3.** Given a location point $p$ and a road network $N = (V, E)$, with $V$ the set of vertexes and $E$ the set of segments, we define the point-to-segment matching of point $p$ as the process of associating $p$ to a segment $e \in E$.

The main issue of point-to-segment matching is represented by the localization error of each point. Such error is variable from few meters to tens of meters, depending on weather conditions and satellites visibility. The state-of-the art map-matching
methods assume that the localization error follows a Gaussian distribution, with fixed mean and standard deviation. This is a really strong assumption that could affect the results: on our dataset we saw that error varies depending on different factors, e.g. in the city center, the presence of big buildings affects the precision of localization. A side effect of the point-to-segment matching is that, where available, any contextual information associated to the each point can be transferred to the corresponding road segments, thus enriching the existing background knowledge on the road network. The basic application, also exploited in this work, takes advantage of the instantaneous speed of vehicles – associated to each GPS point – to estimate real speeds (and therefore travel times) over the road segments. In our work, the gravity model have been applied both for the point-to-segment matching for each trajectory and to estimate travel times for each segment of the road network. To the best of our knowledge, this is the more accurate method to estimate the typical speed of road segments.

In order to prevent any assumption of that kind, in this work we adopt the gravity model proposed in [18].

To speed up the calculation, the candidate segments are the $k$ nearest segments

Figure 3.3: An example from our real dataset on the problem we faced: given two GPS points $a$ and $b$ with their relative timestamp, we search for the path that mostly fit with the travel-time of input points. As depicted, in this case the shortest path is too fast, suggesting that the user was traveling along a different path.
for $p$. Results in [18] show that $k = 8$ is a good compromise.

3.3 Path reconstruction heuristics

The state-of-art methods for map matching on low sampling rate data rely on the “shortest-path” assumption: the most likely path between two consecutive matched points is always the shortest one. That clearly stems from various assumptions, such as the fact that the trip performed has the unique objective of reaching the destination, without any other goal – e.g., traversing more pleasant roads, performing quick bring-and-get tasks, avoiding crowded roads (with or without traffic jams) – and/or the fact that drivers really know what is the most efficient path. In our work we release these assumptions, aiming to realize a more realistic heuristic.

Exploiting the results in [18], each road segment is associated with its estimated travel time, with an error of $10\%$. In the following, we leverage this information to built a time-aware heuristics that associates each pair of points in a user’s trip to the path that best fits with its travel time.

The problem of finding a set of road segments whose travel time fit with the travel time of two GPS points reminds the knapsack problem, which is NP-hard and solvable with a linear programming approach. Yet, in our case we have additional constraints and requirements. First of all, the chosen road segments have to be connected to each other, in order to form a path in the road network. Furthermore, the travel time compatibility requirement alone might be not sufficient to obtain reasonable path reconstructions. For instance, if the real path is slower than expected (w.r.t. estimated travel times over segments), using only the travel time difference as optimization criterion could lead to odd results: in order to reach the optimum, wrong segments might be added, only to create artificial detours that increase the overall travel times to better fit the real one. In order to take care of such extreme behaviours as well as to provide computationally sustainable solutions, we propose an heuristics based on a routing approach: indeed, routing algorithms take implicitly account of road network topology, and their complexity is lower than linear programming approaches.

The method we propose is based on Dijkstra’s shortest path algorithm, using a time-aware heuristic to evaluate the cost of every road segment according on how it fits with the trajectory’s real travel time. The core of our approach is the introduction of a $\text{Timecost}$ function: given a source and a target node, Dijkstra algorithm uses $\text{Timecost}$ to evaluate the cost of every segment examined to find the
shortest path. Hence, we obtain a path that is the shortest in terms of timecost. We can define the solution we found as a path with two constraints: (i) acyclicity and (ii) highest similarity line speed w.r.t. the real line speed. The acyclicity of the solution is guaranteed by Dijkstra algorithm, while the similarity with the real line speed is the result of Timecost minimization. In other words, the solution found is the path with the more similar travel time w.r.t. to the real GPS travel time according to network and speed constraints. Below we provide the details about how Timecost is computed and prove how the choice of the less expensive path according to Timecost finds a solution for the problem we tackled, yet satisfying the constraints we introduced. More formally, we prove that the minimization of timecost yield the more similar path in the road network according to real GPS line speed. As a result, path travel time will approximate the real travel time, yet respecting network constraints.

In fig. 3.4 there is a graphical representation of Timecost computation, while in fig. 1 Timecost is analytically showed through pseudo code; input linespeed is defined as the ratio between the distance between the two consecutive GPS points a and b and the related travel time, i.e. the time difference between b and a timestamps. heading indicates the angle of the straight line between a and b w.r.t. north heading. The timecost of a segment is then obtained by projecting its length onto the straight line defined by a and b. Expected length (exp_length) is the length of the segment if its projected speed would be equal to the real line speed of the car. Projected speed defines the relation between the typical speed of a segment and its projection onto the straight line between a and b.

**Definition 4.** We define the timecost for a road segment r as the difference between the length of r and the expected length of r according to the straight line speed of the vehicle.

**Theorem 1.** Let p be a path in the road network connecting two nodes a and b such that \( Tc(p) = \arg\min_{p \in P} Tc(p) \), with Tc as the timecost function and P the set of all the possible paths connecting a and b. For each \( p' \neq p \) we have:

\[
Tc(p) < Tc(p') \implies |r.speed - p.ls| \leq |r.speed - p'.ls|
\]

where r.speed is the ratio between the straight line distance between a and b and the travel time recorded by GPS device (car linespeed) and p.ls is the ratio between the straight line distance between a and b and p.traveltime (path linespeed).

**Proof.** Suppose on the contrary there exists a path \( p'' \) such that

\[
Tc(p) < Tc(p'') \implies |r.speed - p.ls| < |r.speed - p''.ls|
\]
3.3. PATH RECONSTRUCTION HEURISTICS

Figure 3.4: The computation of timecost for segment $r$: the length of segment is projected onto the straight line trajectory between GPS points $a$ and $b$, then the timecost is computed as the difference between the length and the supposed length of the segment, according to the straight line speed of the car.

For the sake of simplicity, we will assume $p''$ identical to $p$ except for the total travel time, and, accordingly, their $l_{speed}$. Then, for the definition of Timecost function (fig. 3.4), we have

$$|p.\text{exp length} - p.\text{length}| < |s''.\text{exp length} - p''.\text{length}|$$

$$\implies |r.\text{speed} - p.\text{lspeed}| > |r.\text{speed} - p''.\text{lspeed}|$$

Since $s.\text{length} = s''.\text{length}$, we can simplify:

$$s.\text{exp length} < s''.\text{exp length} \implies p.\text{lspeed} < p''.\text{lspeed}$$

Recalling the definition of $p.\text{lspeed}$ we have then:

$$s.\text{exp length} < s''.\text{exp length}$$

$$\implies \frac{1}{p.\text{traveltime}} < \frac{1}{s''.\text{traveltime}}$$

Since $\text{length}_{\text{expected}}$ is defined as $\frac{\text{length}(e) \times \text{time}_{\text{expected}}}{\text{time}(e)}$ (see Fig. 1) and given our initial assumptions on $p$ and $p''$, the expression become

Given our initial assumption and according to Timecost computation, we know that $s.\text{proj speed} = s''.\text{proj speed}$. So, simplifying again

$$p.\text{traveltime} < p''.\text{traveltime} \implies p.\text{traveltime} > p''.\text{traveltime}$$
A TIME AWARE MAP MATCHING METHOD TO ENRICH RAW SPATIO-TEMPORAL TRAJECTORIES

that is impossible.

Algorithm 1: Pseudocode for Timecost computing

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\alpha =</td>
</tr>
<tr>
<td>2</td>
<td>$\text{length}_{\text{projected}} = \text{length}(e) \times \cos(\alpha)$</td>
</tr>
<tr>
<td>3</td>
<td>$\text{time}<em>{\text{expected}} = \frac{\text{length}</em>{\text{projected}}}{\text{linespeed}}$</td>
</tr>
<tr>
<td>4</td>
<td>$\text{length}<em>{\text{expected}} = \frac{\text{length}(e) \times \text{time}</em>{\text{expected}}}{\text{time}(e)}$</td>
</tr>
<tr>
<td>5</td>
<td>return $</td>
</tr>
</tbody>
</table>

In figure 3.4 there is a graphic sample about the computation of timecost for a segment, with the definitions of projected length, expected time, projected speed and expected length. As showed, the more a road segment is compatible with the line speed of the car, the lower will be its timecost. Our Time Aware Dijkstra algorithm is illustrated in figure 2. The main difference with the classic implementation is the evaluation of the weight of the edges the remaining real time is used to compute the straight line speed of the car. It is worth to notice how the line speed to fit the reconstructed path with changes according to already computed path. For each edge analyzed, timeleft is obtained as the difference between of the path traveltime until the parent node and the real traveltime. This allows us to construct a solution that best fit the real line speed at each step. A particular case arise when timeleft < 0, that is the case when the car is traveling at higher speed than road network edges usual speed. In this situation, the algorithm uses the global linespeed as goal linespeed to fit the remaining path. This still satisfies the constraints of the solution we are searching for. More formally, if timeleft < 0 timecost for the edge is calculated using the straight line from the node currently examined and the remaining time to evaluate the current straight line speed. Otherwise, the the source-target straight line speed is used.

3.4 Analysis of the algorithm proposed

The map matching algorithm we propose is split into two task: point-matching and time aware shortest path. The complexity of the former is due to the search of the k-nearest segments. In our implementation, we used a Generalized Search Tree to index the geometries of the road network: the complexity is then $O(\log|E|)$ with $E$ as the set of road segments. A further refinement is possible: a bounding box based on latitude and longitude of the point to be matched can decrease the set of
Algorithm 2: Pseudocode for Time-aware Dijkstra algorithm

Input: A road graph \( G \); a source node \( s \); a target node \( t \); a travel time \( t \)

Output: A list of traversed road segments \( id \).

1. \( Q = s; \)
2. \( \text{parent}[s] = 0; \)
3. \( \text{weight}[s] = 0; \)
4. while \( Q \neq \emptyset \) do
5. \( n = Q.\text{pop}(); \)
6. if \( n \neq t \) then
7. for edge = \( (n, v) \in \text{neighbors}(n) \) do
8. \( \text{timeleft} = t - \text{pathtime}(s, v); \)
9. \( \text{heading} = \text{edge.heading}; \)
10. if \( \text{timeleft} < 0 \) then
11. \( \text{line} = \text{euclidean.distance}(n, t); \)
12. \( \text{linspeed} = \frac{\text{line}}{\text{timeleft}}; \)
13. else
14. \( \text{line} = \text{euclidean.distance}(s, t); \)
15. \( \text{linspeed} = \frac{\text{line}}{\text{travelt ime}}; \)
16. \( w = \text{Timecost}(\text{edge}, \text{heading}, \text{linspeed}); \)
17. if \( \text{weight}[v] > \text{weight}[n] + w \) then
18. \( \text{parent}[v] = n; \)
19. \( \text{weight}[v] = \text{weight}[n] + w; \)
20. \( Q.\text{push}(v) \)
21. else
22. \( p = \text{parent}[t] \) while \( p \neq 0 \) do
23. \( \text{path.append}(\text{edge} = (\text{parent}[p], p)); \)
24. \( p = \text{parent}[p]; \)
25. return reverse(path);

segments in which to search. The computation of the travel time of each segment is a once running process, and it runs in \( O(\log|E|) \ast |P| \), with \( P \) as the number of GPS points.

The complexity of Time-aware Dijkstra depends from the number of nodes needed to visit until finding the target node. Since the road network is a really sparse graph (most of the nodes have only one neighbor), the visited nodes are mainly depending by the distance between the source node and the target node. According to the characteristics of our dataset (see table 3.3), the average distance between two consecutive GPS points is 1,500m. Dividing the space with a grid of 1.5 km square cells, the average number of segments for every cell is 65. Since the complexity of Dijkstra’s algorithm is \( O(|E| + |V|\log|V|) \), with \( E \) as the number of edges and \( |V| \) as the number of nodes. Furthermore, Time-Aware map matching could be run in parallel. For every two consecutive points of a trajectory, the problems of finding the respective paths are independent from each other. This represents an important improvement w.r.t. global algorithms.
3.5 Implementation details

The Time-Aware map matching algorithm is based, as stated, on Dijkstra’s algorithm for shortest path. We chose this algorithm instead of other faster options because of some important properties. A* algorithm is an heuristic algorithm to find shortest path: thanks to the heuristic approach the algorithm can limit the number of visited nodes, so obtaining a lower complexity w.r.t. to Dijkstra ($O(|E|)$). However, to be optimal A* requires a heuristic function that does not overestimate the distance to the target node. This makes the use of A* algorithm with our Time-Aware approach impossible, since we are not able to define that function using our Timecost instead of the Euclidean distance as it is in usual implementations of A*. In other words, Timecost is already an estimation and it also changes depending on the characteristics of the road network: to use it for an heuristic function for A* we should be able to estimate, in terms of Timecost, the distance between two nodes. By definition of Timecost, this value changes depending on some real parameters, such as car line speed. Then, in our Time-Aware scenario Dijkstra’s algorithm ensures reliable results, since it is not depending on any heuristics, returning the shortest path after having visited all the nodes. Another particular detail is related to the point-matching task. As explained, this task is performed through a gravity model that returns the most probable segment to match the input GPS point, without any indication about the exact point of the road segment where the GPS point should fall. Since this information is useful, we used the approach introduced in [28]: the input GPS point belongs to the closer point of the matched road segment, we are then able to create a new node on the road network connecting the matched point and the target of the matched segment, preserving all the characteristic of the matched segment. Of course, travel time is modified according to the new length of the segment. This procedure guarantees a still better estimation of Timecost, resulting in a better global accuracy for our Time-Aware algorithm.

For a better evaluation of our work, we developed a QGIS plug-in based on our Time-Aware algorithm. QGIS is is a cross-platform free and open-source desktop geographic information systems (GIS) application that provides data viewing, editing, and analysis capabilities. The plug-in allows to match a trajectories layer to a road network layer, obtaining the map-matched data related to input trajectories. An alpha version with some sample shapefiles is available at: http://kdd.isti.cnr.it/software/time-aware-matching-plugin-qgis. We intend to go on developing the plugin, with the goal of a next public release.
3.6 Experimental results

The evaluation of our algorithm has been done through various tests. First of all, we compared the results of Time-Aware algorithm applied to the ACM Sigspatial cup dataset, that is a GPS dataset with correct traversed roads reported by drivers. This dataset gave us the possibility to directly assess the accuracy of Time-Aware algorithm and the competitors. Then, we performed other two tests on two large datasets, OctoPisa and San Francisco cabs (see datasets characteristics on tab. 3.3): using data from OctoPisa we checked the coherence of our Time-aware map matched trajectories w.r.t. Variable Message Panel data, that is the number of cars passing by for some particular roads; on both datasets we defined a “middle-point test” to check Time-Aware accuracy time with half-sampled GPS trajectories. Furthermore, we showed how Time-Aware is better than competitors on fitting the original GPS travel time (see Fig. 3.7).

![Figure 3.5: GPS sampling rate cumulative distribution for Octopisa (left) and San Francisco cabs (right) dataset](image)

3.6.1 ACM SigSpatial Cup dataset

Before the application of our Time-Aware map matching to the real dataset, we evaluated its accuracy. In order to do this, we relied on the dataset of ACM SigSpatial cup 2012 ([3]): 10 trajectories with the correct route annotated by drivers. We compared our algorithm with two competitors: Tang-Zhu-Xiao algorithm ([55]) and IVMM ([63]). The former is the winner of the SigSpatial Cup, it is designed for high
A TIME AWARE MAP MATCHING METHOD TO ENRICH RAW SPATIO-TEMPORAL TRAJECTORIES

sampling rate data, but authors state it is performing well with low sampling data too. The latter is the state-of-the-art map-matching method for low sampling rate. The SigSpatial Cup dataset trajectories have a sampling rate of 1Hz: this gave us the possibility to check the performance of our method and the competitors using different sampling rates by selecting different lower-sampled sub-trajectories extracted from the original ones. Besides our Time-Aware map-matching algorithm, we also developed a simpler version, using the same Gravity Model ([18]) for the point-matching task and a Shortest Path heuristics. The accuracy of Gravity Model matching for the validation dataset is reported on Tab. 3.1. It is simply computed by comparing the number of correct assignments over the total number of points. Since we did not have a proper dataset to compute the travel time of all the segments of the road network, we derived these values from the trajectories provided; the travel time of segments without any GPS point associated has been assigned according to the typical speed recorded for its same-class neighbors. We started this spreading from the neighbors of the segments with at least one gps point associated. The metric used to evaluate the correctness of the map matching algorithms is the same used in the SigSpatial Cup, defined as the ratio between segments correctly matched and the total number of correct segments:

\[ Accuracy = \frac{|Correct \ segments \ matched|}{|Ground \ truth \ Segments|} \]

<table>
<thead>
<tr>
<th>Sampling</th>
<th>Points n.</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1s</td>
<td>13345</td>
<td>0.9732</td>
</tr>
<tr>
<td>10s</td>
<td>1352</td>
<td>0.9415</td>
</tr>
<tr>
<td>30s</td>
<td>462</td>
<td>0.9545</td>
</tr>
<tr>
<td>90s</td>
<td>165</td>
<td>0.9939</td>
</tr>
<tr>
<td>120s</td>
<td>129</td>
<td>0.9689</td>
</tr>
</tbody>
</table>

Table 3.1: Gravity Model point-matching accuracy

Accuracy results
The comparison between our Time-Aware map matching and the competitors is shown in Figure 3.6. Our algorithm is outperforming IVMM and Tang-Zhu-Xiao algorithm. Furthermore, Time-Aware heuristic is working better than the simple shortest path, confirming our starting idea: assuming all the drivers traveling along shortest path lead to slightly inaccurate results. Another difference between Time-Aware map matching and the competitors is the
use of parameters. The only parameter used by Time-Aware map matching is the number of k-nearest neighbor in the point-segment matching process; as stated in [18], using $k = 8$ is a good choice, since the accuracy is not increasing using greater values for $k$. IVMM is using a range query with radius $\epsilon = 100m$ to choose the initial candidates for the matching with a GPS fix. Then, a Gaussian distribution with $\mu = 15m$ and $\chi = 10m$ is used to evaluate the every candidate. Tang-Zhu-Xiao algorithm is relying on a set of parameters as well. The initial choice is made by selecting the 50-closest mini-vertex of the road network. Then, all the candidates at distance greater than 18$m$ from the GPS fix are discarded. In the following evaluation of the candidates, some scaling factors adopted from [10] are used to compute the score for a match: $\mu_\alpha = 10$, $c_\alpha = 4$, $\mu_d = 0.17$ and $c_d = 1.4$. Time-Aware map matching avoids of all these parameters, relying only on data characteristics: the results are confirming the goodness of this approach.

![Figure 3.6: Precision of Time-Aware map matching w.r.t. competitors](image)

**Complexity comparison**

Another improvement achieved with Time-Aware algorithm is a valuable speed up in terms of computation time. As stated before, the complexity of Time-Aware map matching is due to the complexity of point-matching process, $O(\log |E|)$, and the complexity of Dijkstra’s algorithm, that is $O(|E| + |V| \log |V|)$. The main difference w.r.t. competitors is on shortest path computation: Time-Aware uses Dijkstra’s algorithm only once for every two consecutive GPS points. IVMM is computing the shortest path between every couple of candidates for every two consecutive GPS points. This means a sensibly higher number of steps to find the solution w.r.t.
3. A TIME AWARE MAP MATCHING METHOD TO ENRICH RAW SPATIO-TEMPORAL TRAJECTORIES

Time-Aware map matching. Tang-Zhu-Xiao algorithm has a similar behavior: in order to compute the most probable candidates for two GPS points, the shortest path between every couple is candidate is computed.

3.7 Experiments on large datasets

![Fig. 3.7: Average travel times of reconstructed path according to the GPS travel time of the original points, for both Time-Aware and Shortest Path approach applied to Octopisa (left) and San Francisco Cabs (right) dataset. △ indicates the average difference between Path and GPS travel times.]

It is not possible to directly evaluate the accuracy of our Time-Aware map matching on a real dataset, since the correct traversed roads are not reported. However, we performed some tests on two different datasets (see tab. 3.3) to highlight the improvement of our algorithm. The first test we made is a comparison between the travel time of different reconstructed paths according to the used heuristic. Then, we introduced the middle-point test, that is a coherence test; details for this test are provided in the next subsection. Thanks to the Pisa’s VMP dataset (Variable Message Panel) used in [45], we developed a further test on Octopisa dataset; we evaluated the coherence of map-matched trajectories w.r.t. data from traffic monitors by using VMPs data. The observed period is the same for both OctoPisa and VMP datasets, hence we can provide a reliable test by checking the correlation among them. Results are showed on table 3.2. As showed, Time-Aware approach is slightly outperforming Shortest Path approach. More specifically, the correlation between cars passing by counted by VMPs and trajectories passing by the VMP
3.7. EXPERIMENTS ON LARGE DATASETS

<table>
<thead>
<tr>
<th>VMP address</th>
<th>Time Aware</th>
<th>Shortest Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>via Aeroporto</td>
<td>0.4604</td>
<td>0.4602</td>
</tr>
<tr>
<td>via Cascine</td>
<td>0.3376</td>
<td>0.3178</td>
</tr>
<tr>
<td>via di Cisanello</td>
<td>0.5880</td>
<td>0.5758</td>
</tr>
<tr>
<td>via Tosco Romagnola</td>
<td>0.4511</td>
<td>0.4527</td>
</tr>
<tr>
<td>via Brennero</td>
<td>0.4770</td>
<td>0.4823</td>
</tr>
<tr>
<td>via San Jacopo</td>
<td>0.6680</td>
<td>0.6536</td>
</tr>
<tr>
<td>via Pietrasantina</td>
<td>0.6070</td>
<td>0.6083</td>
</tr>
<tr>
<td>via Emilia</td>
<td>0.7050</td>
<td>0.6868</td>
</tr>
<tr>
<td>via Pisano</td>
<td>0.5615</td>
<td>0.5510</td>
</tr>
<tr>
<td>Average correlation</td>
<td>0.539</td>
<td>0.532</td>
</tr>
</tbody>
</table>

Table 3.2: Correlation between VMP data and map matched data

road segment is higher if we use our Time-Aware heuristic instead of Shortest Path. In this test we chose Shortest Path as a competitor, since it is performing way better than IVMM and Efficient matching. Although the improvement reported on this test is not high in absolute terms, in the next subsection we will depict how this small difference is not equally distributed, so enforcing the goodness of our approach: for a test application such as leveraging traffic flows on city access points, the use of Shortest Path heuristic lead to a less precise estimation, especially for some particular roads.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>OctoPisa</th>
<th>SF cabs</th>
</tr>
</thead>
<tbody>
<tr>
<td>N. of trajectories</td>
<td>1,382,892</td>
<td>91,244</td>
</tr>
<tr>
<td>N. of GPS points</td>
<td>19,536,742</td>
<td>11,120,908</td>
</tr>
<tr>
<td>N. of users</td>
<td>38,259</td>
<td>500</td>
</tr>
<tr>
<td>Avg. sampling rate</td>
<td>94.376 s</td>
<td>58.45 s</td>
</tr>
<tr>
<td>Avg. distance between consecutive points</td>
<td>1.538 km</td>
<td>0.587 km</td>
</tr>
<tr>
<td>Avg. point-nearest segment distance</td>
<td>8.65 m</td>
<td>16.87 m</td>
</tr>
<tr>
<td>Std. dev. point-nearest segment distance</td>
<td>22.65 m</td>
<td>33.72 m</td>
</tr>
</tbody>
</table>

Table 3.3: Datasets properties
3.7.1 Datasets

As introduced before, we applied our Time-Aware algorithm on two large datasets. Octopisa is a database composed by almost 20 million of GPS points recorded for insurance purposes by devices installed on 40K different cars. The average sampling rate is 90 seconds (see tab. 3.3). In Fig. 3.10 there is a sample of this dataset: even though GPS is the most precise way to get geospatial data, the average error is 8 meters with a standard deviation of 22 meters, so introducing a lot of noise and making the map-matching task harder.

We also used a public dataset ([48]). This dataset is composed by GPS traces of 500 cabs traveling in the San Francisco Area for a period of 30 days. The distribution of GPS sampling rate is shown in Fig. 3.5: while the sampling rate for San Francisco cabs trajectories seems to follow a Normal distribution, Octopisa has a slightly less uniform distribution, probably due to the GPS collecting policies adopted by the provider.

3.7.2 Temporal alignment

In Fig. 3.7 the correlation between GPS travel time and map-matched path travel time is highlighted, for both Time-Aware and Shortest Path approach. It is clear
how our method yields values closer to the real traversal times, therefore providing a better solution w.r.t. to Shortest Path. Using the Time-Aware approach, the average difference between real time and path time is almost 50% less than Shortest Path approach.

### 3.7.3 Middle-point Test

To further validate our Time-Aware heuristic, we provide an accuracy test made on some real datasets. For each trajectory we hide the middle point of every consecutive GPS points triplet and we repeated the map matching, thus counting the number of hidden points correctly matched. We used this middle-point test to assess the coherence of the different matching algorithms w.r.t. the input trajectory. From table 3.4 is evident how the Time-Aware heuristic is better on reconstructing a trajectory with half of the original points. Figure 3.8 shows the comparison between our Time-Aware algorithm and the competitors with a density map: the points above the line represent the trajectories where the performance of Time-Aware is better than competitors.

As depicted in Fig. 3.8, Time-Aware algorithm is still outperforming the results of competitors. Each density plot represent the comparison between Time-Aware algorithm and a competitor, conducted on both datasets. The area above the line represents the cases where Time-Aware algorithm has a better accuracy w.r.t to the compared method. However, it is worth to point out two differences w.r.t the results obtained on the OctoPisa dataset. IVMM and Efficient-match are performing worse, this is probably due to their high dependence on parameters, as delineated in the previous section. Conversely, shortest and fastest path heuristics obtained better results. This was expected, since all the GPS trajectories have been generated by taxi drivers, who have better knowledge about the road network than any other kind of driver. It is worth to notice the different and less accurate scenario on which we performed the test: travel times for San Francisco road network are estimated from speed limits, since GPS from taxi cabs do not record instant speed. This makes the application of speed estimation ([18]) impossible. Despite this lack of information, the Time Aware heuristics still confirms its reliability.

### 3.7.4 Applications

Once validated our method, we used it to map match our OctoPisa dataset (see Tab. 3.3 and sampling rate cumulative distribution on fig. 3.5). GPS error and sampling rate for this dataset mean a lot of noise: Fig. 3.10 gives an idea of the GPS error we avoided with our map-matching algorithm. We used M-Atlas ([26])
to extract trajectories from raw data. Then, we applied Time-Aware map matching to our dataset, adding a reliable semantic layer to the raw data. This gives us the possibility to exploit many useful analysis. We propose here two examples of data analysis with map matched data. In Fig. 3.11 the usage of road network is shown. Segments are colored according to the number of vehicles passing by. This introduces a new way for traffic monitoring, since nowadays transport managers are mainly using fixed and costly structures as video devices. The availability of GPS and geospatial data is a big chance for transport managers: a deeper and precise view on traffic is now easily achievable. Despite VMPs, which are fixed and costly devices, with this data we can perform traffic analysis in a more flexible way. We can filter the trajectories according to different parameter, like distance traveled, direction (city center, city-to-city, hinterland etc). Then, we can decide where to put our ”virtual” traffic monitors to observe the traffic flow in those points. In Fig. 3.9 an example is provided. We selected all the entering trajectories for two cities, respectively Pisa and Lucca, in order to draw the access points of the city. We leverage the traffic flows on these access points using both Time-Aware and Shortest Path, then we plot differences between fluxes in Fig. 3.9. An expert domain could immediately notice how some minor roads become more important than expected: those segments are part of lots of shortest paths, but their travel time does not fit with GPS travel time. Although the two methods seem to not be really different in absolute terms, as we reported with matched trajectories-VMPs correlation comparison (section 3.7), from Fig. 3.9 it is clear how those differences are not equally distributed. This enforce the importance of our Time-Aware approach, that is able to avoid strange matching of minor roads not used in the reality.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Octopisa</th>
<th>San Francisco cabs</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVMM</td>
<td>0.345</td>
<td>0.13</td>
</tr>
<tr>
<td>Tang-Zhu-Xiao algorithm</td>
<td>0.41</td>
<td>0.14</td>
</tr>
<tr>
<td>Gravity Model + shortest path</td>
<td>0.515</td>
<td>0.94</td>
</tr>
<tr>
<td>Gravity Model + fastest path</td>
<td>0.509</td>
<td>0.95</td>
</tr>
<tr>
<td>Time Aware</td>
<td>0.669</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Table 3.4: Average global accuracy on middle-point test
3.8 Conclusions

We proposed a Time-Aware map matching process based on a new approach to the map matching problem. We have shown how the state-of-the-art algorithms are affected by issues, due to their high number of parameters and the adoption of generally false assumption, i.e. the fact that drivers always take the shortest or fastest route to reach a destination. Thus, we provided a parameter-free algorithm that outperforms the state-of-art competitors, both in terms of accuracy and complexity. The goodness of our approach has been showed on three different datasets. The future developments of this work go towards the study of methods that better exploit the knowledge we can acquire about a single individual when a long history of her movements are available. Moreover, we plan to test the proposed algorithm on richer datasets, such as the one from TagMyDay ([54]): a joint project between ISTI-CNR and Pisa Transport Manager office based on volunteer recruiting for mobility data collection. This dataset has a semantic layer added by volunteers whom indicate the destination of their trips in terms of activity performed (e.g. going home, shopping, etc). Finally, an alternative context to explore with our approach is the map matching of GSM data: since this kind of data has a very poor spatial precision, the use of a Time-Aware heuristic might be the key to extract more meaningful traffic information from GSM mobility data.
3. A TIME AWARE MAP MATCHING METHOD TO ENRICH RAW SPATIO-TEMPORAL TRAJECTORIES

Figure 3.9: Using time-aware map-matched data: this plot highlights traffic flow differences between shortest path and time-aware estimation for the access points of Pisa and Lucca. Points size and color represent the higher (red) or lower (blue) traffic according to the application of Shortest Path approach w.r.t Time-Aware. By comparing this result with the domain knowledge, Shortest Path tends to overestimate the traffic of secondary roads hence underestimating the main ones.

Figure 3.10: A sample of our dataset: raw points can give us only a rough idea of road network usage

Figure 3.11: An application of our map-matching: for every road segment we can compute the number of vehicle passing by. Red segments are the most used, while green ones are the most free
The pervasive presence of mobile personal devices in nowadays society has introduced a new chance for scientists: the study of human mobility. The large amount of georeferenced data collected by the plethora of smart devices represent the best proxy to analyse and understand where and why people move. However, there is still a lack of semantics on data collected by positioning devices: we know where and when a person have been, but usually not the reason for that movement. While spatio-temporal accuracy of such data is increasing, the quality in terms of semantic richness is still poor. It is therefore receiving more and more interest in the literature a broad area of research proposing methods to enrich the traces collected by positioning devices with semantic information. State-of-the-art enrichment techniques aim to transform spatio-temporal data into semantic-rich mobility data. Examples of semantic enrichment of spatio-temporal data include trajectory segmentation (where raw data are split into uniform sequences like moves and stops), the inference of the transportation means, or the performed activity during stops ([46]). The understanding of the activity the individual is performing during the movement is particularly important since it can be seen as the reason a person moves. Movement data annotated with the activities is generally called the mobility diary of a person. The inference of the activity carried out by a moving individual from the raw tracks, in absence of any metadata about the intention of user, is a challenging task that can bring highly innovative contribution to the study of human mobility behavior. Several application areas would benefit from an extensive study on people’s mobile activities: traffic management, public transportation, commercials and advertising, security and police, hazard evacuation management, location based services and so on.

The contribution of this chapter is to offer a method to create mobility diaries...
of moving individuals by semantically enriching trajectories with activities carried out during stops. More specifically, we propose a novel algorithm, called ACTIVE, which analysis raw trajectories and contextual data (i.e. Points of Interests) to infer the activity performed by the moving individuals during the stops. The basic assumption is that people stop to visit a place and thus performing an activity there. For example, a person stopping at a museum is performing a cultural activity, while a stop at a restaurant can be associated to an eating activity. In this context, the ACTIVE method aims at inferring which is the most probable activity of the moving individual during a stop, starting from the analysis of the raw movement represented as a trajectory. To do that, we first identify the places where people stopped; secondly, we associate these places to a list of possible visited Point Of Interest (POI) where an activity can be performed; thirdly, we infer the most probable activity to be associated to the stop. Once each stop is associated to the most probable activity performed, we can derive the mobility diary of every user, hence obtaining a refined representation of her mobility.

The basic steps of ACTIVE are: retrieve the POIs nearby the stops, take into account the POI category (e.g. restaurant, supermarket, etc), map POI categories to specific activities and then exploit a gravity model to relate each stop to the most probable activity among the ones offered by the nearby POIs. The use of gravity law as probability measure is based on the intuition that we need to combine the distance (nearest POIs are more likely to be visited) with the number of POIs in a given category (if we have more restaurants than supermarkets, an eating activity is more probable than a shopping one).

4.1 Concepts

This work is based on the notion of users movements, formally referred to trajectory. In the literature the definition of trajectory has several variants. The most intuitive represents a trajectory as the spatio-temporal evolution of a moving object, expressed in terms of consecutive observations. Based on that, we define the user history as the sequence of observations for a given user for all the tracked interval, being one day, one week, months or years.

**Definition 5.** A user history $T_u$ for a user $u$ is an ordered list of points $T_u = p_1, p_2, p_3, \ldots, p_n$, where $p_j = (x_j, y_j, t_j)$, $x$, $y$ and $t$ are the spatial and temporal coordinates of the observed point, and $t_1 < t_2 < t_3 < \cdots < t_n$.

Although a trajectory is generally defined in the literature as a segment of the user history based on a splitting criteria, here, for the sake of readability, we some-
4.1. CONCEPTS

time use the general term trajectory to indicate the user history. We are, in fact, not interested in the trajectory splitting, but only in the stops in a user history.

**Definition 6.** Given a trajectory $T$, a stop $s = t_i, \ldots, t_k$ is a continuous sub-sequence of $T$ representing the absence of movement.

A stop is identified by the absence of movement identifiable in several ways as proposed in [43, 38, 46]. The segment of a trajectory between two stops is called *move* or *trip* and indicates the actual movement between two stops. The segmentation of a trajectory in stops and move was originally called *semantic trajectory* in [53]. However, other more complex definitions have been proposed recently like in [46, 9]. In these works the notion of semantic trajectory goes beyond the simple stop-and-move idea including other contextual aspects, like transportation means, environment, activity or the purpose of the movement. Semantic trajectory thus includes all the possible aspects that can enrich a simple raw trajectory with more knowledge. The process of annotating a raw trajectory with semantic information is called *Semantic Trajectory Enrichment*.

Our work is a contribution in semantically enriching trajectories, focusing on the inference of the activity performed during the stops. The activity carried out during a stop can be seen as the goal of the movement. In other words, the activity explains why a person decided to move from the previous stop to the actual stop (from home to work, from work to shopping, etc). We define the user history annotated with the performed activities as the *mobility diary* of the individual, and the process of inferring the diary from raw data as the *Activity Inference*.

In a urban context, a place that is of interest for the general population and where we can do some activities is denoted as *Point Of Interest* or POI. Each POI has a name, a geographical position, a category and additional metadata information such as opening hours. This is formally defined below:

**Definition 7.** **Point Of Interest.** A Point Of Interest (or POI) is a spatial object $P$ represented as a tuple $\langle \text{POI}_\text{Id}, (\text{Lat, Long}), \text{POI}_\text{name}, C, H \rangle$, where $\text{POI}_\text{Id}$ is the unique identifier, $\text{Lat, Long}$ are the geographical coordinates of the representative spatial point, $\text{POI}_\text{name}$ is a string denoting the name, $C$ is the category, $H$ list the opening hours.

An example of POI is Eiffel Tower in Paris: the representative spatial point is the center of the tower (latitude 48.85837, longitude 2.294481), the categories are “tourist attraction” or “monument” or “tower”, the label “Eiffel Tower” denotes the name and the opening hours are “daily from 9 to midnight”.
Special case of POIs are the places that are of interest only for a specific user, like home, work place, friends’ house, etc. However, in this chapter we refer to POI intending only the places of interest to a community of people and that can be usually found in many applications like GPS navigators, Google maps, Social networks, etc.

In the next sections we will introduce how home and work locations can be detected by trajectory analysis methods like the one proposed in [49].

Given a stop, we indicate its neighborhood as the circular area around the stop that the user can reach walking.

**Definition 8. Stop Neighborhood.** Given a stop $s$, we call the circular buffer area with radius $\delta$ around the stop as the neighborhood $\psi(s, \delta)$ of the stop.

$\delta$ indicates the maximum walking distance. An example of $\delta$ value is 1 kilometers if we assume the person cannot walk for more than 1 kilometer to reach the destination POI.

Each POI located inside the stop neighborhood is a POI reachable from that stop.

**Definition 9. Reachable POI.** A POI is called to be reachable for a given stop if it is in the neighborhood of the stop $\psi(s, \delta)$ for a fixed $\delta$.

A reachable POI is called admissible when the temporal duration of the stop is long enough to go walking to the stop and return back.

**Definition 10. Admissible POI.** A POI is called admissible for a given stop $s$ if it is reachable from $s$ and the duration of the stop is greater or equal to the walking time to go from $s$ to the POI and return back.

The basic assumption of our work is that during the visit to a POI a person may perform an activity, like eating, shopping, studying, playing. The association of an activity to a POI can be based on the category of the POI. To make this association clear, we define a list of activities $A$, a list of POI categories $C$ extracted from the set of POIs, and then we map each POI category to an activity, thus defining a POI-to-Activity mapping $f_M$ as follows.

**Definition 11. POI-to-Activity Mapping.** A POI-to-Activity mapping between a list of POI categories $C$ and a list of activities $A$ is a function $f_M$ where $f_M(c) = a$ where $c \in C$ and $a \in A$.

For instance, consider the Museum of Louvre POI: the category is Museum. If the list of activities contains the item Cultural we can define a mapping $f_M(Museum) = Cultural$, thus uniquely associating each museum to a Cultural activity.
A POI is hence called a candidate POI if it can be the place where the user has performed an activity.

**Definition 12. Candidate POI.** A POI $p$ is called a candidate POI for a given stop $s$ if all the following conditions apply: (i) $p$ is admissible from $s$; (ii) the opening time of $p$ is included in the interval defined by stop start and end time; (iii) the typical activity duration for the mapped POI category is compatible with the stop duration, also considering the time spent to reach the POI and return back to the stop place.

Once selected the candidates, we need a model to infer which one of the candidate POIs is the most probable to have been visited. We chose a simple but powerful probabilistic approach, called *Gravity Model* ([20]).

**Definition 13. Gravity Model.** The Gravity Model is derived from Newton’s Law of Gravitation and used to predict the degree of interaction between two bodies. This degree is proportional to the masses and inversely proportional to the square distance between them, represented by the well known formula $\text{GravityModel} = \frac{\text{mass}_1 \times \text{mass}_2}{\text{distance}^2}$.

In this work we instantiate the original definition of the Gravity Model using the principle of bodies attraction where $\text{mass}_1$ represents the stop point - to which we give value 1 by definition, and $\text{mass}_2$ represents the “mass” of the POI categories. In other words, we provide a probability for the POI categories and not for every single POI. This is in line with our objective of identifying the activities performed during a stop rather than the specific visited POI.

More in detail, for each stop the algorithm instantiates the original definition of the Gravity Model in such a way that $\text{mass}_2$ is the number of reachable POIs of a given category, and the distance is the minimum distance among all the distances of POIs associated to the same activity. This is detailed in the next section.

The output of the ACTIVE algorithm is the *mobility diary*, represented as the sequence of the stops extracted by the user history and annotated with the most probable activity.

**Definition 14. Mobility Diary.** The mobility diary $\text{MD}_u$ of a moving user $u$ is the list of stops annotated with the most probable activity the user may have performed during each stop: $< (s_1, a_1) \ldots (s_n, a_n) >$ where $s_1, \ldots s_n$ are the stops extracted by the user history $T_u$ and $a_1, \ldots a_n$ are the activities inferred for the relative stops.
4. INFERRING HUMAN ACTIVITIES FROM GPS TRACK

4.2 Methodology

The activity inference process aims at semantically enriching a raw trajectory by transforming a set of personal GPS tracks into the mobility diary of a person. We assume that raw trajectories are collected from on-board GPS devices. In this scenario, a person moving by car parks and walks to reach the destination place where her/he performs an activity.

A similar case is represented by GPS trajectories collected with personal devices such as smartphones. The activity inference process becomes simpler because we can assume the person directly reaches the destination place with the tracking device: all the activity inference process collapses to a stop computation and a mapping between POI categories and activities, which is a subset of the proposed process. For this reason here we assume to analyze only vehicles trajectories, being, at the same time, general enough to cover the other case.

It is worth to point out that we base our approach on some assumptions. First of all, during a stop a person is assumed to perform only one activity though in reality sometimes more than one activity could be done: stopping at a bar to take a coffee then buy some food at the supermarket. In this case we consider one activity as the “primary” activity or the main reason of the movement. In the example, the main activity is *daily shopping* at the supermarket while taking a coffee is considered as a secondary activity thus ignored by our method.

Another assumption we made is that a person parks a car and then proceed walking to the desired destination. This may be not always true since, depending on the location, a person could take a tram, a bus or the metro to reach the destination. A future improvement of the current method will consider the presence of bus/tram/metro stop in the buffer around the car stop and perform inferences on the duration of the stop to include the possibilities of using public transportation to reach a further place. However, for the sake of simplicity, we do not consider this option in the current work.

It follows an example of the main concepts and steps of the activity inference process.

Suppose that on Sunday the user $U$ stops at $s_1$ at 10:50 a.m. for 80 minutes as illustrated in Figure 4.1. The POIs *reachable* from $s_1$ and the correspondent distances in meters are indicated by the bigger circle and are: Church (600 m), Dentist (900 m), Bank (240 m), Bar A (120 m), Bar B (2400 m), Bar C (960 m), Restaurant (720 m), Library (600 m), and Gym (900 m). We know the opening hours of each POI and the typical durations of the activities performed in the POI. Assuming that an average walking speed of a person is 1 m/s, $U$ takes respectively 20, 30, 8, 4, 80, 32, 24, 20, and 30 minutes to go and return from each POI.
4.2. METHODOLOGY

Figure 4.1: Example: the stop $s1$ with the reachable and admissible POIs.
Subtracting from the stop duration (80 minutes) these walking travel times, all the POIs except for Bar B are admissible POIs (inner circle). Furthermore, we notice that, depending on the distance, the user can spend different amount of time in each POI and this may be compatible or not with the typical activity duration. Table 4.1 summarizes the available information for each POI, the stop duration, the typical activity durations and the resulting candidate POIs to be considered for computing the most probable activity.

<table>
<thead>
<tr>
<th>POI Name</th>
<th>POI Category</th>
<th>Activity</th>
<th>Opening hours</th>
<th>Activity dur. $\mu$ ($\sigma^2$)</th>
<th>Distance (mt)</th>
<th>Trip dur. (min.)</th>
<th>Time left (min.)</th>
<th>Candidate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Church A</td>
<td>Church</td>
<td>Leisure</td>
<td>Mon-Sat [18:00-19:00]</td>
<td>92 (89)</td>
<td>600</td>
<td>20</td>
<td>60</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sun [11:00 - 12:00]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dentist B</td>
<td>Dentist</td>
<td>Service</td>
<td>Mon-Sat [9:00-13:00]</td>
<td>45 (56)</td>
<td>900</td>
<td>30</td>
<td>50</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mon-Sat [15:30-18:00]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank C</td>
<td>Bank</td>
<td>Service</td>
<td>Mon-Fri [09:00-15:30]</td>
<td>45 (56)</td>
<td>240</td>
<td>8</td>
<td>72</td>
<td>✗</td>
</tr>
<tr>
<td>Bar D</td>
<td>Bar</td>
<td>Food</td>
<td>Mon-Sun [6:30-21:00]</td>
<td>34 (28)</td>
<td>120</td>
<td>4</td>
<td>76</td>
<td>✗</td>
</tr>
<tr>
<td>Bar E</td>
<td>Bar</td>
<td>Food</td>
<td>Mon-Sun [6:30-21:00]</td>
<td>34 (28)</td>
<td>2400</td>
<td>80</td>
<td>0</td>
<td>✗</td>
</tr>
<tr>
<td>Bar F</td>
<td>Bar</td>
<td>Food</td>
<td>Mon-Sun [06:30-21:00]</td>
<td>34 (28)</td>
<td>900</td>
<td>32</td>
<td>48</td>
<td>✓</td>
</tr>
<tr>
<td>RestaurantG</td>
<td>Restaurant</td>
<td>Food</td>
<td>Mon-Sun [11:45-15:00]</td>
<td>34 (28)</td>
<td>720</td>
<td>24</td>
<td>56</td>
<td>✗</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mon-Sun [18:45-23:00]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Library H</td>
<td>Library</td>
<td>Shopping</td>
<td>Mon-Sun [9:00-13:00]</td>
<td>38 (45)</td>
<td>600</td>
<td>20</td>
<td>60</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Mon-Sat [15:30-19:00]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: This table summarizes the main data for the example: POI name, POI category, Mapped activity, Opening hours of the POI, Typical activity duration described by mean $\mu$ and variance $\sigma^2$ (in minutes) associated to the activity, Distance of the POI from the stop (in meters), duration of the trip to go walking and return from stop to POI (in minutes), remaining time (in minutes) to perform the activity at the POI, and the resulting list of candidates POIs ("✗" means that the POI is excluded).

The candidate POIs are selected by applying some spatio-temporal constraints. First of all, the opening hours allow to discard POIs which are closed during the stop: this is the case of the Bank and the Dentist, closed on Sunday. Then, by comparing the remaining time (subtracting the walking travel time from the stop duration) and the typical duration of the activities, we discard the POIs which are too far from the stop or for which the remaining time is not sufficient to perform the activity.

The typical duration of the activities is a distribution described by mean $\mu$ and standard deviation $\sigma$. The computation of the corresponding range of the activity duration is described in the following Section 4.3.1. This constraint filters out the following POIs: Bar B because it is too far away, Bar A and Restaurant because the remaining time is higher than the maximum time that can be spent for this activity according to the range computed as in Equation 4.1.

With the remaining candidate POIs, we compute the probability for $U$ to perform
4.3. THE ACTIVE PROCESS

a specified activity during his stop by using a gravity model as defined in Def. 13. The probability for each activity category is computed as follows:

\[
P(\text{Leisure}) = \frac{2}{600} \cdot \frac{1}{\varphi} = 0.59 \\
P(\text{Food}) = \frac{1}{920} \cdot \frac{1}{\varphi} = 0.11 \\
P(\text{Shopping}) = \frac{1}{600} \cdot \frac{1}{\varphi} = 0.30
\]

\(\varphi\) is the normalization factor corresponding to: \(\frac{1}{\left(\frac{2}{600} + \frac{1}{920} + \frac{1}{600}\right)}\).

4.3 The ACTIVE process

The activity inference process of ACTIVE is depicted in Figure 4.2. The inference process needs as input: the user history, some domain information like the POIs with the related metadata, and the POI-to-Activity mapping function. The process includes several steps: (i) the Stop detection in which the stops are identified from the user history; (ii) the home and work computation where the location assigned to activities “home” and “work” are computed; (iii) POIs retrieval where the Points of Interests are collected for each stop, and (iv) the activity annotation step where the most probable activities associated to the POIs are identified to annotated the stops not previously assigned to home and work.

![Figure 4.2: A schema of the ACTIVE process](image)

The Stop Detection component takes the raw trajectories as input and compute the points of stop. There are many methods for detecting stops, as illustrated in [46]. Essentially, they follow two approaches: the clustering-based like in [5] and the heuristic-based like in [58]. In the experiments presented in Section 4.4 we adopt the algorithm implemented in M-Atlas (a platform for data mining of mobility data described in [26]) that takes inspiration from [5] and represents a trade-off between
precision and efficiency. Given a raw trajectory composed of a set of points, the algorithm searches the points that change only in time, i.e. points that stay in the same spatial position for a certain amount of time quantified by the temporal threshold. At the same time, a spatial threshold is used to remove both the noise introduced by the imprecision of the device and the small movements that are of no interest for a particular analysis. These thresholds are used for detecting the candidate stops. The setting of the stop parameters depends on the specific application or analysis purpose. For instance, for detecting activities like refueling, or stop at the traffic light we need to set a very short stop duration, while for detecting activities like shopping or eating we need to set a longer period to avoid noise in the data. In the experiments reported in Section 4.4 we use method described in [26]. However, the choice of the stop computation method does not affect the activity inference process we propose.

Once the stops are identified, the **Home-Work Detection** component finds home and work locations. This step is required since home and work activities are not related to any POI and the corresponding stops have to be excluded from ACTIVE input. Home and work can be identified in several ways. One approach described in [39] identifies as home and work the two most frequent locations of a user. An alternative solution proposed in [19], identifies home and work basing on the duration of the stop and on the time of the day (long stop during the night is probably the home, while a long stop during the working days is probably the work place). In the experiments section we use the method proposed in [39] although the ACTIVE algorithm is parametric respect to the specific home-work method used.

The **POIs Retrieval** component retrieves the POIs in the geographical area of interest. POIs can be retrieved from different sources such as Foursquare\(^1\), Google Places\(^2\), Yellow Pages\(^3\) or other local providers. The availability of POIs depends on the area under observation and some providers could have a better coverage in some areas while others better in other areas. For the experiments illustrated in Section 4.4 we use Google Places since it provides a good coverage of the areas under analysis. Independently from the data source, it is important to make sure that the POIs are representative of the most common categories, so as to infer a large number of activities with high accuracy. In fact, when a category is not represented by any collected POI, the corresponding activity (associated through the POI-to-Activity mapping function) can never be inferred by our method.

---

\(^1\)Foursquare: www.foursquare.com  
\(^2\)Google Places: www.places.google.com  
\(^3\)Yellow Pages: www.yellowpages.com/
The Activity annotation component is the core of the inference algorithm: it combines the POIs and their metadata information, like opening hours, to the stops to filter the candidate POIs. The Gravity Model is applied to assign probability to the activities mapped to the candidate POIs categories.

The algorithm that implements the ACTIVE process is introduced in the next section.

### 4.3.1 ACTIVE Inference algorithm

ACTIVE, the Activity Inference algorithm described in Algorithm 3, is composed of three modules: i) Stop Computation and Home-Work Detection, ii) Candidate POI selection with filters (Algorithm 4 and Algorithm 5) and iii) Probability computation based on the gravity model (Algorithm 6).

#### Algorithm 3: ACTIVE

*Input*: A set of user histories $T_r$; A dataset of POIs $D_{POIs}$; A list of activities $A_L$.

*Output*: A set of Mobility Diaries $T_A$.

1. for $t \in T_r$ do
2.   $List_{of\ Stops} = \text{StopDetection}(t)$;
3.   $List_{of\ H&W\ Stops} = \text{Home&WorkDetection}(List_{of\ Stops}, t.user)$;
4.   $List_{of\ Stops} = List_{of\ Stops} - List_{of\ H&W\ Stops}$;
5.   for $s \in List_{of\ Stops}$ do
6.     $List_{of\ POIs} = \text{SelectPOI}(s)$;
7.     $Activity = \text{ComputeProbability}(List_{of\ POIs}, s, A_L)$;
8.     $s' = \text{Assign}(Activity, s)$;
9.     $t_A = \text{Add}(s')$;
10.  $T_A = \text{Add}(t_A)$;
11. return $T_A$;

ACTIVE takes as input a set of raw trajectories, a dataset of POIs, a list of activities and a POI-to-Activity mapping function, and returns a set of mobility diaries.

For each user history, the StopDetection(.) function (line 2) processes the GPS points of the raw trajectories and returns the stop points. This function uses a spatio-temporal threshold checking if the moving object remains into a spatial buffer $\delta$ for a given time interval $\tau$ as already discussed above. At line 3, the function Home&WorkDetection(.) - related to user history $(t.user)$ - returns the stops corresponding to Home and Work locations that have to be excluded from the activity inference process (line 4). At line 5, for each stop, the SelectPOI(.) function (described by Algorithm 4) returns the list of candidate POIs. At line 7, the ComputeProbability(.,.,.) function (described by Algorithm 6) applies the Gravity Model strategy and returns the most probable activities performed during the stop. An
activity annotated stop is created (line 8) and added to the final annotated trajectory representing the mobility diary (line 9).

The SelectPOI function, described by the Algorithm 4, takes as input a stop and returns a set of candidate POIs for the stop. For each POI selected by the function NearestPOIs(.) at line 1, the function CheckFilters(.,.,.) (described by Algorithm 5) selects the candidate POIs (line 2).

The CheckFilters function, described by Algorithm 5, takes as input a stop, a POI and a list of possible filters (opening time, admissible POIs and stop duration), and returns “true” if the POI is a candidate POI for that stop. The filters have the following meaning:

Opening time checks if a visit to a POI is compatible with the opening time of the POI. Given the timestamps of the stop we verify if the POI is open during that temporal interval.

Admissible POIs is a refinement of the Maximum Walking Distance (MWD) used in [24]. Assuming a walking speed of 1 m/s, a POI is admissible if the stop duration is greater or equal to the walking time required to reach the POI from the stop and go back. For example, a stop of 6 minutes includes all the stops which are within 3 minutes walking away, while excludes POIs that are more than 3 minutes walking away. The distances are computed on the road map, including pedestrian paths, after applying a mapping between POI location and the road network. For the map matching task, we use the mapping algorithm described in 2, where GPS points from low-sampled trajectories are mapped on the road with an accuracy of 77%. Once stops and POIs are mapped on the road, an algorithm to compute the shortest path is applied.

Stop duration checks if the remaining time from the stop duration and the walking time to go from the stop to the POI, is compatible with the activity duration associated to a POI. The typical activity duration can be inferred from external data sources like interviews or surveys. Assuming a normal distribution, for each activity, we compute the average duration ($\mu$) and its standard deviation ($\sigma$). Then, for each stop we compare the stop duration with a maximum (Upper Limit) and minimum
(Lower Limit) defined as:

\[
\text{Lower Limit} = \mu - Z \ast \sigma \\
\text{Upper Limit} = \mu + Z \ast \sigma
\]  

(4.1)

where \( Z = 0.6 \). The choice of \( Z = 0.6 \) implies that the probability \( P \) of finding a

duration \( X \) among the limits is \( P = \{ \mu - 0.6 \ast \sigma < X < \mu - 0.6 \ast \sigma \} = 46\% \). We

chose a low value for parameter \( Z \) in order to get rid of noise: all the stop with a
duration too different for each mean duration of all the activities has been excluded.

**Algorithm 5: CheckFilters**

| Input: A stop point \( s \); A POI \( p \); A list of filters \( f \).
| Output: A boolean value \( \text{Compatible} \), indicating if the POI visit is compatible with the stop, according to the filter.

1. \( \text{Compatible} = \text{False} \);
2. if \( \text{(f contains 'opening time')} \) /* Opening time filter */ then
3. \( \text{if } (s.\text{timestamp} \subseteq p.\text{opening time}) \text{ then } \text{Compatible} = \text{True}; \)
4. if \( \text{(f contains 'admissible POI')} \) /* Admissible POI filter */ then
5. \( \text{wd} = \text{distance(stoppoint, poi point)}; \)
6. if \( \text{(wd} \ast 2 < \text{stop duration)} \text{ then } \text{Compatible} = \text{True}; \)
7. else \( \text{Compatible} = \text{False}; \)
8. if \( \text{(f contains 'stop duration')} \) /* Stop Duration filter */ then
9. \( \text{if } (p.\text{stop duration} \subseteq \text{mean duration(p.category)} \pm 0.6 \ast \text{std duration(p.category)) then } \text{Compatible} = \text{True}; \)
10. else \( \text{Compatible} = \text{False}; \)
11. return \( \text{SelectedList}; \)

The **ComputeProbability** function described by Algorithm 6 takes as input the set of candidate POIs, the stops, the list of activities, the mapping function, and returns the most probable activity performed during the stop. For each candidate

POI, the gravity model is applied according to the Equation 4.2 and Definition 13.

More formally, for every stop \( s \) we determine the probability \( P \) of visiting a POI and performing an activity as:

\[
P(s, \text{act}) = \frac{|\{p \in \text{SelectedPOIs}(s)|f_M(p.\text{category}) = \text{act}\}|}{\min(d(s, p)^2)}
\]  

(4.2)

where, \( \text{SelectedPOIs} \) returns the candidate POIs selected using the Algorithm 4 given the stop \( s \). \( p.\text{category} \) indicates the category of POI \( p \) and \( d \) is a function returning the distance between the stop and the set of POIs \( p \) associated to the same activity. As we show in Algorithm 6, the candidate POIs are the input for the **ComputeProbability** algorithm. Thus, we associate a probability to each possible activity related to the stops. To do this we take into account the characteristics of the area where the user stopped and not only the distance of the POIs from the
stops. For example, when a stop is in an area with many restaurants and few shops, the Gravity Model gives more importance to restaurants w.r.t. shops, then making a better distinction between the two possible mapped activities (Food or Shopping).

Algorithm 6: `ComputeProbability`

Input: A set of POIs $P$; The stop $s$; The list of activities $AL$; the POI – to – Activity mapping function $f_M$.

Output: The activity performed $Activity$.

1. $Activity = \text{None}$;
2. $MaxProb = 0$;
3. for $act \in AL$ do
4.     $POIs_{act} = \{p \in P : f_M(p) = act\}$;
5.     $dist = \min(distance(s, p) \text{ for } p \in P)$;
6.     $mass = |POIs_{act}|$;
7.     $prob = mass/dist^2$;
8.     if $prob > MaxProb$ then
9.         $Activity = act$;
10.        $MaxProb = prob$;
11. return $Activity$;

We conclude this section by briefly commenting on the time complexity of ACTIVE. The stop detection task is $O(t)$, with $t$ as the number of points of the user history. Home and Work detection is based on a density clustering process, requiring $O(t \ast \log(t))$ steps. The remaining demanding task is the collection of the POIs in the stop neighborhood for each stop: with a spatial index this task can be accomplished with $O(\log(p))$ steps, with $p$ as the size of POIs dataset. We can state that this algorithm can be run in a reasonable time with no specific need for high performance infrastructure.

4.4 Experiments

In this section we present and discuss the experiments performed to compute the accuracy of ACTIVE in real datasets. We first introduce the datasets - both annotated trajectories and POIs - used in the experiments, then we discuss the results obtained by ACTIVE in terms of accuracy with respect to the ground truth and compare the results with a null model. Finally, we show an example of application of ACTIVE to a large - not previously annotated - GPS dataset, and a spatial classification of urban areas according to the activities detected by ACTIVE.

4.4.1 Trajectories Datasets

The validation of the algorithm is performed on trajectories belonging to three different annotated GPS datasets and using POIs collected from different sources.
Every GPS dataset provides both users tracks and the annotated activities (diaries) for each trip, thus providing a ground truth to measure the accuracy of ACTIVE inference. The three datasets are detailed below:

- **GPS_Flanders**: This dataset collects GPS annotated trajectories of volunteers moving by car in Flanders (Belgium) from January 2007 to February 2009. The trips are recorded by in-vehicles GPS devices, and annotated by the users with the activities performed during the stop. Details: Nr. of Trajectories = 30,000; Nr. of users = 28; Nr. of annotated stops = 30,670.

- **GPS_Pisa**: This dataset contains car trajectories from users moving in the city of Pisa, in Tuscany (Italy). This dataset covers a period of 5 weeks in May/June 2011. The trajectories are annotated by domain experts based on the knowledge of the area, the time and the duration of the stop. The way in which the trajectories have been annotated introduces a bias in the data since it is done a-posteriori by users different from the car drivers. However, since the domain experts know the area very well, we believe that this annotation can be considered quite reliable. Details: Nr. of Trajectories = 370; Nr. of users = 67; Nr. of annotated stops = 7,938.

- **GPS_TagMyDay**: This dataset has been collected during TagMyDay\(^4\), a project promoted by KDD laboratory at CNR and University of Pisa. The project aimed at involving the population in the collection of mobility data. Volunteers tracked themselves with a smartphone app and annotated the trajectories by using a web application. The annotations consist on both the activities performed during the stops, and the transportation means used to move between stops. Details: Nr. of Trajectories = 14,323; Nr. of users = 167; Nr. of annotated stops = 14,286.

For all the datasets, the trajectories are described by longitude, latitude and timestamp, while the diaries contain, for each user, the list of activities performed during each stop with timestamp and duration associated. Unfortunately, due to severe privacy constraints, these datasets cannot be disclosed to the public, therefore we cannot make them available for download.

### 4.4.2 Stops, POIs and Activities

Stops are computed from the user histories with a function that computes the size of the minimal convex region including a set of points as in [?]. In other words,
we detect a stop when a subset of the GPS measurements of the raw trajectory $T$ remains into a spatial buffer of radius $r$ for a reasonably long time interval $\tau$. Here the empirical evaluations suggested to use as parameters $r = 50$ mt and $\tau = 10$ min. This means that we consider the parts of trajectories whose points stay in a buffer of 50 meters for more than 10 minutes as stops. We use 10 minutes as the minimum temporal interval to stop the car and perform an activity. It is clear that very short stops like refueling are likely to be missed with these parameters.

The POIs have been downloaded by using Google Places APIs and consist of name, location, and type (i.e. the commercial category).

Only in the case of GPS_Pisa we use an additional POI dataset collected from the local Yellow Pages, adding a further test on the accuracy of ACTIVE, that is, by design, highly dependent on the quality of the POI dataset used.

The POI categories have been grouped into six macro categories corresponding to the activities generally used in the transportation domain. However, our method is general and does not depend on the specific activities used. What is important is to establish a clear correspondence between POI categories and activities by means of the POI-to-Activities mapping function. We do this manually, by building a correspondence table between POI categories and activities (Table 4.2).

<table>
<thead>
<tr>
<th>Activity</th>
<th>POI Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>{ATM, Bank, Car rental, Dentist, Doctor, Hospital, Pharmacy, Finance, Insurance,</td>
</tr>
<tr>
<td></td>
<td>Gas station, Travel agency, Post office, . . . }</td>
</tr>
<tr>
<td>Food</td>
<td>{Bakery, Bar, Cafe, Food, Meal takeaway, Restaurant}</td>
</tr>
<tr>
<td>Daily Shopping</td>
<td>{Grocery or supermarket, Shopping mall}</td>
</tr>
<tr>
<td>Shopping</td>
<td>{Book store, Clothing store, Electronics store, Florist, Furniture store,</td>
</tr>
<tr>
<td></td>
<td>Home goods store, Jewelry store, Library, Pet store, . . . }</td>
</tr>
<tr>
<td>Education</td>
<td>{School, University}</td>
</tr>
<tr>
<td>Leisure</td>
<td>{Airport, Amusement park, Church, Gym, Museum, Night club, Park, Spa, Stadium,</td>
</tr>
<tr>
<td></td>
<td>Zoo, . . . }</td>
</tr>
</tbody>
</table>

Table 4.2: Mapping POI-to-Activity.

It is worth recalling that the POI-to-Activity mapping does not include Working and Activity at Home, since they are not related to POIs available in the POI data sources and they are precomputed by ACTIVE using state of the art methods.

According to the mapping in Table 4.2, Figure 4.3 shows the distributions of POIs related activities of the Google places and Yellow page datasets used in the experiments.

We can notice that in Google Places dataset the activity categories are better represented and are more balanced than the ones in the Yellow Page dataset. This is because the distribution of the POIs along the different categories is more uniform.

\footnote{FP7 EU DATASIM project: http://www.datasim-fp7.eu}
4.4. EXPERIMENTS

Figure 4.3: Distribution of the ones mapped to POI categories from Google Places (Left), and Yellow pages (Right). We can see how Yellow pages is unbalanced towards the services category, while Google Places has more “Food” related POIs.

The Yellow Pages dataset has in fact a higher number of POI belonging to Services and very few to the other categories. We will see with the experiments how this bias affects the activity inference result.

Since the activity duration is not yet a metadata available in the POI datasets, we infer this information from US Census data, freely available on the Bureau of Labor Statistics website\textsuperscript{6}. Lacking of local data, we chose to infer the activity duration from a real dataset instead of simulating them. This survey contains activity diaries resulting from interviews on a statistically significant sample of population. It describes 76,584 individuals interviewed about their daily routine on a specific day, classified according to a very detailed taxonomy\textsuperscript{7}, together with the timestamp and related duration. Since the taxonomy is much more detailed and structured than our activities list reported in Table 4.2, we manually map the survey activities into the activity list selected for this work.

Figure 4.4 shows the distribution of the durations for the activities. We can notice that the typical durations are reasonable: Education (attending school or lessons at the University) has the long duration while daily shopping has a shorter duration than shopping. Starting from these distributions, we compute mean ($\mu$) and standard deviation ($\sigma$) for each activity to define the activity duration limits (Equation 4.1).

It is worth noticing that the opening hours of the POIs are not still part of the metadata of our datasets, therefore we manually compiled a timetable based on the typical opening times of the different POI categories. Table 4.3 shows the time table

\textsuperscript{6} American Time Use Survey: http://www.bls.gov/tus/datafiles_0312.htm
\textsuperscript{7} American Time Use Survey taxonomy: http://www.bls.gov/tus/lexiconnoex0312.pdf
Figure 4.4: Distribution of the activities durations taken from the US Census Survey.

for a subset of POI categories. It is fair to assume that real data will be available directly from the POI provider in the next future, thus giving more accurate results.

<table>
<thead>
<tr>
<th>POI Category</th>
<th>Mon-Sat Opening Hours</th>
<th>Sun Opening Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bar</td>
<td>[7:00 - 23:00]</td>
<td>[7:00 - 23:00]</td>
</tr>
<tr>
<td>Restaurant</td>
<td>[11:30 - 15:00][18:30 - 22:00]</td>
<td>[11:30 - 15:00][18:30 - 22:00]</td>
</tr>
<tr>
<td>Bank</td>
<td>[08:30 - 13:30][14:45 - 16:15]</td>
<td>closed</td>
</tr>
<tr>
<td>Gym</td>
<td>[09:00 - 23:00]</td>
<td>closed</td>
</tr>
<tr>
<td>Hospital</td>
<td>[00:00 - 24:00]</td>
<td>[00:00 - 24:00]</td>
</tr>
<tr>
<td>Museum</td>
<td>[10:00 - 18:00]</td>
<td>[10:00 - 18:00]</td>
</tr>
<tr>
<td>Night club</td>
<td>[22:00 - 05:00]</td>
<td>[22:00 - 05:00]</td>
</tr>
<tr>
<td>Post office</td>
<td>[08:15 - 13:30]</td>
<td>closed</td>
</tr>
<tr>
<td>Shopping mall</td>
<td>[08:00 - 21:00]</td>
<td>[08:00 - 21:00]</td>
</tr>
</tbody>
</table>

Table 4.3: POI categories and opening hours used in the experiments

4.4.3 Evaluation

To evaluate ACTIVE we performed tests on the GPS datasets presented above that represent our ground truth.

For each dataset of trajectories we computed the stops, identified the home and work locations, then apply filters to the remaining stops to select the candidate POIs. Once we have identified the candidate POIs for each stop, we apply the Gravity Model to assign a probability to activities associated to the stop.
Table 4.4 shows an example of an annotated trajectory representing the mobility diary of a user.

<table>
<thead>
<tr>
<th>User id</th>
<th>Timestamp start</th>
<th>Timestamp stop</th>
<th>Timestamp end</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>13159</td>
<td>2011-11-28 8:45:12</td>
<td>12:00</td>
<td></td>
<td>51.280</td>
<td>3.413</td>
<td>Working</td>
</tr>
<tr>
<td>13159</td>
<td>2011-11-28 14:28:40</td>
<td>14:45</td>
<td></td>
<td>51.0302</td>
<td>3.4212</td>
<td>Shopping</td>
</tr>
<tr>
<td>13159</td>
<td>2011-11-28 14:50:01</td>
<td>18:30</td>
<td></td>
<td>51.280</td>
<td>3.413</td>
<td>Working</td>
</tr>
<tr>
<td>13159</td>
<td>2011-11-28 19:00:19</td>
<td>00:00</td>
<td></td>
<td>51.170</td>
<td>3.119</td>
<td>Home</td>
</tr>
</tbody>
</table>

Table 4.4: Example of semantically enriched trajectory with activities.

It is worth noticing that our method is based on the presence of the POI where the activity is carried out in the neighborhood of the stop. In other words, the activities we can infer for a given stop correspond to the activities mapped to the categories of the candidate POIs. When there is a lack of adequate POI - due to an incomplete POIs dataset, ACTIVE could not be able to make any inference. For this reason we evaluate our method only respect to the inferable activities. The notion of inferable activities for a given stop is specified below.

**Definition 15. Inferable activities.** Given a stop $s$ and an activity $a$, we call the activity inferable for the stop $s$ if the category of at least a candidate POI $p$ of $s$ is mapped to the activity $a$. Formally $\exists p$ such that $F_M(cat(p)) = a$.

The inferable activity is the basic measure we need to compute the global accuracy of ACTIVE, defined below:

$$\text{Global accuracy} = \frac{\text{Nr. of correctly inferred activities}}{\text{Nr. of inferable activities}}$$

Note that the accuracy computation takes into account only the inferable activities, that are the ones having at least one candidate POI in the neighborhood associated to the activity to infer. When this condition is not satisfied, this lack of information does not allow our method to do any inference.

Table 4.5 shows the activity inference performances on GPS_Pisa and using both Google Places and Yellow pages datasets, while Table 4.6 shows the results on GPS_Flanders and GPS_TagMyDay using Google Places. We indicate by Tot. $N.$ Stops the number of stops having at least one POI in their neighborhood. Stops with no POIs in the neighborhood are initially discarded; Ave. $N.$ Candidate POI for stop is the average number of Candidate POIs in the neighborhood of a stop. This gives a measure of the possible inference we can do for a stop. Tot. inferable activities represents the number of different activities associated to the candidate POIs in a stop neighborhood; Tot. correctly inferred activities is the number of stops that are annotated with the correct activity; Not inferable activities is the number
Table 4.5: Accuracy measure and statistics of the GPS_Pisa dataset.

<table>
<thead>
<tr>
<th></th>
<th>NoF</th>
<th>WD</th>
<th>WD + DC</th>
<th>NoF</th>
<th>WD</th>
<th>WD + DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tot. N. Stops</td>
<td>1431</td>
<td>637</td>
<td>462</td>
<td>1389</td>
<td>904</td>
<td>618</td>
</tr>
<tr>
<td>Avg. N. Candidate POI for stop</td>
<td>29</td>
<td>5</td>
<td>2</td>
<td>29</td>
<td>21</td>
<td>14</td>
</tr>
<tr>
<td>Tot. inferable activities</td>
<td>1041</td>
<td>358</td>
<td>206</td>
<td>1132</td>
<td>649</td>
<td>306</td>
</tr>
<tr>
<td>Tot. correctly inferred activities</td>
<td>351</td>
<td>188</td>
<td>183</td>
<td>329</td>
<td>214</td>
<td>124</td>
</tr>
<tr>
<td>Not inferable activities</td>
<td>90 (6%)</td>
<td>90 (22%)</td>
<td>256 (51.4%)</td>
<td>237 (17.06%)</td>
<td>255 (28.06%)</td>
<td>312 (51.4%)</td>
</tr>
<tr>
<td>Global Accuracy</td>
<td>26.1%</td>
<td>52.5%</td>
<td>88.09%</td>
<td>28.55%</td>
<td>32.97%</td>
<td>68.13%</td>
</tr>
</tbody>
</table>

Activities relative accuracy:

<table>
<thead>
<tr>
<th>Activity</th>
<th>NoF</th>
<th>WD</th>
<th>WD + DC</th>
<th>NoF</th>
<th>WD</th>
<th>WD + DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>36.5%</td>
<td>31%</td>
<td>30.2%</td>
<td>42.5%</td>
<td>35.2%</td>
<td>75.1%</td>
</tr>
<tr>
<td>Shopping</td>
<td>32.1%</td>
<td>42.9%</td>
<td>0%</td>
<td>10.2%</td>
<td>8%</td>
<td>12.05%</td>
</tr>
<tr>
<td>Daily Shopping</td>
<td>20.2%</td>
<td>31.6%</td>
<td>70.6%</td>
<td>25.6%</td>
<td>31.5%</td>
<td>28.4%</td>
</tr>
<tr>
<td>Eating</td>
<td>35.29%</td>
<td>31.8%</td>
<td>0%</td>
<td>14.7%</td>
<td>21.2%</td>
<td>29.6%</td>
</tr>
<tr>
<td>Leisure</td>
<td>7.6%</td>
<td>15.01%</td>
<td>16.2%</td>
<td>12.8%</td>
<td>19.7%</td>
<td>27.3%</td>
</tr>
</tbody>
</table>

of activities not associated to any candidate POI; Global accuracy is the accuracy measure computed according to Equation 4.3.

We performed three runs of the algorithm using different sets of filters: no filter (NoF), the simple Walking Distance (WD), and the Walking Distance plus the Activity Duration (WD+DC). It is clear how the application of filters improves the global accuracy. As depicted in Figure 4.5, the introduction of filters increases the accuracy, while decreases the number of inferable stops. Activities became not inferable since filters reduce the numbers of admissible POIs, in some cases excluding all the reachable POIs for a stop. This is indeed the expected behavior of filters: excluding POIs in order to prevent inference errors.

The quality of the results of ACTIVE inference algorithm are strictly based on the quality of domain information, i.e. POIs dataset, availability of opening times and typical activity duration. For this reason, we evaluated ACTIVE on GPS_Pisa dataset with two different POIs dataset: Google Places and Yellow Pages. The two datasets have different features, and they mainly differ for the distribution of POIs category (see Figure 4.3). Yellow pages are focused on services companies, and results are highlighting this feature: while the accuracy obtained on detecting Services activity is really high (75%), the other ones are not at the same level. The best performances for ACTIVE are achieved on GPS_Pisa using Google Places dataset: the application of filters increases the Global accuracy from 26.1% to 88.09%.

In order to further analyze the accuracy of ACTIVE, we run some tests comparing the ACTIVE results with a Null Model and a Random Model as reported in Figure 4.5. Although these two measures are usually employed in graphs, here we want to prove that the results of ACTIVE are not random: for this reason we com-
4.4. EXPERIMENTS

<table>
<thead>
<tr>
<th>Tot. N. Stops</th>
<th>GPS_Flanders</th>
<th></th>
<th>GPS_TagMyDay</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NoF W D W + D C</td>
<td>NoF W D W + D C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tot. N. Stops</td>
<td>6596 9 1093</td>
<td>1837 8 1053</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg. N. Candidate POI for stop</td>
<td>21 9 29</td>
<td>22 8 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tot. inferable activities</td>
<td>3578 70</td>
<td>1531 259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tot. correctly inferred activities</td>
<td>919 41</td>
<td>307 255</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not inferable activities</td>
<td>306 (45%) 127(64.4%)</td>
<td>306(16.6%) 1419(61.7%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Accuracy</td>
<td>25.6% 21%</td>
<td>42.04% 43.8%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities relative accuracy:

| Services | 31.5% 23.4% | 48.7% 22.44% |  |
| Shopping | 22.04% 34.4% | 41.3% 53.22% |  |
| Daily Shopping | 0% 0.2% | 0% 32.43% |  |
| Eating | 46.5% 48.9% | 9% 45.12% |  |
| Leisure | 8.3% 11.05% | 35.7% 41.8% |  |

Table 4.6: Accuracy measure and statistics of GPS_Flanders and GPS_TagMyDay, using Google Places.

Figure 4.5: Comparison of the Global Accuracy of ACTIVE w.r.t. a Null model and a Random model using GPS_TagMyDay dataset and Google Places.
pare ACTIVE annotation with two kinds of random annotations. In the Random Model the activity is a pure random selection among all the possible activities. In the Null Model the stop annotation is made by randomly picking one of the activities mapped to the candidate POIs. We see that in the Random Model, the application of the filters cause no effects on the accuracy that remains constant. In the Null Model, instead, the filters affect the accuracy because the model chooses among the activities of the candidate POIs. Still we have the gravity-based assignment that makes ACTIVE outperforming the Null Model. In other words we can say that the Gravity Model, together with the filters, implements a strategy of choice that takes into consideration the real aspects of the human mobility and behavior. In both of the models we used GPS, TagMyDay trajectories dataset because of the more precise annotations, and Google Places POI dataset.

4.5 Exploiting ACTIVE to infer the activity dynamics of a city

As introduced at the beginning of this chapter, the activity annotation can be useful in several urban analysis. We show here a simple case study where we apply ACTIVE to annotate a real and large GPS dataset, and we use the resulting annotated stops to make some simple inferences about the dynamics of a city. The objective here is to give a flavor of the kind of analysis that mobility diaries enable and suggest the potential of having such semantically rich datasets.

<table>
<thead>
<tr>
<th>Properties</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>N. of trajectories</td>
<td>1,382,892</td>
</tr>
<tr>
<td>N. of GPS points</td>
<td>19,536,742</td>
</tr>
<tr>
<td>N. of users</td>
<td>38,259</td>
</tr>
<tr>
<td>Avg. sampling rate</td>
<td>94.376 s</td>
</tr>
<tr>
<td>Avg. distance between consecutive points</td>
<td>1.538 km</td>
</tr>
<tr>
<td>Avg. point-nearest segment distance</td>
<td>8.65 m</td>
</tr>
<tr>
<td>Std. dev. point-nearest segment distance</td>
<td>22.65 m</td>
</tr>
</tbody>
</table>

Table 4.7: Octopisa Dataset properties.

Table 4.7 shows the details of OctoPisa, a raw GPS dataset collected in the city of Pisa (Italy) during the period May-June 2011. The annotation of the stops with activities allows to identify which activities have been performed in the various areas of Pisa in a definite period of time, thus depicting a dynamic map of the city. As shown in Figure 4.6 (A) and (B), the map changes over the time: in the weekend (Fig 4.6 (A)) there are large yellow areas where the main activity is Leisure that are missing during the weekdays. For example the central yellow area crossing the
river in Figure 4.6 (A) is much smaller during the weekdays. This area covers the city center where there are many clubs and shops very busy during the weekend. The eating areas are different because during the weekdays the areas where there are small restaurants and bars where workers and students go for lunch emerge. In the weekend the areas where there are restaurants and pizzerias typically attended for dinner emerge. The red areas of the daily shopping remains almost the same: this is because these stores are open all days. The light orange areas labeled with education emerge only during the weekdays and correspond mainly to areas with University and schools (closed in the weekend).

It is worth to point out that the clustering of the activities annotated to trajectories has a dynamic aspect not covered by the simple clustering the POIs of the city. This is because the activities performed by users may change during the day and during the week like in weekdays and weekend while the POIs categories are purely static.

![Figure 4.6: Results of activities density-based clustering for weekend (A) and working days (B). Each shape represent a dense area of stops annotated with an activity.](image)

### 4.6 Summary

We proposed a method for automatically inferring the activities of users during their movements when tracked by a GPS device. This problem is particularly important since we aim at semantically enriching GPS trajectories with meaningful information that can be useful in several application domains, from transportation science to sociology. We detect the stops and match these stops to the possible visited POIs, in turn mapped to the activity to be done. We assume that the moving objects are
cars, thus making the activity inference more challenging when the car is parked far from the destination place.

The identification of the performed activities is done by ACTIVE, an algorithm which combines several filtering criteria - distance of the POI from the stop, stop duration, time of day - with the probability computed with a Gravity Model. We have evaluated the algorithm for semantic enrichment using three different GPS trajectories datasets already annotated. The accuracy shown by ACTIVE is very good reaching rates up to 88%.

Although these results are promising, we need to highlight the fact that the sparseness of the POI datasets is a major problem, especially when the tracked user is moving in countryside areas, as we noticed for GPS_Flanders dataset. We can however assume that the POI datasets are becoming day-by-day richer therefore in a near future the performance of ACTIVE will increase when combined to richer POI datasets.

Other issues remain to be better investigated, like the mapping between POIs categories and activities: activities performed at a certain place may not be uniquely identified like going to a restaurant with some friends can be an “Eating” activity but also a “Social” activity. Other issues that need deeper investigations are: (1) inference of multiple activities during a stop; (2) the detection and proper enrichment of very short activities like “refueling” and “bring and get” activities; (3) improve the activity inference by considering all the activities sequence of a trajectory and not only isolated stops.
Integrating Domain Knowledge and Mobility Data Mining to boost Routing Planners

Mobility is a complex phenomenon: billions of users traveling across millions of road segments reaching thousands of destination for different reasons. Governing this process is a demanding task, especially for mobility managers; Big Data represents more than a hope for all the actors of this process whom aim to better understand mobility, in order to find solutions to optimize the growth of urban areas. In this context, wisdom of the crowd is a precious source: every driver makes choices to reach the desired destination. Such choices are guided by an optimum that is not presumable by classics distance/time minimization functions. Even though route planners and personal navigation assistant are nowadays helping many drivers on their routing, such devices still rely on suggesting shortest/fastest path to drivers. Users whom know the domain where they are driving, though, decide their route according to their experience. The unprecedented chance offered by Big Data is to learn from this experience to understand which is the optimum drivers are looking for.

Quality of the roads, speed limit, better air quality, presence of cheaper gas stations: manifold can be the features of the optimum path. This research line aim to learn how to optimize car traffic conditions directly by observing who is generating such traffic. The first step towards this optimization is represented by a deep analysis of users’ trajectory w.r.t. to the relative shortest/fastest path. In the following section we are describing the work done in this field. The starting point are user profiles (\cite{?}): an analysis of user common routines highlighted how only 35% of users follow the shortest path, while the majority is choosing an alternative path. The characteristics oh such alternative paths could be considered by routing
planners in their suggestions; developing an experienced route planner is the goal leading future works in this scenario.

Route planners are systems which help users selecting a route between two locations. When providing directions, web and mobile mapping services generally suggest the shortest route. Popular route planning system such as Google Maps, Open Street Maps etc. generate diverging directions using static libraries of roads and road attributes [62]. However, they often ignore both the time at which a route is to be traveled and, more important, the preferences of the users they serve. Since cities are becoming crowded and jammed, smart route planning are gathering an increasing interest. In such a context, a route planner which takes into account users’ preferences [37], and which exploits the crowd expertise w.r.t urban mobility in order to identify the best route, can be more desirable and helpful than an ordinary route planner [26].

A route planner which exploits individual mobility models to improve the planning will have a real advantage from these models only if the users do not follow the shortest path in their systematic movements but deviate from them. Consequently, the target of this work is twofold. The first one is to understand and estimate how much the systematic movements of a user are different from the shortest paths between the origin and destination locations. The intuition is that a user which lives and acts in a certain territory do not automatically select the shortest path. This can happen for many reasons: e.g. traffic conditions, road quality, for passing close to the cheapest petrol station, for avoiding roads with control of speed etc. However, independently from the reasons, if there is a divergence between the systematic route with origin point \( o \) and destination point \( d \), and the shortest route from \( o \) to \( d \) suggested by a route planner, then also other users could benefit from this kind of knowledge which comes from individual expertise on a certain area. This lead to the second and main target: a boosted route planner that, when is possible, proposes as alternative to the shortest path a route which is frequently followed by someone. This planner would be a route planner coming from the wisdom of the crowd in mobility.

By exploiting *individual mobility profile* models [57] and *trajectory map-matching* [18] for a set of users in Pisa and Florence province, we retrieved the systematical movements of the users, named *routines*, and we mapped these routines along a road network. By calculating the shortest path from the origin \( o \) to the destination \( d \) of each routine with an ordinary route planner we obtained the movements a user would have followed when the path is not known and there is not expertise of the area. Then we compared the routines with the corresponding shortest path. Thanks to this analysis we are able to (i) quantify how much human mobility differs from the shortest path and, on the other hand how good can be an approximation of human
mobility made with the shortest paths, (ii) at which level appears the divergence between the routine and the shortest path w.r.t. origin/destination, and (iii) which are the road intersections, areas and flows of movements in which users mobility detaches more in comparison with the shortest paths.

Our experiments show that about 30-35% of the routines follow the shortest paths, while the others follow routes which are on average 7 km longer. In addition, 20% of the routines deviate at the very beginning from the suggested paths. Despite those differences, 60% of the route returned by the route planner would have belonged to the individual mobility profiles. Consequently, even if the analyzed drivers follow routines quite similar to the routes suggested by a route planner, they deviate from them not to minimize the travel distance but for some other unknown reasons. Finally, we found that exists a sort of collective “common sense” among the drivers: when moving from a certain origin to a certain destination nearly all of them deviate in the same area. This indicates that different users which systematically drive along the same roads develop similar individual mobility behaviors.

5.1 Proposed Analytic Model

In the following we describe the analytic model adopted to discover how much the shortest/fastest path can approximate the systematic movements of a user, and how much a route planner could improve its performances by using the wisdom of systematic drivers.

Given a set of users $U$ a and set of road segments $S$, for each user $u \in U$, we calculate the individual mobility profile

$$P_u = \text{getmedoids}(\text{group}(H_u, ms, \varepsilon, \text{dist}))$$

Then for each routine $r_i \in P_u$, we map match the routine on the road network

$$r_i^* = \text{mapmatch}(r_i, S, k)$$

We name map matched individual mobility profile $P_u^* = \{r_1^*, \ldots, r_k^*\}$ the profile of a user $u$ containing the routines mapped on the road network.

We define a route planner

$$\bar{m} = \text{routeplanner}_{\text{type}}(o, d, S)$$

as a function which returns the best path $\bar{m} = \{o, \bar{p}_2, \ldots, \bar{p}_{n-1}, d\}$ w.r.t. the type of search $\text{type} \in \{s, f\}$ (where $s$ stands for shortest and $f$ stands for fastest) on the road segments $S$ where $o$ is the origin point and $d$ is the destination point.
Finally, for each routine $r_i^* = \{o_i, \ldots, d_i\} \in P_u^*$ we calculate the path returned by the route planner $\tilde{r}_i = \text{routeplanner}_\text{type}(o_i, d_i, S)$.

We indicate with $\tilde{P}_u^\text{type} = \{\tilde{r}_1, \ldots, \tilde{r}_k\}$ the routed individual mobility profile of a user $u$ containing the best paths returned by the route planner for the origin and destination of the routines.

Summing up, given a set of users $U$ and their individual history $H_u \forall u \in U$, and the road network segments set $S$ we obtain:

1. Mobility Profiles ($P_u \forall u \in U$) as the result of the application of $\text{group}()$ and $\text{getmedoids}()$ for each $u \in U$, using using the individual history $H_u$ as input parameter;
2. Map-Matched Profiles ($P_u^*$) as the result the application of $\text{mapmatch}()$ for each $r_i \in P_u, \forall u \in U$
3. Route-Planned Profiles ($\tilde{P}_u^\text{type}$) as the result of the application of $\text{routeplanner}()$ on the origin and destination points $o_i, d_i$ of the map matched routines $r_i^* \in P_u^*$ for each $u \in U$. Type can be either Shortest or Fastest path.

Figure 5.1 shows the steps of the analytic mobility model. In the next section we will observe the differences between $P_u^*$ and $\tilde{P}_u^\text{type}$, $\tilde{P}_u^f$. We remark that the shortest path is the path which minimizes the distance, while the fastest path is the path which minimizes the travel time.

5.2 Experiments

In the following we evaluate how much systematic users described by their map matched individual mobility profile $P_u^*$ deviate from the shortest and fastest routes contained in the shortest mobility profile $P_u^s$ and fastest mobility profile $P_u^f$ for the provinces of Pisa and Florence. Moreover we analyze which are the node on the road network $S$, the areas and the flows more affected by deviations.

Dataset

As a proxy of human mobility, we use real GPS traces collected for insurance purposes by Octo Telematics S.p.A\(^1\). This dataset contains 9.8 million car travels performed by about 160,000 vehicles active in a geographical area focused on Tuscany (Italy) in a period from 1st May to 31st May 2011. Figure 5.2-left depicts a sample of the considered trajectories. In our analysis we split geographically the dataset in

\(^1\)http://www.octotelematics.com/it
5.2. EXPERIMENTS

Figure 5.1: Steps of the analytic mobility model. Input: individual history $H_u$, road network segments set $S$. Output: individual map matched mobility profile $P_u^*$, individual shortest/fastest mobility profile $\bar{P}_{u}^{\text{type}}$. $P_u$ is calculated by using the Mobility Profiling functions. Then, the Map Matching module produces $P_u^*$ by using the routines in $P_u$. Finally, $\bar{P}_{u}^{\text{type}}$ is obtained by using the Route Planner on the origin and destination points (highlighted in the red dotted circles) of the routines in $P_u^*$.

Figure 5.2: (Left) A sample of the considered trajectories in Pisa province. (Right) Mobility profiles extracted in Pisa province.

provinces to consider the fact that each area has its type of mobility with characteristics depending on the surface, on the topology and on the number of inhabitants. In this chapter we present the results obtained for the provinces of Pisa and Florence. A user is analyzed in one province if at least one of his/her trajectories passes through that province. In particular we analyzed a subset of 3,000 representative users which have traveled along a total of about 500,000 trajectories. The individual history $H_u$ represents our input data.
Figure 5.3: Distributions of number of trajectories (top - left), number of routines (top - right), routine relative support (bottom - left), trajectories and routines starting time (bottom - right).

Mobility Profiles Analysis

To perform the *Mobility Profiling* step, we used as profiling function \( \text{profile}() \) the clustering algorithm Optics [6], and as distance function \( \text{dist}() \) a function which compares the points distances along the trajectories (or an interpolation of them) and returns the average of these comparisons. In order to obtain sound and reliable routines we performed some preliminary tests to set the best parameters to extract the mobility profiles \( P_u \). We choose \( \varepsilon = 500m \) and \( ms = 8 \) since a routine is a movement that must be repeated a significant number of times during a month. Figure 5.2-right depicts an example of profile extracted in Pisa province modeling the users’ systematic movements.

In Figure 5.3 (top) we can observe the distributions of the number of trajectories and number of routines per user (left and right respectively). All the users selected have more than 150 trajectories and most of them has 160 with an average of about 200 trajectories. Most of the individual mobility profiles \( P_u \) contain 1 – 4 routines. The average length of a routine is about 8.87 km (± 8.96 km of standard deviation), while the average duration is about 20 min (± 12 min standard deviation). In Figure 5.3 (bottom - left) we can observe that most of the routines have a relative support of 0.2 of the trajectories. This means for example that given a user with 160 trajectories and a routine with support equals to 0.2, then that routine is supported by about 30 trajectories, i.e. a trajectory per day on average in the observation period. Finally,
5.3 Deviations Analysis

As first experiment we analyzed the deviation in term of space difference from the routines in $P_u^*$ to those in shortest path $\bar{P}_u^s$, and the deviation in term of time difference from the routines in $P_u^*$ to those in fastest path $\bar{P}_u^f$. In particular, for each user $u \in U$ analyzed, for each routine in $r_i^* = \{o_i, \ldots, d_i\} \in P_u^*$, we calculated the difference in length with the corresponding route in $\bar{P}_u^{s,f}$, i.e. the route $\bar{r}_i$ which starts in $o_i$ and ends in $d_i$. Note that the following results are biased by the route planner used: by applying different route planners the shortest and fastest path obtained could be different.

In Figure 5.4 we can observe the space and time differences distributions. With respect to the shortest path (left in the figure), in both dataset there is a consistent set of routines with space difference equals to zero. This indicates that 30%-35% of the routines (for Pisa and Florence respectively) follow the shortest path suggested by the route planner. The remaining routines differentiate on average of 7 km (see Table 5.1). On the other hand, in Figure 5.4 (right) none of the routines follows exactly the fastest path. Just few routines, i.e. the 10%, follow the fastest routes with less than a minute of difference. All the others differentiate consistently (20 min on average Table 5.1). In addition, we observed that 15% of the drivers in Pisa and 10% of the drivers in Florence have the individual mobility profile exactly equal
965. INTEGRATING DOMAIN KNOWLEDGE AND MOBILITY DATA MINING TO BOOST ROUTING PLANNERS

Figure 5.5: (Left) Distribution of the percentage of road traveled before the routine deviates from the shortest/fastest path. (Right) Ratio of shortest and fastest routes belonging to the clusters of the corresponding routines by varying the minsize parameter.

<table>
<thead>
<tr>
<th></th>
<th>short - space diff</th>
<th>fast - time diff</th>
<th>short - pbd</th>
<th>fast - pbd</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>med    avg    std</td>
<td>med   avg    std</td>
<td>med   avg    std</td>
<td>med   avg    std</td>
</tr>
<tr>
<td>Pisa</td>
<td>02.31  07.16  13.56</td>
<td>07.42  26.92  58.13</td>
<td>07.07  25.14  35.52</td>
<td>07.96  23.19  32.33</td>
</tr>
<tr>
<td>Florence</td>
<td>03.64  10.22  18.45</td>
<td>07.31  19.06  29.90</td>
<td>02.97  07.58  13.54</td>
<td>01.05  01.58  21.58</td>
</tr>
</tbody>
</table>

Table 5.1: median, average and standard deviation of the space difference (km), time difference (min) and relative percentage of road traveled before the deviation (pbd).

to the shortest mobility profile ($P^s_u = \bar{P}^s$). On the contrary, none of the user has all the routines equal to the fastest path, i.e. $P^* = \bar{P}^f$.

In Figure 5.5 left is reported the percentage of road traveled before the deviation (pbd), both for Pisa and Florence. It is obtained by observing after how much $r_i^*$ deviates from $\bar{r}_i$ after the start point $o_i$ (for $\bar{r}_i \in \bar{P}^s_u$ and $\bar{r}_i \in \bar{P}^f_u$). We can notice how 20% of the systematic movements deviate from the shortest/fastest paths at the very beginning. The distribution is a long tailed power law with average percentage before deviation of 7% and 3% for Pisa and Florence respectively (see Table 5.1). Furthermore, how already observed, there is a consistent subset of routines (12-15%) which do not deviate from the shortest path. This do not happens for the fastest path.

Finally, we studied the percentage of shortest/fastest movements which would have belonged to the clusters by varying the minsize (ms) parameter (Figure 5.5 right). We calculated for each user $u \in U$ the trajectory distance (using the same distance function $\text{dist}$ applied for the clustering) between the short/fast paths $\bar{r}_1 \ldots \bar{r}_k$ and the trajectories belonging to the corresponding cluster $M_1 \ldots M_k$. For
minsize = 8 (the value used for the clustering), 60% of the movements returned by the route planner would have belonged to the clusters in both shortest and fastest path. This indicates that the movements returned by the route planner are similar enough to the trajectories belonging to the cluster to be considered part of them. This fact is quite interesting if we consider that the space and time difference between routines and suggested routes are in some cases not negligible, and that the routines generally deviate not far from the origin point.

The conclusion is that systematic drivers generally deviate from the routes suggested by a route planner at the very beginning of their movements, and that in general they do not optimize their travel time but try to minimize the travel distance. However, even the drivers deviate from the short/fast routes, these routes are in many cases very similar to the routines systematically followed.

5.4 Towards an Experienced Route Planner

Before presenting the analysis of this section we remark that routines are movements repeated many times (on average 15 times) during the observation period. Thus, if drivers systematically deviate from what is supposed to be the shortest (or the fastest) path there should be a valid reason. Given a user moving for the first time in a certain area, it could be better for him/her to follow the routines described by “expert driver” instead of the routes suggested by a route planner.

A route planner could be boosted by exploiting the knowledge given by the individual mobility models. Such a route planner should consider various information:
(i) the road intersections where the systematic drivers deviate more, (ii) the areas where those intersections are concentrated, and (iii) the main flows of movement containing deviations. In the following we analyze these three factors to understand their impact and which are their possible uses. Due to lack of space in the following we focus the analysis only on the deviation of the routines against the shortest path.

We refer to the road intersections as deviation nodes. They correspond to the first nodes in the set of road segments $S$ from which the routines in $P_u^*$ deviate from the route in $\bar{P}_u$. To count the number of deviations, instead of considering only the number of routines, we weighted each routine $r_i^* \in P_u^*$ with the number of trajectories that support it. In Figure 5.6 we can observe the deviation nodes in which there are at least 100 trajectories which deviate. The darker and the bigger is a marker, the higher is the number of deviations performed by the routines on that node. As expected, for both cities, the highest numbers of deviation nodes appear into the city center. This confirms the fact that in the city is very difficult to follow the shortest paths. Moreover, in both cities we can observe some particular areas not in the city center (those highlighted in the green dotted squares) with an high number of deviations. They correspond in both cases (i) to the main access points to/from the city center, and (ii) to the roads close to the airports. This is a signal that these areas are probably affected by consistent traffic and the systematic users which have to pass through them prefer longer but less stressful routes.

To analyze the deviations’ areas we divided the territory using a grid with cells of 2.5 km of radius. The heat-map of the deviations is shown in Figure 5.7. The darker is a cell, the higher is the number of trajectories which support the routines deviating there. For these images no filters are applied. The first insight is that
the users acting in province of Florence have an active role even in the mobility of Pisa but the viceversa is not true. Indeed, most of the cells with more deviation in Pisa occur also in the Florence heat-map. From the intersection of the two images emerges that most of the systematic deviations take place along the main road between Pisa and Florence (named SGC Fi-Pi-Li) with a concentration in the area around Empoli. This probably happens because most of the people living in Empoli, which is in province of Florence, go systematically to Pisa for working. For example, instead of following SGC Fi-Pi-Li that is an highway but has a lot of traffic, many drivers could prefer as alternative the road SS67 which runs along SGC Fi-Pi-Li but has much more turns and is not an highway. In Figure 5.8 (left) we report the distribution of the number of cells per routines’ deviations. It is a power low distribution indicating that there are few cells where most of the systematic users decide to take alternative routes. Those are the cells that more than the others the boosted root planner should consider when suggesting the routes which exploit the wisdom of the crowd.

We defined a flow as a triple of cells \((\text{origin}, \text{deviation}, \text{destination})\) where \(\text{origin}\) is the cell origin of the routine, \(\text{deviation}\) is the cell where \(r_i^*\) deviates from \(\bar{r}_i\), and \(\text{destination}\) is the ending cell of the routine. In Figure 5.9 we can observe the flows containing the routines supported by at least 100 trajectories. Through this approach we can observe the main flows along with most of the drivers deviate from the shortest paths. We can observe how in Pisa province there are various flows of entrance to and exit from the city center. The flow with more deviations (the purple biggest arrows) are just under the city center starting from the airport area up to the suburbs. They are surrounded by a large number of in-coming and out-coming flows. We remark that in many cases the deviation from the shortest path appears at the very beginning of the movement. Thus the flows reported mainly highlight the part of the movement after the deviation. Some deviation flows do not have a
mutual reverse flow of the same importance. For these cases the deviation is more evident only in one direction. On the other hand, in province of Florence, the flows in the city center are on average shorter than those outside. In addition, the biggest flows are present in the airport area (big green arrow in the center) and close to the exit of the highways (big blue arrow bottom right and big aqua green arrow in the center). Figure 5.8 (right) shows the distribution of the number of flows per routines’ deviations. Similarly to the cells, the distribution is long tailed indicating a small set of flows where many routines deviates from the shortest/fastest path. A route planner having this kind of knowledge should recommend paths which run along these flows and are similar to the individual routines. Indeed, by applying appropriate weights on the road network segments in $S$ the route planner could provide solutions boosted by the routes systematically followed by expert drivers.

Finally, we analyzed the difference between the flows described above and the flows built using only origins and destinations. In other words given a origin-destination flow $(\text{origin}, \text{destination})$ how many flows $(\text{origin}, \text{deviation}, \text{destination})$ pass through the same deviation? We name this indicator flow similarity in deviation. This value give us a hint of how much a certain deviation is stable along a flow. A flow similarity in deviation of X% indicates the percentage of $(\text{origin}, \text{deviation}, \text{destination})$ flow on the number of origin-destination flows $(\text{origin}, \text{destination})$ which pass through the same deviation cell. E.g. given the following origin-destination flows \{A $\rightarrow$ B, X $\rightarrow$ Y\} and the flows \{A $\rightarrow$ C $\rightarrow$ B, A $\rightarrow$ C $\rightarrow$ B, A $\rightarrow$ D $\rightarrow$ B, X $\rightarrow$ Z $\rightarrow$ Y, X $\rightarrow$ Z $\rightarrow$ Y\}, then the percentage of flow difference is 80%. In our dataset of Pisa and Florence we obtained the following results:
5.5. SUMMARY

- Pisa: 83% (short), 78% (fast)
- Florence: 87% (short), 85% (fast)

These high percentages are a clear signal that the deviation along the various flows are not a matter of individuals, but that are known and subscribed from the majority of the drivers. It is a sort of “common sense” which surprisingly emerges at collective level even though all the mobility models used in the proposed analysis are individual.

Finally, we analyzed the difference between the flows described above and the flows built using only origins and destinations. In other words given a origin-destination flow \((\text{origin}, \text{destination})\) how many flows \((\text{origin}, \text{deviation}, \text{destination})\) pass through the same deviation? We name this indicator flow similarity in deviation. This indicator give us a hint of how much a certain deviation is stable along a flow. The value X% means that X% of the total number of origin-destination flows \((\text{origin}, \text{destination})\) pass through the same deviation cell of the corresponding flow \((\text{origin}, \text{deviation}, \text{destination})\). E.g. given the following origin-destination flows \(\{A \rightarrow B, X \rightarrow Y\}\) and the flows \(\{A \rightarrow C \rightarrow B, A \rightarrow C \rightarrow B, A \rightarrow D \rightarrow B, X \rightarrow Z \rightarrow Y, X \rightarrow Z \rightarrow Y\}\), then the percentage of flow difference is 80%. In our dataset of Pisa and Florence we obtained the following results:

- Pisa: 83% (short), 78% (fast)
- Florence: 87% (short), 85% (fast)

These high percentages are a clear signal that the deviation along the various flows are not a matter of individuals, but that are known and subscribed from the majority of the drivers. It is a sort of “common sense” which surprisingly emerges at collective level even though all the mobility models used in the proposed analysis are individual.

5.5 Summary

In this work we analyzed the deviation of the systematic movements from the shortest and fastest paths suggested by a route planner on a set of drivers in Pisa and Florence provinces. We found that systematic drivers deviate from the routes suggested by a route planner at the very beginning of their movements, and that they generally try to minimize the travel distance more than the travel time. Moreover, we observed that the shortest paths are in many cases very similar to the systematic movements from which they deviate. Through our analytic model we were able to select the areas and the flows with the highest number of systematic deviation.
We discovered that given a flow from an origin $o$ to a destination $d$ nearly all the users which systematically move from $o$ to $d$ deviate in the same area. Our analysis shows that, for some unknown reasons, the traveled systematic movements give to the drivers a feeling that their route is better than the shortest or fastest paths suggested by a route planner. This kind of knowledge can be exploited by a route planner which can weight the cost on the edges with the number of supported trajectories instead of with the length or with travel time. Following this approach, a user which travels for the first time in a certain area could be helped in selecting the route by the wisdom of the drivers which systematically pass there. Also a city manager could gain worth information from our analysis. Indeed, he/she could favor the cars circulation along the routes followed by systematic drivers and improve the others which are in fact not exploited enough.
Part III

Conclusions, ongoing and future works
While main contributions of this thesis regard the semantic enrichment of GPS data through the integration with Open Data knowledge bases, other attempts have been performed in order to evaluate new possible research lines to follow. We posed most of the efforts on the exploration of GSM and Twitter data, aiming to develop an automatic process to infer the characteristics of an event exploiting the relative georeferenced tweets. Here an event is defined as a high and not usual presence of people into a certain area (i.e. the coverage area of one or more GSM towers) at a certain timestamp, while tweets referenced with the same spatio-temporal features have been considered to infer the characteristics of the event. Another ongoing work is the improvement of results obtained in Chapter 5: the investigation on drivers’ routing policies is worth further experiments, since it could reveal meaningful patterns useful both for mobility managers and drivers themselves. Finally, we propose some side works on slightly different types of Mobility Data. We apply or know-how on mobility data analysis to the almost unexplored world of Sports Analytics. Data Revolution is also happening in the sports science field, both for professionals and amateurs: the availability of a huge amount of data from moving players or moving cyclists leads us to a first analysis and validation of such data with data mining techniques.
Detect and recognize events by combining CDR and Twitter data

Although CDR data (see Section 1.3) are collected for billing purposes, they also contain valuable knowledge. Due to the penetration of GSM phones, CDR data represent the most reliable proxy for human mobility. Applications are manifold: from urban classification according to users’ profile (e.g. Sociometer ([56])) to Traffic Flows Estimation ([34]). This research line, still to improve, aims to explore the richness of geo-referenced data by analyzing and studying the traces users left while performing their activities. Starting from a massive dataset of Call Data Record, i.e. the whole billing information of ”ELT Estonian Telecom Company”, we can identify where events are happening through a data mining process. We define an event as a higher than usual aggregation of people. Thanks to the richness of our dataset, we know exactly the number of people connected to every GSM antennas. Once identified special events, we want to add semantics from more structured dataset. We are actually collecting data from Twitter API, Foursquare, Google Places and estonian Web site, with the goal of developing a framework for automatic events identification and classification. This work will also assess the value of different data sources, by highlighting their ability to describe what is happening in a monitored spatial region.

The limits of CDR knowledge extraction arise when we aim to add deeper semantics to our analysis. For instance, once detected an event in terms of people presence, we would know the reasons leading that particular aggregation of people. Attempts towards this direction have been done by involving Social Media Data in the knowledge extraction process. In [1], authors proposed a framework to identify events from relative georeferenced tweets by means of the number of related keywords, and an estimation of both tweet start time and geographic location. Chen et al. [13] exploit the spatio-temporal information of photos in Flickr to detect events. Since the
lack of statistical significance of Social Media data is well known (e.g.: geoferenced tweets are 1% of total tweets, Twitter has “only” 302 million active users), the next step is to combine a highly significant data source with a semantically significant one.

CDR data have been also used to detect events. Following we will focus on a particular task, that is Peak Detection, introduced in [56]. In this scenario, a Peak represents a not usual presence of users over a particular and spatially defined area. A common implementation of Peak Detection process consists in different steps: (i) define the geographical area to analyze and to partition it into a set of regions, then (ii) define a series of Timeslots based on a time-frame parameter, i.e. months, days, or hours. Once defined those parameters, it is possible to create (iii) a spatio-temporal grid where each observation of an input dataset can be assigned to one of the cells. The number of observations that fall in a cell defines its density. Input data is partitioned into two sets: a training dataset and a test dataset. Here an observation is a single Call Detail Record regarding an user \( u \) connected to a GSM tower. For both datasets the spatio-temporal grid of densities is computed. The first is used to compute the densities of a typical period for each region. The second dataset is then compared against such typical period in order to detect significant deviations. Based on the densities obtained for each region and each time slot over the training dataset, an expected density value is computed for each region, by averaging the densities measured at the same time slot of all the periods in the time window covered by the dataset. For instance, we might obtain an expected density for each pair (region, hour of the day), i.e., 24 values for each region, assuming 24 one-hour time-slots. Then, for each region and each time-slot, the corresponding density is compared against its expected value: if the difference is significant, an event of form (region, weight, time slot) is produced, representing its spatio-temporal slot and a discretized measure (weight) of how strong was the deviation. In particular, events are detected on the base of three parameters:

- a granularity of deviations, expressed as a percentage relative to the expected density;
- a minimum relative deviation, also expressed as a percentage, used to select significant deviations;
- an absolute minimum deviation, expressed as an integer number, used to discard extreme cases with very low densities.

The weights used in defining events will be multiples of the granularity, and an event for a region and a time slot will be built only if the deviation of its density with
regard to the corresponding expected density is larger than the absolute minimum deviation and in percentage is larger than the minimum relative deviation.

6.1 Case Study

We performed a preliminary analysis to develop a framework for Event Detection using GSM and Social Media. We used a dataset of 150 millions Call Detail Records over the whole country of Estonia, covering the period December 2014-March 2015. After an anonymization task, data have been aggregated according to a temporal slice of one hour. Then, for each GSM tower (and its relative location), we have:

![Total Calls hourly aggregated](image)

**Figure 6.1:** Distribution of the number of calls over the observed period

**Definition 16.** A *Hourly Aggregation(t,h)* for a GSM tower *t* and a hourly timestamp *h* provides the total number *calls*\(_{\text{total}}\) of calls/sms recorded, the number *calls*\(_{\text{foreign}}\) of calls performed by foreign users, and the number *calls*\(_{\text{onceamonth}}\) of occasional calls from tower *t*.

In our analysis we focused on the number of occasional calls *calls*\(_{\text{onceamonth}}\), defined as follows:
Definition 17. An **Occasional Call** for a tower $t$ and a hourly timestamp $h$ is a call performed by an user who connected to $t$ at most one day over the last month.

![Occasional Calls hourly aggregated](image)

Figure 6.2: Distribution of the number of occasional calls over the observed period

Figure 6.2 shows the temporal distribution of occasional calls. Apart from new year’s eve, that is easily recognizable, it is not possible to infer any other event. As highlighted above, the spatial component of CDR data plays a key role in the Peak Detection process. In this scenario, to define a peak, we computed the average ($\mu_t$) and the standard deviation ($\sigma_t$) of hourly occasional calls for each GSM tower $t$. Then, for each time slice we selected those GSM tower where $\text{calls}_{\text{onceamonth}} = \mu_t + 2 \times \sigma_t$ ($95^{th}$ percentile). Furthermore, towers recording a peak during the same time slice have been spatially aggregated by means of a density clustering, hence obtaining the geographical area where events are occurring.

CDR data are capable to define the importance of an event in terms of space, time, and population. In Fig. 6.3 (top) are represented clusters of peaking GSM tower in Tallinn during new year’s eve. As expected, all almost the neighborhoods are spotting, since the event is involving the whole city. In Fig. 6.3 (bottom) we added tweets downloaded from Twitter Stream API related to the peaking areas and time slices.
Figure 6.3: Occasional user presence at Tallinn City Center during New Year’s Eve (top) and related georeferenced tweets (bottom)
Detection of event such as New Year is kind of useless (as it is for Christmas, etc.): the semantic added by tweets is obvious. Hence, it is worth to firstly analyze the significance of Twitter Data w.r.t. to our GSM dataset. As depicted in Fig. ?? the number of georeferenced tweets regarding the whole country is not high. We performed a more meaningful validation of Twitter dataset, by observing the correlation of the number of tweets for each cell with the relative number of occasional calls occurred. For each cell, we computed the Spearman correlation coefficient between the time series of tweets number and the one related to call numbers, to check whether the volume of tweets collected within a GSM Tower is related to people presence in that area. In Fig. 6.5 there is the result of this preliminary analysis: in Tallinn Center, the maximum correlation coefficient is 0.3, observed in the city mall area. It is not exactly the expected result: a coefficient of 0.3 indicates a low correlation, meaning that tweets and GSM peaks (and then events) are not directly related. This means that a simple inference of events semantic from tweets could be extremely biased.

Next steps towards an Event Detection framework will require further analysis, involving additional datasets such as the one provided by the EU project ASAP (1).

Figure 6.4: Correlation between tweets and call volumes for each GSM tower.

---

1http://www.asap-fp7.eu/ - A Scalable Analytics Platform (ASAP)
Figure 6.5: Distribution of georeferenced tweets over the observed period
DETECT AND RECOGNIZE EVENTS BY COMBINING CDR AND TWITTER DATA
7

Extract and exploit Mobility Routing Pattern

Once knowledge about traffic have been mined from Mobility Data and enriched through the integration with additional knowledge bases, new chances to explore the extracted information arise. Although mobility pattern express frequent users’ movements in terms of origins and destinations of their trips, or traversed regions of interests [27], more sophisticated models are required to get deeper information about users’ behavior while moving. Once users trajectories are translated into sequences of road network segments –as done in chapter 3 –, further analysis could be performed regarding, for instance, users’ routing policies. To reach a destination, a driver has to take decisions about which is the best suitable route. Such decisions could also be required during the trip, to overcome a traffic jam or particular road network characteristics that are not evident anyway (e.g. pavement conditions); in this chapter we present some ongoing works focused on the analysis of users behavior while traversing the road network.

7.1 Leveraging Road Network Hierarchy through Big Data

The possibility to transform GPS trajectories into sequences of road segments opened up new perspective for our mobility analysis. One of the feasible evolutions of results presented in Chapter 5 regards the analysis of Road Network structure. Roads are planned and built according to a hierarchy that nowadays is not data driven; the question which is worth to answer is: are main roads according to drivers choices the same as the main roads designed by urban planners? We started to face this not easy nor obvious challenge. We consider the evaluation
of shortest/fastest path and their relative popularity among users as the first step of a data driven Road Network analysis. Hence, the aim of this work is to identify those locations in a road network where drivers significantly deviate from the shortest/fastest path. This analysis is based on the results of our Time-Aware map matching method introduced in Sec. 3. Once a dataset of GPS trajectories has been transformed into a set of paths, expressed by means of road segments sequences, we compared such real trajectories with the shortest path of the respective origin and destination. We define a deviation to occur on a road segment $r$, if (i.) $r$ is a segment which is part of the real trajectories $t$ but not part of the respective shortest path shortest($o_t, d_t$), where $o_t$ and $d_t$ are origin and destination for trajectory $t$. Furthermore (ii.), $r$ has to be the first segment of $t$ which does not satisfy (i.).

Once deviations has been computed, we have performed some preliminary analysis. As showed in Fig. 7.2, majority of deviations occur in the beginning of the trip, while Fig. 7.1 highlight the distribution of deviations in terms of absolute spatial difference w.r.t. the shortest path. It is clear how paths chosen by drivers are quite similar to the shortest path in terms of distance traveled. Further analysis will involve correlation between distance from users location (e.g. home, work) and deviations ratio, i.e. check if the distance of the destination from user’s home and, consequently, the minor knowledge of the road network, generates more deviations.

In Fig. 7.3 the main deviation nodes for Tuscany road network are highlighted. Most of deviations occur on highways nodes, meaning that users tent to prefer
popular and nominally fastest roads to the shortest ones. A temporal analysis, i.e. the same indexes computed by comparing the fastest path instead of shortest, will be part of this work, in order to provide a deeper understanding of road hierarchy.

7.2 Evaluate Routing Policies according to the Wisdom of the Crowd

An exciting further challenge in the wide scenario of Human Mobility Understanding is the comprehension of how users travel across a Road Network. In this ongoing work, we started to answer a simple question: *what if we would only follow other users to decide how to reach a destination, without having any knowledge on road segments length and travel time?*

Even though we have the proper tools to set the stage, a routing evaluation has to consider many features to be accurate. First of all, as we mentioned, user experience plays a key role. Routing strategies followed by residents can not have the same influence w.r.t. commuters or tourists. Furthermore, the presence of Point of Interest – and their relative popularity – affects traffic and trips. As a preliminary analysis, we used results of our Time-Aware map matching method, i.e. all the map-matched GPS trajectories, to define a new function to minimize in order to create a new routing policy. Given a road network $R$, for each $r \in R$ we assign $load(r) = \frac{1}{|T_r|}$,
where $|T_r|$ is the number of trajectories that contains the segment $r$. Then, given two nodes $s$ and $t$, by searching for the path that minimize the function $load()$ we obtain the most frequented path between $s$ and $t$. A further refinement consists in a data pre-processing, by filtering trajectories according to an users experience index. Such measure is computed as:

$$exp(o, d, u) = \frac{|\forall t \in T_u, origin(t)=s/destination(t)=d|}{|T_u|}$$

where $u$ is a user, $o$ an origin node and $d$ a destination node. Due to sparsity of our dataset, we have considered 1-$km^2$-sized cells as origin/destination locations, instead of single nodes. Cells are identified by an index $i \in \mathbb{N}^2$. In other words, user experience represents how important are the trips from $o$ to $d$ in the travel history of user $u$. The first comparison we made regards the difference between paths obtained by minimizing the function $length()$ (i.e. the classic shortest path) and the ones returned by the minimization of the function $exp()$ defined above. First results are depicted in Fig. 7.4: for all the trips that are starting and arriving in cell, we plotted the distribution of the difference between shortest path and most frequented path. On x-axis there is the user experience in terms of a filtering percentage, i.e. each bin $x$ represents those trajectories $T_U$ with origin $o$ and destination $d$ belonging to all users $u \in U$ where $exp(o, d, u) = x \times 100$, $\forall u$. As we can observe, the increasing of user experience let the error decrease. Experienced users tent to follow the shortest path better than occasional users.

This work is still a preview, further experiments have to be performed in order to assess the quality of a routing policy based on the so called wisdom of the crowd.
For instance, other features could be considered, such as users’ most visited locations and/or the relative radius of gyration. Exploiting users routing strategies is a challenging task to conduct an empirical study on how users use the road network, or, in general, what are the features that characterize the optimum path for an user.

Figure 7.4: How experience/wisdom of user influences the difference between chosen and shortest path
7. EXTRACT AND EXPLOIT MOBILITY ROUTING PATTERN
A particular type of Mobility Data: sports analytics

The increasing availability of Mobility Data have been already widely discussed in the previous chapters. Mobility Data, though, could not only refer to the common meaning of human mobility. Football players, for instance, are also producing spatio-temporal data while playing on a pitch, and cyclists are recording lot of information about their workouts: distance traveled, elevation gain, heart-rate and so on. Sport Data mining is capturing the attention of the scientific community, and the proliferation on available data is a further chance to stress and test well known data mining and machine learning methods. The understanding of a football team strategy or the evaluation of a training program are complex task to model, and this hardness is defining new challenges that data scientists are going to take. As a side work of this thesis, we conducted some data exploration attempts –[17], [15] – in the area of Sports Data Mining.

8.1 Analyzing Cyclists’ performance with Data Mining techniques

The recent emergence of the so called online social fitness constitutes a good proxy to study the patterns underlying success in sport. Through these platforms, users can collect, monitor and share with friends their sport performance, diet, and even burned calories, giving an unprecedented opportunity to answer very fascinating questions: What are the main factors that shape sport performance? What are the characteristics that distinguish successful sportsmen? Can we characterize the role of social influence on fitness behavior? We present the results of a study conducted
on a sample of 29,284 cyclists downloaded via APIs from the social fitness platform Strava.com. We defined two basic metrics: a measure of training effort, that is how much a cyclist struggled during the workout; and a measure of training performance indicating the results achieved during the training. Analyzing the relationship between these two metrics, an interesting result immediately emerges: at a global level, there is no correlation between effort and performance. This means that, in general, the performance is not simply a function of training: two athletes with the same level of training have different performance. However, by deeply investigating workouts time evolution and cyclists training characteristics, we found that athletes that better improve their performance follow precise training patterns usually referred as overcompensation theory, with alternation of stress peaks and rest periods. Studies and experiments related to such theory, up to now, have always been conducted by sports doctors on a few dozen professionals athletes. To the best of our knowledge, our study is the first corroboration on large scale of this theory, mainly confirming that engine matters, but tuning is fundamental.

### 8.1.1 Cyclists’ Mobility Data

Strava.com\(^1\) is a social fitness platform where cyclists and runners all over the world can share, compare and compete with each others personal fitness data via mobile and online apps. It makes fitness a social experience, providing motivation and camaraderie even if athletes are exercising alone. When a user signs up for an account, he/she can download a free app for tracking her rides or runs. Once he/she completed a ride or run, data are automatically sent to Strava.com. By the app, a user is able to find popular and competitive segments nearby the place where she is located, and participate in virtual rides or races with other users. Moreover, the platform also includes a social dimension to the experience, allowing users to follow friends and their activities, join clubs and create new ones.

Using Strava.com APIs, we downloaded a set of features regarding a sample of the 29,284 users from around the world (Table 8.1 briefly describes our dataset). We selected, from the total 29k riders, a subset of 1,868 users with the following characteristics: i) they have more than 30 training sessions in the period November 2012 - May 2013 (25 total weeks), in order to select users that are active throughout the period of observation; ii) they live and perform workouts in the northern hemisphere, in order to obtain similar seasonal weather conditions and include countries with the stronger cycling diffusion and tradition (for instance France, Italy, and USA). As Figure 8.1 shows, the number of training sessions per week is not constant during the period of observation: in cold seasons (from November to February) we

\(^1\)http://www.strava.com/
8.1. ANALYZING CYCLISTS’ PERFORMANCE WITH DATA MINING TECHNIQUES

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total users</td>
<td>29,284</td>
</tr>
<tr>
<td>Users in period Nov 2012 - May 2013</td>
<td>1,868</td>
</tr>
<tr>
<td>Total Km traveled</td>
<td>$4.8023 \times 10^7$ km</td>
</tr>
<tr>
<td>Total elevation gained</td>
<td>$4.58612 \times 10^7$ m</td>
</tr>
<tr>
<td>Total training time</td>
<td>195,625h 53m 43s</td>
</tr>
<tr>
<td>Estimated power production</td>
<td>11.796472 MW</td>
</tr>
<tr>
<td>Total training session analyzed</td>
<td>88,632</td>
</tr>
<tr>
<td>Average training session per user per week</td>
<td>1.89</td>
</tr>
</tbody>
</table>

Table 8.1: Description of the Strava dataset

observe a law training activity, but as the weather conditions start to improve the number of active users grows more and more.

For each training session of each rider we have the following information: moving time, traveled distance, elevation gain, heart rate stream with a sample rate of 3 seconds, estimated average watts produced. Furthermore, for each session we have the crossed competitive segments. In particular, every segment has its elapsed time, elevation gain, VAM (mean ascent velocity), and estimated average watts (Table 8.2 summarizes the available information). We use data from segments to evaluate the performance of the rider, while, as explained in the following section, data from training session – i.e. heart-rate stream – has been used to define the intensity of the workouts.

![Figure 8.1: Number per week of cyclists performing workouts in the period of observation.](image)

In order to study the workouts time evolution and detect those producing the best benefits, we used well known Cycling Science metrics such as Training Suffer Score (TSS) and Mean Ascent Velocity (VAM) to extract, for each rider and each week of training, the following information:
Available information

| Training sessions | moving time  
traveled distance  
elevation gain  
heart rate stream  
estimated average watts |
|-------------------|------------------|
| Crossed segments  | elapsed time  
elevation gain  
mean ascent velocity (VAM)  
estimated average watts |

Table 8.2: Description of available training session and segment information

- Sum of TSS of every training session performed during the week;
- VAM variation achieved in the week, computed as the difference between the average VAM of all the segments faced during the week and the same value of the previous week;
- Estimated Watts variation achieved in the week, calculated using the average estimated watts for all the segments faced during the week. Though the Strava.com watts index is an estimation from other data (rider’s weight, road climb factor, etc), we use it as a comparison with results obtained with VAM index.

We considered a period of observation of 25 weeks, from November 2012 to the end of April 2013, for two main reasons. First of all, since Strava’s APIs have been closed to public access on June 1st, this did not give us enough time to collect data regarding rides performed in May. Secondly, the chosen period is the first and important half part of a cycling season, that usually starts with the winter initial workouts and it is oriented to the first days of May, when the most important amateur races take place. The same, with different scale factor, applies for professional cyclists: Giro d’Italia usually begins in the first week of May.
8.1. Analyzing Cyclists’ Performance with Data Mining Techniques

<table>
<thead>
<tr>
<th>Clustering</th>
<th>Cluster</th>
<th># Users</th>
<th>Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSS</td>
<td>$T_1$</td>
<td>929</td>
<td>low trained</td>
</tr>
<tr>
<td></td>
<td>$T_2$</td>
<td>552</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>$T_3$</td>
<td>387</td>
<td>would-be profes.</td>
</tr>
<tr>
<td>VAM variation</td>
<td>$V_1$</td>
<td>418</td>
<td>low trained</td>
</tr>
<tr>
<td></td>
<td>$V_2$</td>
<td>543</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>$V_3$</td>
<td>481</td>
<td>would-be profes.</td>
</tr>
<tr>
<td>Watts variation</td>
<td>$W_1$</td>
<td>314</td>
<td>low trained</td>
</tr>
<tr>
<td></td>
<td>$W_2$</td>
<td>643</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td>$W_3$</td>
<td>480</td>
<td>would-be profes.</td>
</tr>
</tbody>
</table>

Table 8.3: Description of resulting clusters

Once we characterized each athlete with her weekly TSS, VAM variation and Watts variation, we split the whole population in clusters of similar users with respect to each single metric. To perform this task, we used the K-MEANS clustering algorithm [44] with $k = 3$ as the number of clusters parameter, chosen as the number of clusters produced by the density-based DBSCAN clustering algorithm [44]. Table 8.3 summarizes the characteristics of the resulting clusters for TSS clustering, VAM variation and Watts variation clustering.

8.1.2 Data Driven learning of the best training strategy

The TSS based clustering (Figure 8.2, left) split the riders in a realistic manner. The three behaviors highlighted are easy to find across the “peloton”, as the population of bike riders is called between the domain experts. Cluster $T_1$ (blue dashed curve) identifies the “low trained” rider, who does not have so much time (or will) to perform long training sessions; Cluster $T_3$ (green dashed curve) represents the opposite “would-be professional” rider, which dedicates a big part of her life to cycling. Between them, in cluster $T_2$ (red solid curve) there is the “normal” rider, with training workload increasing as well as spring arises and weather conditions improve.

VAM and Watts variation clustering show similar trends and seem to be coherent with the previous results, with an important evidence that confirms the main motivation of this work. In these plots, the three clusters show an initial striking difference: a starting condition where riders’ “engines” seem to have different “capacity”. Going forward in time, the values almost start to converge, indicating us the efficiency of the three different approaches to training. With respect to the VAM variation clustering (Figure 8.2, center), the most interesting is the behavior shown by cyclists in cluster $V_2$: even though in the beginning the average increment is lower than cluster $V_3$, it finally reaches the highest peaks during the important part of the season. In a virtual battle of the clusters, cluster $V_2$ would win, highlighting
how fundamental are quality of training and workout planning for a training season. In an individual aerobic sports like cycling, where the focus is on the individual physiological parameters, not only the “engine” matters, but type and quality of the training plan play a fundamental role.

In Figure 8.3 (left) we show the TSS time evolution of two users extracted from the most interesting and competitive clusters $V_2$ and $V_3$ (VAM variation clustering). Such users are the closest to the centroid of the respective clusters, and present very different training characteristics: while the $V_3$ user starts with high intensity since the beginning of the winter, user $V_2$ seems to adopt a more focused periodization of the training. Indeed, she starts with a low-stress winter preparation and increases the intensity during the season, with two resting periods where the intensity decreases (red solid curve in Figure 8.3 left). This allows the user to tolerate harder training when needed, that is in spring season, as we mentioned before. The benefits of such kind of “periodized” training plan are evident from Figure 8.3 (center), where the VAMs of the athletes are shown.

Figure 8.3 (right) shows the relationship between TSS and VAM for the clusters, introducing a refined “intensity-of-training” index. In fact, the ratio TSS/VAM could be viewed as a measure of energy consumption, a way to enrich the information about the intensity of training given by TSS. The $V_3$ user has a lower average
Figure 8.3: Comparison of two users from clusters $V_2$ and $V_3$ in terms of TSS (left), VAM (center) and TSS/VAM ratio (right).

consumption ($\mu_{V_3} = 0.60$) than the $V_2$ user ($\mu_{V_2} = 1$), highlighting the trend of the $V_2$ user to train harder. In fact, the higher $V_2$ consumption is in contrast with the higher average TSS of cluster $V_3$, making the TSS/VAM an accurate parameter to evaluate the quality of training. Furthermore, the TSS/VAM curve of $V_2$ user has the highest peak around the 15th week, followed by a low-consumption period. Looking at the VAM plot of $V_2$ (Figure 8.3, center), user’s performance starts to significantly grow just around the same period. High stress peak, resting, performance increasing: this is the exact physiological process known as overcompensation. The main result obtained is an evidence: quality of training influences performance. To the best of our knowledge, this is the first experiment on this field made on this large number of athletes, mainly confirming that “engine” matters but “tuning” is fundamental.

### 8.2 Network Theory to model Football Players Mobility

Sports analytics in general, and football (soccer in USA) analytics in particular, have evolved in recent years in an amazing way, thanks to automated or semi-automated sensing technologies that provide high-fidelity data streams extracted
from every game. In this paper we propose a data-driven, network-based approach and show that there is a large potential to boost football analytics. We construct networks based on observational data from a football game (passes, goal attempts, crosses, etc.) and compute a set of indicators from these networks. We observe a strong correlation among individual and combined values of such indicators and the success of team, and therefore perform a simulation on the four major European championships (78 teams, almost 1500 games). The outcome of each game in the championship was replaced by a synthetic outcome (win, loss or draw) based on the network indicators computed for each team. We found that the final rankings in the simulated championships are very close to the actual rankings in the real championships, reaching a correlation of 0.85 between simulated and real rankings. Our results are surprising given the simplicity of the proposed indicators, suggesting that a complex systems' view on football data has the potential of revealing hidden patterns and behavior of superior quality.

8.2.1 Football Data

We have data about all the games of the four main major European leagues (Italy, Spain, England, Germany) in the season 2013/2014, provided by the TIM company\(^2\) (official sponsor of Italian league). The Italian, Spanish and English leagues have 20 teams each playing 38 games, the German league has 18 teams each playing 34 games. In total our dataset stores information about 1,446 football games. A football game is described by a sequence of events on the field (passes, crosses, assists, goal attempts and goals), with a mean of 450 events per game and a total of \(\approx 600,000\) events in our dataset (Table 8.4). Each event consists in the following information: the timestamp of the event, the player who generated the event, the position of the ball on the field when the event is generated, the position of the ball on the field when the event ends, the outcome of the event (successful or unsuccessful). For example in Table 8.5 the event “pass” identifies a successful pass made by the player Messi at position (65.4, 20.2) of the field; the event “goal attempt” indicates an unsuccessful goal attempt made by the same player.

Since each event specifies the destination zone on the field, these data allow us to reconstruct the ball trajectory and the position of the ball during the goal attempts (Figure 8.4). However we do not have direct information about the destination player, i.e. the player to which the pass is directed. We infer this information by sorting all the events by time and making a spatial agglomeration: given player \(A\) who generates a pass event \(p\) toward zone \((x, y)\) at time \(t\), if player \(B\) generates an

\(^2\)http://www.tim.it/home
Table 8.4: Size of our dataset of football games.

<table>
<thead>
<tr>
<th>Season 2013/2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>leagues</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>Italy, Spain, England, Germany</td>
</tr>
<tr>
<td>teams</td>
</tr>
<tr>
<td>78</td>
</tr>
<tr>
<td>20 Italy, Spain, England - 18 Germany</td>
</tr>
<tr>
<td>games</td>
</tr>
<tr>
<td>1,446</td>
</tr>
<tr>
<td>360 games per league in average</td>
</tr>
<tr>
<td>events</td>
</tr>
<tr>
<td>600,000</td>
</tr>
<tr>
<td>450 events per game in average</td>
</tr>
</tbody>
</table>

event from zone \((x, y)\) at time \(t' > t\) we assume that player \(B\) is the destination player of the event, otherwise we discard the event \(p\).

Table 8.5: Example of events during the game Real Madrid-Barcelona (Spanish league).

<table>
<thead>
<tr>
<th>event</th>
<th>time</th>
<th>player</th>
<th>origin</th>
<th>destination</th>
<th>outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>pass</td>
<td>17:24</td>
<td>Messi</td>
<td>(65.4, 20.2)</td>
<td>(67.8, 44.1)</td>
<td>successful</td>
</tr>
<tr>
<td>attempt</td>
<td>18:12</td>
<td>Messi</td>
<td>(98.4, 15.0)</td>
<td>(118.7, 15.0)</td>
<td>unsuccessful</td>
</tr>
<tr>
<td>assist</td>
<td>45:00</td>
<td>Bale</td>
<td>(78.56, 12.2)</td>
<td>(78.5, 36.0)</td>
<td>successful</td>
</tr>
<tr>
<td>cross</td>
<td>78:54</td>
<td>Ronaldo</td>
<td>(102.2, 30.1)</td>
<td>(115.7, 30.2)</td>
<td>successful</td>
</tr>
</tbody>
</table>

8.2.2 A football game as a network

We represent the behavior of a team during a game by a passing network, i.e. a weighted directed network where nodes are entities on the field and edges represent ball displacements between the two entities. Formally, the passing network of a team \(A\) is a weighted directed graph \(G_A = (V, E)\), where \(V\) is the set of nodes and \(E\) is the set of edges. The weight function defined on the edges \(\omega : E \rightarrow \mathbb{N}\) returns the number of passes between the two nodes of team \(A\). While edges represent ball displacements during the game, nodes can represent different entities like players or zones of the field. We consider two types of passing networks: (i) the player passing network \(P\), where nodes are players and edges are passes between players; (ii) the zone passing network \(Z\), where nodes are zones (obtained by splitting the field into cells of size \(1m \times 1m\)) and an edge \((z_1, z_2)\) represents all the passes performed by any player from zone \(z_1\) to zone \(z_2\). To clarify the concept of player passing network, let us consider the network in Figure 8.5 extracted from a portion of a game of Juventus (Italian league). The 11 nodes represent the team’s players, the edges indicate the passes between the players, the size of the edges indicate the number of passes between the players. We observe that node 7 (Andrea Pirlo) has the highest
degree being the most central player in the network. Figure 8.6 shows the zone passing network extracted from a portion of a game of Barcelona (Spanish league) with 21 nodes (zones of the field) where edge are proportional to the number of passes between the zones. Here we observe the presence of a dominant zone (in the bottom part of the figure) where most of the passes take place.

The player passing network and the zone passing network are abstractions of the team’s behavior that synthesizes the passing history during a game in a compact model, but can be constructed efficiently from the event data.
Figure 8.4: Ball trajectories (red lines) and goal attempts (blue and yellow dots) of Juventus and Roma in a game of the Italian major league. Each trajectory is formed by all the origin and destination points of team’s passes, assists, crosses and goal attempts.
Figure 8.5: A player passing network extracted from a portion of a game of Juventus (Italian league). Node are team’s players and are located on the field according to their mean position during the game. Edges represent passes between players, and the size of an edge is proportional to the number of passes between the players. The player passing network is an oversimplification of a football game, but it provides an immediate insight on the role of a player, and on the distribution of passes among the players. We observe for example that node 7 (Andrea Pirlo) is the most central node in the network.
Figure 8.6: A zone passing network extracted from a portion of a game of Barcelona (Spanish league). Node are zones on the field, edges represent passes performed by any players between two zones, the size of an edge is proportional to the number of passes between the zones. The zone passing network is a richer representation than the player passing network, since it has more nodes and edges.
8. A PARTICULAR TYPE OF MOBILITY DATA: SPORTS ANALYTICS
9

Conclusions

In this thesis we investigated the potential of Mobility Data and their ability to reveal pattern useful both for single users and mobility managers, owing to the availability of GPS, GSM, Social Media and Open Data sources. We faced the challenge firstly focusing on GPS data and traffic estimation, providing efficient methods to extract traffic knowledge from low sampled GPS trajectories, discovering where drivers move. Then, we proposed an algorithm to answer the question about why do they move, by developing a probabilistic model based on the combination of GPS trajectories and domain knowledge such as a Point of Interests dataset. We pulled up the semantic level by using the results of our map matching algorithm as a further data source to understand how drivers route across road network. Ongoing there are works regarding the exploration of new scenarios: the focus will be on GSM and Social Media Mobility Data, even though we are going to follow other research lines as well.

Works discussed in this thesis pointed out the value of Data Integration as a means for achieving a richer level of semantics on Mobility Data. The future developments of research on these lines will therefore involve the integration of several different data sources regarding the same subject (an event an its features, an user and his/her behaviors, etc.). However, the accessibility of such kind of data is expected to become one of the hardest challenges to face in the near future. Indeed, Big Data are already recognized as the most valuable asset for large ICT companies. This scenario could be the principal limit for the evolution of Data Science: data produced by billions of users of all over the world are stored into data centers of few internet companies, and the exclusive property of those inestimable data sources belong to companies who gathered them and not to the users who actually generated them.

On the other hand, while a complete understanding of human mobility requires
the possibility to access such kind of massive datasets regarding Mobility Data, users should be allowed to know and decide about everything is done with their data. Coherently with this perspective, in his formalization of the “New Deal on Data” [47] Pentland introduced the concept of Personal Data Store, a system that allows users to keep all the rights over their data, including that of selling them to the company they choose and getting back revenues.

The future of Mobility Data analysis is still uncertain. Many challenges are still open, while requirements are increasing. The World Economic Forum is promoting a change of perspective towards an user-centric model for personal data management ([21],[22],[23]). The basic idea is to introduce high levels of transparency and full control for the user on the life cycle of his personal data (e.g., collection, storage, processing, sharing). Thus, the user has to have an active role into a righteous and fruitful ecosystem based on personal data. Although architectures such as Personal Data Store are evolving, a privacy-safe environment where applying Data Analysis method is still missing. Mobility Data Integration will play a key role in this field. First of all, it could reveal the real value of users’ data, letting them aware of what they are providing in terms of information. Secondly, the requirement of users’ involvement will foster the develop of analysis tools meaningful both for scientists and users. In such a scenario, drivers will be aware that their GPS data are going to be used to optimize the structure of road networks, thus letting users and researchers share a common goal. This collaboration between “real world” and research is the best possible achievement that Data Science can get, making this era of Data Revolution one of the most interesting ever.
Bibliography


