Graz University of Technology

Master Thesis

Trip purpose imputation from mobile phone trajectories using an Artificial Neural Network

conducted at the
Institute of Neural Engineering
Graz University of Technology, Austria

in co-operation with
Invenium Data Insights GmbH
Graz, Austria

by
Bernhard Stadlbauer, BSc

Supervisors:
Assoc.Prof. Dipl.-Ing. Dr.techn. Reinhold Scherer
Univ.-Prof. Dr.-Ing. Martin Fellendorf

Graz, February 28, 2019

Zusammenfassung

Um den ständig steigenden Bedürfnissen an die Mobilität der heutigen Gesellschaft gerecht zu werden, bedarf es einens optimalen Ausbaus des Verkehrsnetzes. Eine aktuelle und repräsentaive Beurteilung des Ist-Zustandes ist notwending und um eine angemessene Planung durchführen zu können. Durch die weite Verbreitung von Mobiltelefonen bietet eine Standortanalyse, basierend auf den Trajektorien bestehend aus Positionen von Mobilfunkdaten-Events, eine Momentaufnahme der individuellen Mobilität. Diese Arbeit zielt auf die Fahrtzweckerkennung einzelner Trajektorien ab, da diese die Basis der Planung bildet. Ziel ist es, ein System zu entwerfen, zu testen und zu validieren, welches in der Lage ist eine rohe Mobilfunktrajektorie in eine Serie von Aktivitäten, wie zum Beispiel Zuhause zu sein oder Einkaufen zu gehen, umzuwandeln. Die größten Herausforderungen sind die schlechte zeitliche und vorallem die schlechte räumliche Auflösung der Mobilfunk-Positionen (je nach Gebiet 100m-5000m) sowie das Fehlen von annotierten Trainingsdaten.

Zur Verbesserung der zeitlichen und räumlichen Auflösung wird in dieser Arbeit eine neue Vorverarbeitungsmethode für Mobilfunktrajektorien in Form eines Extended Kalman Filters mit variabler Messkovarianz vorgestellt. Ein Neuronalen Netzwerk, dass auf Basis von Daten einer öffentlichen Umfrage trainiert wurde, wird dann zur Automatisierung der Aktivitätenerkennung stationärer Segmente verwendet. Lange Aktivitäten wie zu Hause oder am Arbeitsplatz zu sein werden mit einer Genauigkeit von mehr als 78% erkannt. Kürzere Aktivitäten (z.B. Einkaufen oder Tennisspielen) lassen sich jedoch nicht zuverlässig unterscheiden. Die räumliche Auflösung der aktuellen Mobilfunktechnologie ist der limitierende Faktor. Die Ergebnisse dieser Studie deuten darauf hin, dass die vorgeschlagene Methode es ermöglicht nützliche Informationen über für die Stadtplanung relevante Fahrtzwecke zu gewinnen.

Schlüsselwörter: Fahrtzweckerkennung, Aktivitätenerkennung, Mobilfunktrajektorien, Neuronales Netzwerk, Big Data

Abstract

Optimal expansion of the transport network is important to meet the ever-increasing demands of mobility. Timely and representative assessment of the current state is necessary in order to be able to carry out appropriate planning. Mobile phones are widely in use. Analysis of mobile phone location data provides a snapshot of individual mobility and allows effective planing. This thesis aims at detecting trip purposes as they are the base for planning decisions. Its goal is to design, test and validate a data processing pipeline capable of converting a raw mobile phone location trajectory into a sequence of characteristic activities such as being at home, at work or going shopping. The biggest challenges are the poor temporal and especially spatial resolution (accurate to 100m-5000m depending on the location) of phone trajectories and the lack of labeled training data for supervised learning. To train machines to segment a trajectory and assign activities, labels describing the activity of training segments are needed.

To enhance temporal and spatial resolution, this work introduces a new preprocessing method for mobile phone trajectories in the form of an Extended Kalman Filter with variable measurement covariance. A neural network based classifier trained on public survey data is then used to automate the labeling of stationary trajectory segments.

Long activities such as being at home or at work can be detected with an accuracy greater than 78%. Shorter activities (for instance going shopping or playing tennis), however, cannot be reliably distinguished . The spatial resolution of current mobile network technology is the limiting factor. The results of this study suggest that the proposed method allows extracting useful information on certain trip activities needed in urban planning.

Keywords: Trip-purpose imputation, Activity recognition, Mobile phone trajectories, Artificial neural network, Big data

Contents

1 Introduction **13**
1.1 History . . . 13
1.2 Trip purpose . . . 14
1.3 Mobile phone trajectories . . . 14
1.4 Trip purpose imputation . . . 17

2 Methods **19**
2.1 Data collection . . . 19
2.2 Processing pipeline . . . 22
2.2.1 Background . . . 22
2.2.2 Preprocessing . . . 27
2.2.3 Trip purpose imputation . . . 32

3 Results **41**
3.1 Processing pipeline . . . 41
3.1.1 Speed filter . . . 41
3.1.2 Kalman filter . . . 41
3.1.3 Segmentation . . . 43
3.1.4 Trip purpose imputation . . . 45
3.2 Application Example: One day of data . . . 46

4 Discussion **51**
4.1 Processing pipeline . . . 51
4.1.1 Speed filter . . . 51
4.1.2 Kalman filter . . . 51
4.1.3 Segmentation . . . 51
4.1.4 Trip purpose imputation . . . 52
4.2 Application Example: One day of data . . . 52
4.3 Improvements . . . 53
4.4 Outlook . . . 53

Acronyms

AI Artificial Intelligence. 25

ANN Artificial Neural Network. 25, 26, 28

CDR Call Detail Recording. 8, 9

CNN Convolutional Neural Network. 25

CPU Central Processing Unit. 28

CReLU Concatenated Rectified Linear Units. 26, 33

CSV Comma Separated Values. 14

EKF Extended Kalman Filter. 17

GD Gradient Descent. 27

GPS Global Positioning System. 7–11, 17–19, 34

GPU Graphics Processing Unit. 28

IMSI International Mobile Subscriber Identity. 9, 11, 14

KDE Kernel density estimation. 19

LSTM Long Short Term Memory. 25

MLP Multi Layer Perceptron. 25

MNL Multinomial Logic Model. 25

OEU 'Österreich unterwegs'. iii, 7, 13, 15, 20, 23

POI Point of Interest. 23, 25

ReLU Rectified Linear Unit. 26

RNN Recurrent Neural Network. 25

RTS Rauch-Tung-Striebel. 19

SGD Stochastic Gradient Descent. 27

UKF Unscented Kalman Filter. 17

WIT What-If Tool. 28

1 Introduction

The overall population of the world is still growing, which is not only putting pressure on Earth's natural resources, but also increasing stress in many other domains. Creating enough living space, waste management or commuting traffic are examples of challenges we are already facing, and that are expected to grow in the future.

Urban planning has ever been demanding and most decisions are made based on the experience of domain experts. However, due to the increasing computational power available, it is now possible to make educated choices, not just based on experience, but on insights created from huge amounts of data collected and processed near real-time. Demands with regard to mobility can be identified and suitable alternatives offered. In a concrete example, one could look at all commuters that drive into a city, check where they live and provide a better public transport system based on this information.

This work targets the base of these analyses by proposing a novel method to obtain bespoken insights from big data. It tackles the problem of imputing a trip purpose, such as traveling to work when the input is a mobile phone trajectory. Similar to a Global Positioning System (GPS) track, those trajectories are created by mobile phone providers, estimating the location of a phone user throughout the day. In this work trip purposes will be expressed in Courier style, for instance `Home` or `Work`.

Even though every mobile phone provider creates vast amounts of location data as a byproduct of running their regular service, the main problem of supervised machine learning, having labeled training data, persists. The high spatial uncertainty of the recorded locations ranging from a maximum accuracy of a few hundred meters in a city to multiple kilometers in the countryside pose an additional problem. To have ground truth data, this work collected a sample of about 250 days, where not only the mobile phone trajectory of a participant was retrieved, but they were also asked to record their GPS track and complete journal of all their activities.

The goal of this work was to train and validate a state of the art machine learning classifier. Contrary to many preceding studies [17, 37], this work tries to provide a clear and reproducible measure of the annotation quality.

1.1 History

To get insights into human mobility, the gold standard has long been questionnaires. Participants were asked to fill in details about their daily trips. These forms were then manually entered into computers to be able to analyze them. As one can imagine, this is very costly and does not scale well for bigger sample sizes. So only small and sometimes not representative samples were taken, which were then extrapolated to the whole population. Also, only temporal patterns were recorded, as there was no way of accessing spatial information.

With the rise of home computers and public access to the internet around the 1980s, data acquisition was simplified as participants could now directly fill out forms, eliminating the manpower needed for digitizing the samples. However, the problem remained that subjects had to manually fill out forms, leaving room for errors whilst transferring the data.

Another big information source was added when GPS receivers became publicly available. This meant that, in addition to the temporal records, it was now possible to record spatial records for deeper analysis of the datasets. With mobile phones becoming capable of recording GPS tracks, this got even easier. There is no longer a need for an extra device to acquire tracks, which eliminates the possibility of forgetting the GPS receiver. An excerpt of studies using GPS datasets can be found in Table 1.1.

The most recent development are apps that track a person's movement, preprocess the GPS data, and present the user with an optimized form of annotating the collected data, lowering the burden of participating as much as possible.

This thesis takes this idea one step further. By annotating mobile phone data, participants would only need to carry their phone. This is achieved by developing a system that can automatically annotate anonymized trajectories. It does not only simplify the whole process, but also scales well for big sample sizes and other countries as long as there is sufficient cell phone coverage. Datasets containing mobile phone trajectories are harder to come by, as one relies on cell phone providers to give out data, leading to far less studies using them. Table 1.2 shows an excerpt of studies using these datasets.

1.2 Trip purpose

In the classical sense, a trip purpose is defined by the activity performed at the end of a trip. As an example, if one travels from his or her home to work, the trip purpose would be `Work`.

As those purposes can be split indefinitely finely, a finite set of possible activities has to be defined. The present work uses a definition based on a survey done by the Austrian government, therefore the trip purposes used there were taken. An overview can be seen in the data collection section (Table 2.1).

1.3 Mobile phone trajectories

The raw data used for extracting trip journals are anonymized mobile phone trajectories. Mobile phone data, also called Call Detail Recordings (CDRs), can be recorded and displayed in different ways. A single CDR contains, for example, call start time, end time, duration and can further contain location information, which is important for this application. Based on the processing done by the phone provider, this can be the location of the cell tower used (as used in [17,20]), or the provider can estimate a location based on simple triangulation or more sophisticated movement models (used in [1,37]).

As the field of insights from phone data is rapidly growing at the moment, companies specialize in preprocessing raw mobile phone events from a cell phone carrier to get better spatial estimations. Simplified, their algorithms blend information about cell towers such as their locations as well as how the antennas on those towers are oriented to improve the location estimation. All mobile phone trajectories collected in this work are preprocessed by a proprietary algorithm from VIAVI Solutions Inc. (San Jose, CA, USA).

If one collects all recorded datapoints of one person and sorts them by their timestamp, this is then called a trajectory. In Figure 1.1 one can see an example of a raw mobile phone trajectory as well as the matching GPS track taken from the dataset recorded for this thesis.

Anonymization CDRs are usually connected to the user by the use of the International Mobile Subscriber Identity (IMSI). This identifier is linked to the SIM card of a user. To protect the privacy of users, the IMSI is replaced by a random identifier before it leaves the cell phone carrier. To ensure that a person cannot be tracked over longer periods of time, this identifier is changed every 24 hours. This is important, as for analysis purposes, as no features that would include a time span of more than a day, for instance 'number of weekly visits', can be extracted from the data.

Table 1.1: Overview of previous GPS based datasets. Translated from [19]

Region	Year of data collection	Device	Sample size	Collection period	Used in
Atlanta (USA)	1990	Car	30 Vehicles	3 days (2 waves per week)	[38]
Borlänge (Sweden)	2000-2002	Car	186 Vehicles	more than 30 days	[39]
Netherlands	2007	GPS device	1104	7 days (15 waves)	[3]
Minneapolis (USA)	2008	Car	187 Vehicles	13 weeks	[21, 22]
Nagoya (Japan)	2008	Smartphone	30 subjects	5 weeks	[10]
Greater Cincinnati Region (USA)	2009	GPS beacon	Total of 4.133 ways	3 days	[32]
Shanghai (China)	2009	GPS beacon	50 subjects	3 days	[6]
New York City (USA)	2010	GPS beacon	49 participants	1 to 5 days	[4]
Wuhan City (China)	2010	GPS beacon	10 participants	1 month	[16]
Waterloo (Canada)	2011	GPS beacon	108 cyclists	2 weeks	[36]
Zurich (Switzerland)	2012	GPS beacon	156 participants	7 days	[26]
Einhoven and Rotterdam (Netherlands)	2012-2013	GPS beacon	329 participants	3 months (3 waves)	[7]
Flandern (Belgium)	2013	Car	28 Vehicles	1 Year	[9]
Shanghai (China)	2013 - 2015	Car	28 Cars	7 Days	[40]
Vienna (Austria) and Dublin (Ireland)	2014	GPS beacon and smartphone	37 (Austria), 17 (Dublin)	8 Weeks	[25]
Stockholm (Sweden)	2015	Smartphone	171 participants	7 days	[2]

Table 1.2: *Overview of previous mobile phone based datasets. Translated from [19]*

Region	Year of data collection	Sample size	Collection period	Used in
France	2007	23 million (Orange Network)	154 Days	[17]
Europe	2009 - 2011	80 users of a European Cell phone providers	Longer than year	[20]
Boston (USA)	2010	2 million Users of different US cell phone carriers	2 months	[1]
Vienna (Austria)	2012	1 million users of an Austrian cell phone provider	2 weeks	[37]

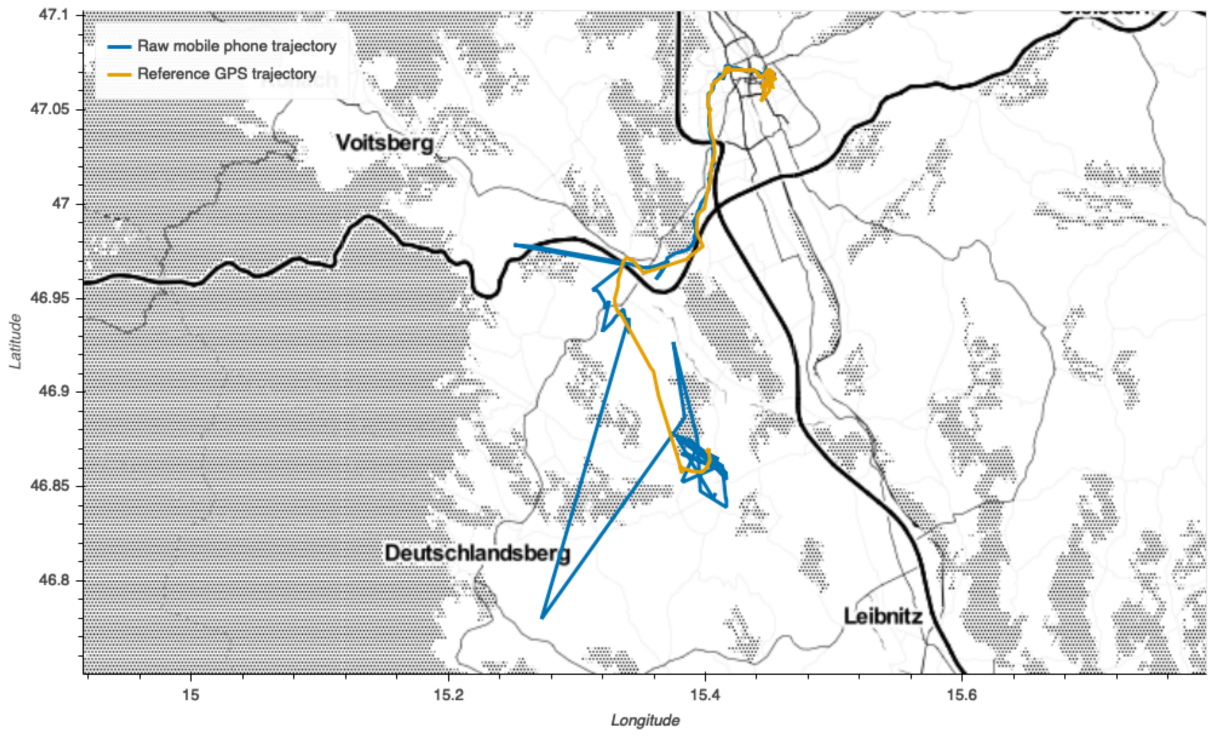

Figure 1.1: Raw Trajectory. 21.03.2018, subject 2

1.4 Trip purpose imputation

When one tries to impute trip purposes from raw trajectories, the process is twofold. In a first step, called preprocessing, the trajectory is cleansed (see noisy raw trajectory; Figure 1.1) and clustered into stationary and moving segments. In a second step, all stationary sections are annotated with an activity.

As one can imagine, the quality of the labeling is highly dependent on the preprocessing. There is currently no gold standard on how to preprocess raw phone trajectories, so this work implements and validates the complete processing pipeline. The following paragraphs describe the main purpose as well as the challenges faced by each block:

Preprocessing has two main goals. First, one tries to remove as much noise induced by the measurements as possible. Secondly, the trajectory has to be segmented into stationary and moving segments. The main challenge in preprocessing is the calculation of the best possible reconstruction of the true path. This is especially true for mobile phone trajectories, as they have a neither a good spatial nor temporal resolution.

Annotation is the task of assigning an activity to every stationary segment. As can be assumed, the detail to which activities can be differentiated strongly depends on the spatial and temporal resolution of the segments. The main difficulty lies in the distinction of activities based on features from the segments.

To summarize, the goals of this work are:

1. Collection of a reference dataset to validate the proposed methods.
2. A new method to smooth trajectories with poor spatial and temporal resolution.
3. Use of modern pattern recognition algorithms to automate annotation.

2 Methods

The goal of this work is to design, implement and validate a system that can perform trip purpose imputation on raw mobile phone trajectories. Similar to previous work, this is done by first splitting the trajectories into stationary and moving segments. Afterwards, features are extracted from the stationary segments, which are then fed into a classifier to predict the activity.

The current chapter is split into two main parts. The first provides an overview of the data collected for this thesis. The second part describes the complete processing pipeline needed to convert a raw mobile phone trajectory to a series of annotated segments.

2.1 Data collection

GPS can be seen as the ground truth for mobile phone trajectories, therefore both were collected to test and validate the developed methods. Participants were asked to record their daily trips using Modalyzer [15], a smartphone-based app which records GPS tracks. The app also provides automatic segmentation, meaning that trips are already segmented in stationary and moving segments. Moving segments are automatically tagged with a mode of transport. Participants then had to fill in a trip purpose for the stationary ones. As an example, if someone would take the bus to get from their home to work, the trip purpose would be `Work` and the mode of transport would be `Bus`.

In addition to the GPS tracks and the trip diaries, the corresponding mobile phone trajectory was retrieved. This means that participants had to have a phone from the partnering mobile phone carrier, as the carrier would then selectively revert the anonymization of the IMSI for the chosen phones. If a participant had a different carrier, or did not want to allow deanonymization on their phone, a test phone was provided.

The dataset was recorded in two waves. The first wave included 8 users and ranged from March 2018 to May 2018; however, not all users were participating all the time. In total, 94 days were recorded in those months. This first wave was conducted as part of the master thesis by A. Lechner [19]. The second wave started mid-September and lasted until November 2018 and data was collected withing the frame of the current thesis. The details about the study can be found in [19]. The second wave was recorded with seven entirely new participants. Their mean age was 30.86 years, two of them were female the rest male. Combining the two waves changed the overall mean age to 33.12 years and overall there were 11 male and 5 female participants.

In total 223 days were recorded, resulting in a total number of 1248 stationary segments in the dataset. As participants also recorded `Transit`, which is the change of transportation mode, these segments have been removed. A split by the trip purpose can be found in Table 2.1.

In the current work, all proposed methods are applied to a trajectory recorded on the 21.03.2018 by subject 2 (see Figure 1.1). This trajectory was chosen as it is located both in the city as well as on the countryside. Additionally, it features multiple transportation modes. As a reference, Figure 2.1 shows all the nearby cell towers. The recorded trip journal can be seen in Table 2.2.

A second example (subject 15 on the 01.11.2018, Figure 2.2) is used to show a worst case scenario. This sample was recorded entirely on the countryside. When comparing the reference

Table 2.1: Number of stationary segments grouped by the trip purpose for the data collected in this thesis. `Drop off` *could for example be bringing a child to school or picking it up again.* `Education` *applies to all pupils and students when they are in school.* `Leisure` *can be anything from sports to going out in a bar.* `Private` *are activities such as going to the doctor or a hairdresser.* `Visiting` *is used when one is invited to somebody else's home.* `Work related` *is used to denote work-related tasks not happening at work*

Trip purpose	Number of Segments	Percent
Drop off	14	1.49
Education	7	0.74
Home	450	47.77
Leisure	126	13.38
Other	15	1.59
Private	46	4.88
Shopping	85	9.02
Visiting	24	2.55
Work	172	18.26
Work related	3	0.32

GPS track with the mobile phone trajectory, it is especially interesting to see that the mobile phone data-points are sometimes mapped to the east, whereas in reality, the subject has never been there. Further investigation revealed that this happened mainly during the stationary segments at home. In the Figure, `Home` would be the point most north on the trajectory. Therefore, it can be concluded that there is a bias and not just random noise when measuring the location. Also, looking at Figure 2.2 it looks as if the company providing the location estimates would map somewhere close to the cell towers.

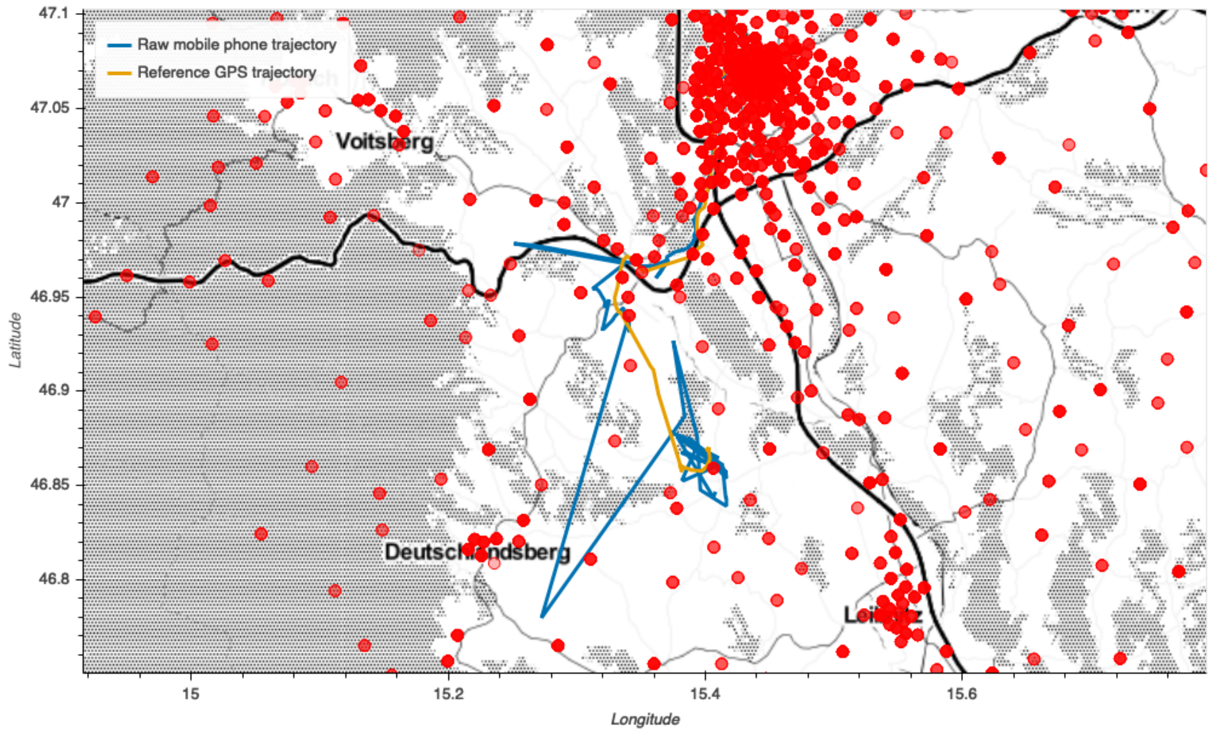

Figure 2.1: Raw trajectory. Red dots indicate the position of cell towers. 21.03.2018, subject 2

Table 2.2: Recorded trip journal, 21.03.2018, subject 2. All times are UTC

Start time	End time	Stationary/Moving	Transportation mode	Activity
10:00	15:12	Stationary	-	Work
15:12	15:19	Moving	Walking	-
15:19	15:29	Moving	Tram	-
15:29	15:32	Moving	Walking	-
15:32	15:38	Stationary	-	Transit
15:38	16:15	Moving	Train	-
16:15	16:21	Moving	Car	-
16:21	24:00	Stationary	-	Home

Table 2.3: Recorded trip journal, 01.11.2018, subject 15. All times are UTC

Start time	End time	Stationary/Moving	Transportation mode	Activity
00:00	13:28	Stationary	-	Home
13:28	13:31	Moving	Car	-
13:31	13:44	Moving	Walking	-
13:44	14:23	Stationary	-	Private
14:23	14:35	Moving	Walking	-
14:36	14:39	Moving	Car	-
14:39	24:00	Stationary	-	Home

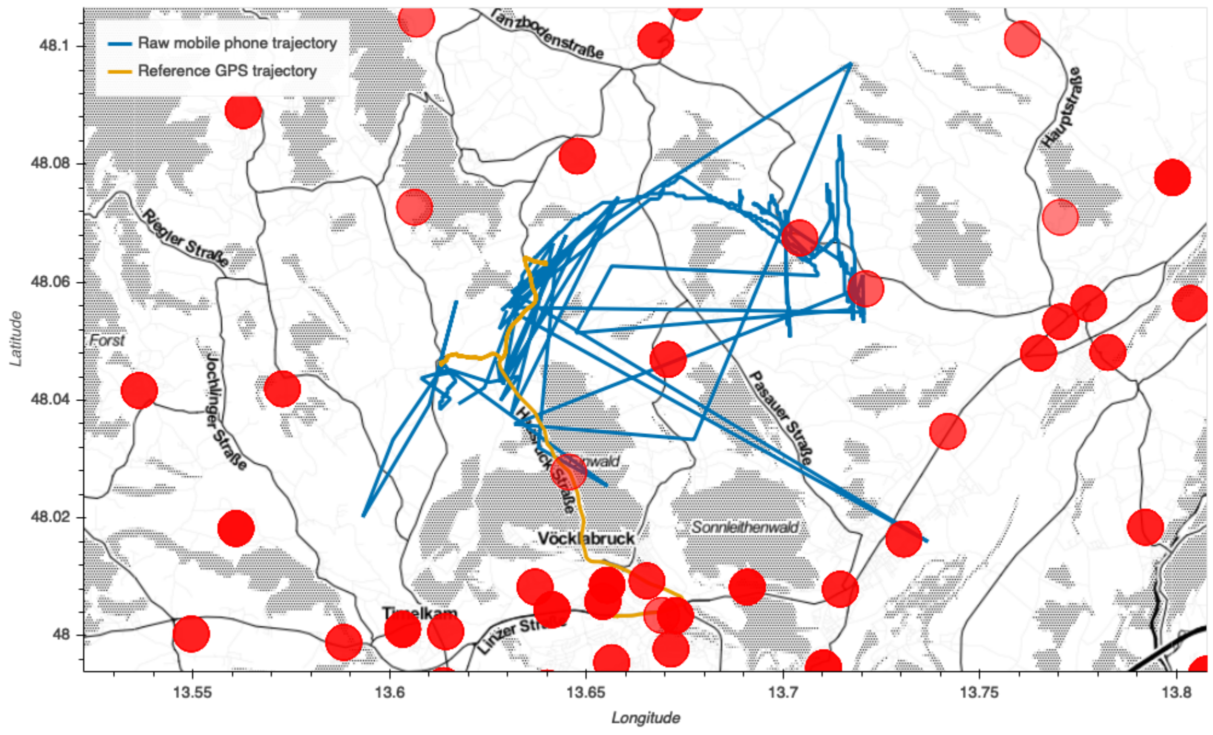

Figure 2.2: Raw Trajectory on the countryside. Red dots indicate the location of cell towers. As a space reference, the diameter of the red dots is relative to the scale of the map (compare also Figure 2.1). 01.11.2018, subject 15

2.2 Processing pipeline

The following section is split into three main parts. The first provides some background information on Kalman filters as well as Artificial Neural Networks (ANNs). The second describes all steps taken to preprocess trajectories. The third and final part deals with the annotation of the found stationary segments. An overview of all steps described can be found in Figure 2.3. First the data is cleaned by a set of rules (Data selection) then observations with a too high speed are removed (Speed filter). The resulting trajectory is smoothed with a Kalman filter and then segmented into stationary and moving segments. For the stationary segments features are then extracted and fed into an ANN to annotate them with a trip purpose (Annotation).

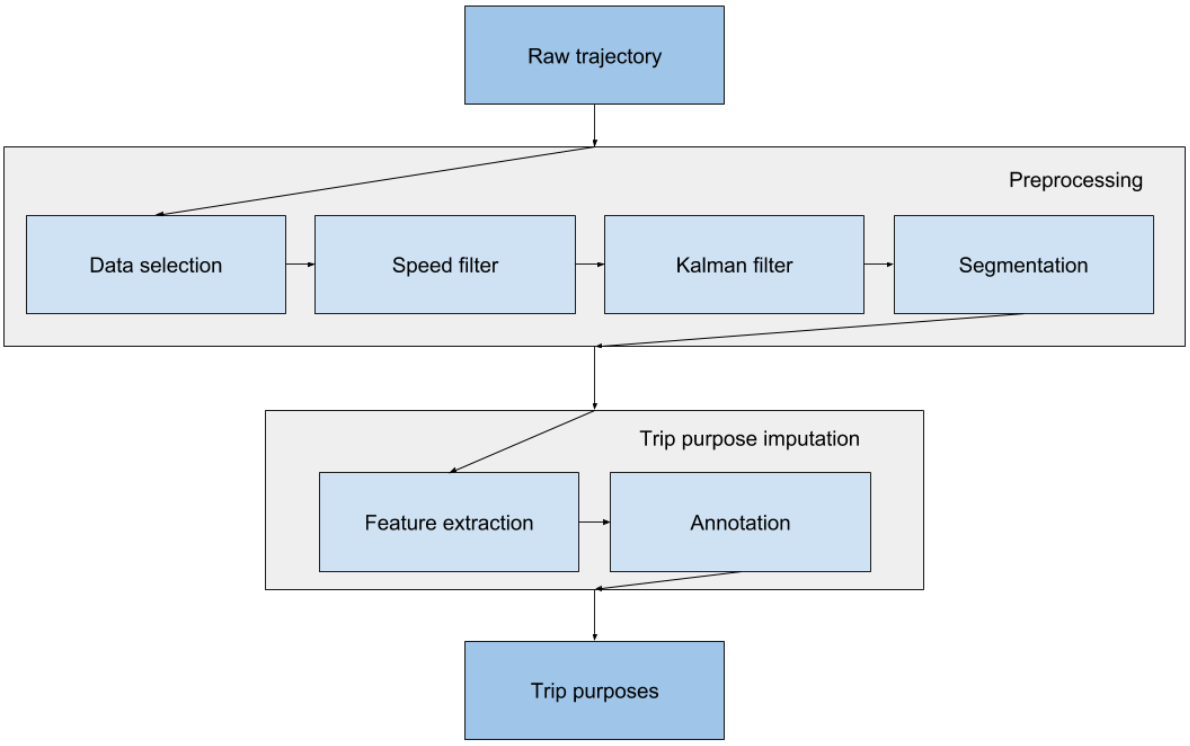

Figure 2.3: Overview of the processing for mobile phone trajectories

2.2.1 Background

Extended Kalman filter

The Kalman filter was first introduced in the 1960s by R.E. Kalman [18] and is a general method to smooth measurement series, taking the uncertainty of observations into account. It uses its internal filter state $\hat{\mathbf{x}}_k$ in conjunction with an internal dynamic model (written as matrix $\mathbf{F}_k$) to estimate the next step.

$$\hat{\mathbf{x}}_k = \mathbf{F}_k \hat{\mathbf{x}}_{k-1} + \mathbf{B}_k \mathbf{u}_k \tag{2.1}$$

One can also include an external input $\mathbf{u}_k$ controlled by the control-input matrix $\mathbf{B}_k$; however there is usually no external input when measuring spatial trajectories. Therefore, it will not be considered in the following equations.

Also, as can be seen in Equation 2.1, the update step is performed using matrix $\mathbf{F}_k$ allowing only for linear changes in state. As most real-world models are non-linear, the extended Kalman filter has been proposed. It assumes that $\hat{\mathbf{x}}_k = f(\hat{\mathbf{x}}_{k-1})$. As one will see further down, $\mathbf{F}_k$ is still needed for the propagation of the covariance, and is thus it is estimated using a linear approximation at point $\hat{\mathbf{x}}_k$:

$$\mathbf{F}_k = \frac{\delta f(\hat{\mathbf{x}}_k)}{\delta \hat{\mathbf{x}}}\bigg|_{\hat{\mathbf{x}}_k} \tag{2.2}$$

The full Kalman filter is then usually written in two steps, predict and update. Predict increases the internal state as well as the internal covariance matrix $\mathbf{P}_k$ using $\mathbf{F}_k$.

$$\hat{\mathbf{x}}_k = f(\hat{\mathbf{x}}_{k-1}) \tag{2.3}$$

$$\mathbf{P}_k = \mathbf{F}_k\mathbf{P}_{k-1}\mathbf{F}_k^{\mathrm{T}} + \mathbf{Q}_k \tag{2.4}$$

Where $\mathbf{Q}_k$ is the process noise matrix, added to allow for small errors in the model. The scaling of $\mathbf{Q}_k$ is one of the critical hyperparameters that need to be set from the outside.

The update step then takes the next measurement into consideration, correcting the estimate $\hat{\mathbf{x}}_k$ and the internal covariance $\mathbf{P}_K$ based on the uncertainty $\mathbf{R}_k$ of the measurement $\mathbf{z}_k$:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k \tag{2.5}$$

$$\mathbf{S}_k = \mathbf{R}_k + \mathbf{H}_k\mathbf{P}_k\mathbf{H}_k^{\mathrm{T}} \tag{2.6}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\mathbf{S}_k^{-1} \tag{2.7}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k + \mathbf{K}_k\tilde{\mathbf{y}}_k \tag{2.8}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{2.9}$$

where $\mathbf{H}_k$ is a function that converts the internal state $\hat{\mathbf{x}}_k$, which can have an arbitrary number of dimensions, into the measurement dimensions of $\mathbf{z}_k$ (which are usually limited), so $\mathbf{z} = \mathbf{H}\hat{\mathbf{x}}$. As a note, $\mathbf{K}_k$ is usually known as Kalman gain, as it is the factor that determines how much to trust the measurement compared to the estimate made using the internal state.

Artificial neural networks

This work uses an ANN (sometimes also called Multi Layer Perceptron (MLP)) to classify the activity of the segments created in the previous sections. The basic idea of a simple perceptron has already been proposed as early as 1957 [29], which also led the the first big hype in machine learning. This abruptly ended when Minsky and Seymour published a book about the limitations of the perceptron in 1969 [24], leading to the first Artificial Intelligence (AI) winter.

The main drawback was not being able to train a network containing multiple perceptrons. This changed with the introduction of backpropagation for neural networks in 1986 by Rumelhart and colleagues [30], leading to the next big advances. Still, standard backpropagation was not able to train deep networks (deep meaning that they have many hidden layers) and the hardware limited the application of the algorithm on huge datasets. This led to yet another AI winter. The reason neural networks and especially deep learning is so popular in the present is mainly due to new network structures such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), new training algorithms such as Adam or Adagrad and new types of neurons that can hold information from the past such as Long Short Term Memory (LSTM) cells.

With the dataset in this work being fairly small (ca. 300,000 samples), and also having only 7 features per sample (compared to the 30,000 features when training with 100x100 pixel color images), no deep network structure should be required. An example of the general structure can be seen in Figure 2.4. When one increases the number of hidden layers as well as the number of hidden units in those, this increases the maximum complexity that can be learned.

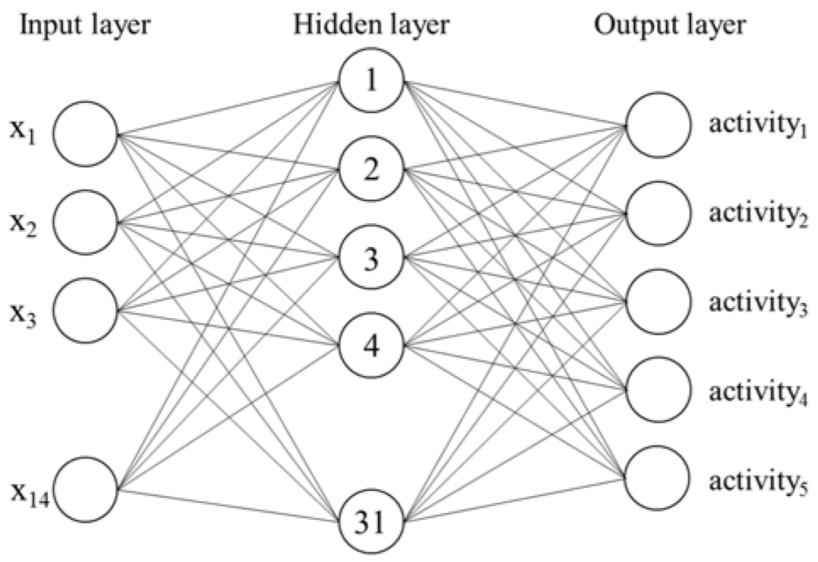

Figure 2.4: Structure of fully connected feed-forward ANN. The number of neurons in the input layer corresponds to the number of features (see 2.2.3). The number of neurons in the output layer corresponds to the number of different trip purposes that should be predicted. One can choose to add an arbitrary number of hidden layers with an arbitrary number of neurons inbetween, these are Hyperparameters that needs to be optimized. Image taken from [10].

As the input and the output layer of the network are fixed (see Figure 2.4), one still has a lot of hyperparameters to set, the following gives an overview of those, including options and their influence on the result:

Network layout As already discussed above, one can choose the number of hidden layers as well as the number of hidden neurons in them. A greater number increases the model's capacity to learn, but is also prone to overfitting, as this gives the model the capability to 'memorize' the dataset.

Activation function Each node or neuron in the network takes the inputs of all neurons, weighs them with their specific weights, sums them up and then its output is this sum passed through the activation function.

$$a_j^{[l]} = g^{[l]}\left(\sum_k w_{jk}^{[l]} a_k^{[l-1]} + b_j^{[l]}\right) = g^{[l]}(\mathbf{z}_j) \tag{2.10}$$

with $[l]$ denoting the l-th layer, k being the k-th neuron, a being the activation and w_{jk} weight for connecting the k-th neuron in the previous layer with j-th neuron in the current layer and b denoting the added bias. The activation function g, usually some kind of nonlinear function, can be exchanged and is the hyperparameter to optimize. Traditionally this was set to a *tanh* or a *sigmoid* function. Recent advances in ANN however, suggest that Rectified Linear Units (ReLUs) or Concatenated Rectified Linear Unitss (CReLUs) are a better choice, reaching a better overall performance.

As usual in multiclass classification tasks, the activation function of the last layer is a softmax function:

$$a_j^{[last\_layer]} = \sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{2.11}$$

with $\mathbf{z}$ being a vector consisting of all results that are usually fed into activation functions. This outputs one probability for every class, and all probabilities add up to one.

Optimization algorithm Learning in the neural network means changing the weights $w_{jk}^{[l]}$ (see (2.10)) in the network. This is done by an algorithm called backpropagation. In a nutshell, one passes *batchsize* number of samples trough the network (forward pass) and checks the outcome against the truth. The average error is then calculated and passed backwards through the network (backward pass). Weights are then changed based on their contribution to the error. Weights (connections) with bigger error will receive a greater change.

The basic algorithm for this procedure is called Gradient Descent (GD). In its original form, *batchsize* would be equal to the samplesize, meaning the network sees all samples before making a change to the weights. As this would take too long with big datasets, one now normally uses Stochastic Gradient Descent (SGD), where the batchsize is usually in the range of 1-10 samples. Strictly speaking, SGD requires the *batchsize* to be 1, using bigger batches is usually reffered to as Mini-batch GD. It has also been shown that, even though one does not go down the error surface in an ideal way, the randomness introduced by SGD helps to prevent getting stuck in local minima.

After calculating the gradient, SGD will then change the weight relative to a factor called *learning rate*. As one would want this to be bigger in the beginning to allow for quick learning and then becoming finer as training continues, people started to introduce decays depending on the *iteration* for example exponential.

Nowadays, algorithms performing an optimization of the learning rate have been proposed. Some of the most used ones are:

- AdaGrad
- AdaDelta
- Adam
- RMSProp

Disadvantages of the algorithms above include the need to save a bigger state space and a greater computational cost when calculating the gradient as well as the step size.

Learning rate As discussed above in *optimization algorithm*, the *learning rate* poses a problem with regular GD. If one has to set it, it is usually beneficial to use some kind of decay, e.g. exponential decay to ensure a faster convergence.

Batch size As mentioned in *optimization algorithm*, one usually takes mini batches of about 32 samples to make an update.

Number of epochs An iteration includes the forward pass of *batchsize* samples and the following backpropagation of the error. An epoch is done as soon as the network has seen every single sample once. If one increases the number of iterations, the network gets to see the data more often and has the ability to learn more. On the other hand, this can result in a loss of generalization if there is not regularization. A common method for finding the number of iterations is to split the training set once more, to obtain a set which can serve as determining base when to stop. This technique is called 'early-stopping'.

Dropout One of the biggest improvements in training neural networks has been the introduction of a technique called dropout in 2014 by Srivastava et al. [33]. It uses the very simple idea of setting the output $a_j^{[l]}$ of a percentage of neurons to zero during one iteration. This forces the network not to be dependent on certain neurons, distributing knowledge over all neurons. Dropout can be seen as regularization technique and is usually written in the form of a *keep probability*, which is the percentage of neurons to keep active.

In addition to the network layout, one needs to optimize, validate and implement the network. These topics are covered in the following paragraphs:

Optimization The current standard for finding the best set of hyperparameters is to use a grid search. As one can imagine, having eight unknowns to optimize, this requires a lot of time. Therefore, it is essential to use sensible search ranges. However, this work's aim is not to provide an optimized version of the network, but rather a proof of concept that a network can learn information given the features described in the next section.

Validation When using k-fold cross-validation for smaller datasets, this (normally) increases the immense computation by the factor of k. As the dataset is fairly large, however, one usually shuffles the data and does a stratified split into training set, validation set and testing set. Since no cross-validation is done, one needs to make sure to have big sample sizes on the testing as well as on the validation set. This work uses a split of 50%-25%-25% for training, validation, and testing respectively.

As for the performance measure used, one has to watch out for the class imbalance (see Table 2.1). Even though there are nine classes in total, a classifier that would always predict `Home` would already score close to 50% accuracy. Therefore, not just the accuracy (precision), but also a confusion matrix is reported.

Implementation The whole network was implemented in TensorFlow™, a graph-based computing framework originally developed by Google™. The idea behind this approach is that one designs a graph of interconnected operations and TensorFlow™, then does all the mathematics behind the scenes. The low-level implementation of the graph is optimized to the current computing architecture, offering options to also run on a Graphics Processing Unit (GPU) featuring a greater speed for the matrix multiplications needed in ANNs. This greatly reduces computation time.

Numerical features were normalized by their mean and standard deviation on the testing set. Categorical features were one-hot encoded using TensorFlow's™ 'Feature Columns'.

The whole training process was visualized using TensorBoard™, insights into the classification were achieved using the What-If Tool (WIT).

2.2.2 Preprocessing

As in any application of machine learning, the basis for good and correct results lies in a clean dataset. To accomplish this, many different options have been considered in the past. Data preprocessing in the field of trip purpose imputation still relies heavily on the knowledge of domain experts, as for most cases the datasources as well as the exact problem formulation differs. Preprocessing of spatial trajectories is often done only with rule-based methods [3,25,36], which due to their discrete nature usually don't generalize well.

Data selection

Trajectories consist of a variable number of datapoints per day, depending heavily on the activity of the mobile phone user. The more active a user is, the more datapoints are created during a day; however, not every event like a call or using the internet triggers an event.

As can be imagined, purpose imputation is next to impossible when there are not enough datapoints during a day. Therefore, trajectories that consist of less than a certain amount of observations are removed. Finding this threshold is a trade-off between removing possibly incorrect samples vs. keeping enough valuable training data. On average, a user in the dataset collected for this thesis had 658.41 events per day. In comparison, the average Austrian in the datastream coming from the phone provider has 421 observations per day.

Empirical investigation has shown that trajectories having below 250 observations do not contain enough information to be useful and are therefore rejected. For the dataset collected in this study, in total 12 trajectories have been removed because of this rule.

Speed filter

All location estimation from CDRs is subjected to noise. As with measurements in general, this can be split into randomly and systematically made errors. A systematic one has a constant bias, for instance a cell tower with an incorrect saved location. This would produce a wrong result whenever this tower is involved in the estimation. And if one averages all observations at this tower, the mean error would be the distance to the real location. Random measurement noise, however, is noise that occurs through stochastic processes while measuring. If one would average measurements taken on the same spot, the mean position would approach the real coordinates the more measurements there are.

So additional to whole trajectories being rejected, certain observations might also be incorrect. Rule-based approaches target different properties of a trajectory; for this thesis, only one rule using the maximum speed was applied. An algorithm was then developed to remove observations with a speed higher than this maximum. It was set to 230 km/h per hour, as the fastest possibility in Austria would be a Railjet (train) traveling at a maximum speed of 230 km/h. A complete description of the algorithm used can be found in Algorithm 1.

Algorithm 1 Speed based filter for one trajectory

```
1:  procedure HIGHSPEEDFILTER
2:      $previous\_index \leftarrow 0$
3:      for $i = 1$ to number of observations - 1 do
4:          $\Delta t \leftarrow timestamp(observations[i]) - timestamp(observations[previous\_index])$
5:          $\Delta x \leftarrow distance(location(observations[i]), location(observation[previous\_index]))$
6:          $speed \leftarrow \Delta x / \Delta t$
7:          if $speeed < 230 km/h$ then
8:              Keep observation at position $i$
9:              $previous\_index \leftarrow i$
10:         $i \leftarrow i + 1$
```

Table 2.4: Percentage of rejected sample based on the cutoff speed

	Cutoff speed (km/h)	Percentage of rejected observations
0	300	5.23
1	250	6.58
2	230	7.36
3	200	8.80
4	150	12.65
5	100	19.86

This threshold reduced the number of observations on the dataset collected by this thesis by 7.36%. A comparison of how many observations would have been rejected depending on the speed threshold can be found in Table 2.4. As one can see, thresholds above 200km/h lead to a decrease of less than 10% of the data. If one lowers the threshold further down, however, a lot of datapoints are rejected and one would not just remove noise but possibly correct measurements.

Kalman filter

The process of removing random noise from temporal measurement sequences has been a topic not only limited to spatial trajectories. Although, at the moment in most areas where research is done with mobile phone trajectories they are mostly not processed [1, 17, 20, 37] or only with rule-based methods as described in the current above. The standard for GPS preprocessing has long been Kalman filtering or its altered versions such as the Extended Kalman Filter (EKF) or the Unscented Kalman Filter (UKF) (see 2.2.1 for background on the Kalman filter).

Even though previous work on preprocessing of mobile phone trajectories done by Horn et al. [14] suggested that rule-based methods outperform Kalman filtering, this work suggest a novel approach, using an EKF with a variable measurement covariance $\mathbf{R}$ based on the current location of the measurement.

In this concrete case of using mobile phone data, the internal state vector, the measurement vector as well as observation model ($\mathbf{H}$) look like this:

$$\hat{\mathbf{x}} = \begin{bmatrix} x \\ y \\ v \\ \phi \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.12}$$

Here x and y are the longitude and latitude respectively, v is the overall speed and ϕ is the direction of the speed vector.

Usually a Kalman filter works with a constant and comparatively small step size Δt. This is true for GPS data as it is usually sampled with approximately 1 Hz. Unlike GPS data, mobile phone data is fairly sparse, depending on the phone usage. Irregular time deltas are the norm, with value ranging from a few seconds up to half an hour or even an hour in the worst cases.

This leads to an internal model that should incorporate the step size as a factor. The model chosen here is a simple linear model, assuming that the subject moves into direction ϕ with constant velocity v for a given Δt:

$$f(\hat{\mathbf{x}}) = \hat{\mathbf{x}} + \Delta t \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ y \\ v \\ \phi \end{bmatrix} + \Delta t \begin{bmatrix} v\cos\phi \\ v\sin\phi \\ 0 \\ 0 \end{bmatrix} \tag{2.13}$$

Looking at equation 2.2 one can then derive **F**:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta t \cos\phi & -v\Delta t \sin\phi \\ 0 & 1 & \Delta t \sin\phi & v\Delta t \cos\phi \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

In addition to having many observations, the measurement noise in GPS data can, as a first approximation, be seen as a normal distribution, with an accuracy that is, depending on the location is somewhere in a range of few tens of meters. On the other hand, location estimates based on mobile phone data are usually quite noisy, based on how many cell phone towers are situated in close proximity. If there are fewer towers, and/or they are further away, the accuracy drops considerably.

As a result, when people usually take one measurement covariance **R** when using GPS data, with mobile phone data one has to consider **R** as a function of the current location. This work uses a Kernel density estimation (KDE) to estimate the variance on a certain location. The covariance is then assumed to be diagonal matrix, having this variance as diagonal entries. The base for the KDE are the locations of all cell towers from the data providing carrier in Austria. A density is then estimated using an exponential kernel with a bandwidth of 400 (using the `scikit-learn` [27] KDE implementation). This gives a density estimate for every point in Austria. The estimate is then inverted and scaled with an empirical factor to give an estimate for the variance.

This leaves only **Q** as an unknown. The first idea was to run grid search to find the optimal parameters. As an error measure, the mean square error of the trajectories was taken. This lead to unsatisfying results, however, as the mean square error does not seem to be the best measure in this case. As a result, optimization was done by hand, using a custom-built tool to visualize the internal filter state as well as the covariance matrices. The final state of **Q** was set to:

$$\mathbf{Q} = \begin{bmatrix} 10(\Delta t)^2 & 0 & 0 & 0 \\ 0 & 10(\Delta t)^2 & 0 & 0 \\ 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0.001\Delta t \end{bmatrix} \tag{2.15}$$

The filter was initialized using the first measurement as x and y, v and ϕ have been set to zero. No smoothing (e.g. Rauch-Tung-Striebel (RTS) algorithm [28]) was used, as this would increase computation time to an unfeasible amount.

Segmentation

If one would had an ideal trajectory, segmentation could easily be done by just looking at the current speed. As can be seen in Figure 2.5, this cannot be done with raw mobile phone trajectories. One could now smooth this curve, but as there are many possible different modes of transportation used, a simple filter would not be able to keep all the information. This is especially relevant with regard to a slow walk in the city, compared to a commuting train ride home.

The idea now is to use the velocity v from the internal filter state $\hat{\mathbf{x}}$ as a velocity profile. Due to the nature of the Kalman filter implemented, it only allows changes to the velocity trough the process noise (**Q**) and is therefore limited in its changing rate. As the change is not only dependent on the value of **Q**, but also on the Kalman gain **K**, the velocity will be filtered based on the internal state. This results in less smoothing for small velocities and more for higher velocities.

The segmentation algorithm will then split the trajectory based on this velocity. As a first threshold, all observations with a velocity less than v_{th} = 8km/h are marked as stationary, the

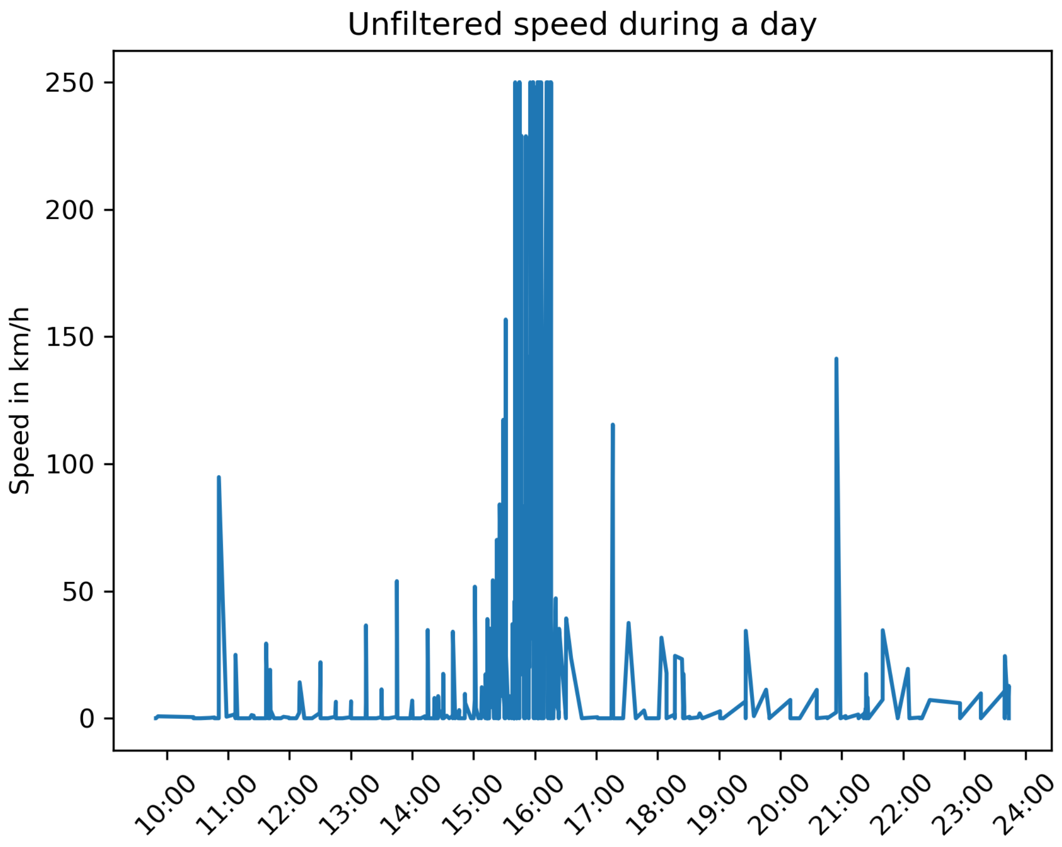

Figure 2.5: Speed profile of the raw trajectory. For better visualization the maximum speed was limited to 250 km/h, peaks would go up to around 800 km/h. 21.03.2018, subject 2

rest is marked as moving. This first threshold was set arbitrarily and does not matter as much as long as it is high enough to allow for a lot of stationary segments. These will then be merged or converted to moving segments in the following steps. As the velocity of the trajectory is still not perfect, a rule-based approach, similar to the ones used in [9, 25], was used to remove unfitting segments.

All in all, two rules were applied, the first one defining that a stationary segment must at least have a length of $t_{min\_stationary}$. The second rule states that all stationary segments that are closer than a certain distance $\Delta x_{min\_stationary}$ should be merged together, removing the moving segment in between them.

As also discussed in previous work [1, 9, 25, 37], a suitable time for $\Delta t_{min\_stationary}$ has to be chosen. In particular it should be long enough so as not to detect short stops at red lights or when someone switches the mode of transport, from example car to a train. Of course, a simple threshold cannot achieve a perfect separation; however, with a sufficiently long time $\Delta t_{min\_stationary}$, this can at least be improved.

To find the ideal values for $t_{min\_stationary}$ and $\Delta x_{min\_stationary}$ a grid search was performed. Search ranges were 5 to 20 minutes in steps of one minute for $t_{min\_stationary}$ and 300 to 1500 meters in 100 meter steps for $\Delta x_{min\_stationary}$. As a reference for segmentation, mobile phone trajectories from the collected dataset were annotated based on the trip records that were filled out in the Modalyzer app. To score the performance of the algorithm, each trajectory was split into artificial observations with a Δt of five minutes. A state, either stationary or moving, was then assigned to each of these datapoints. As the average person spends more time being stationary rather than moving, one has to be careful when choosing a measure for the accuracy.

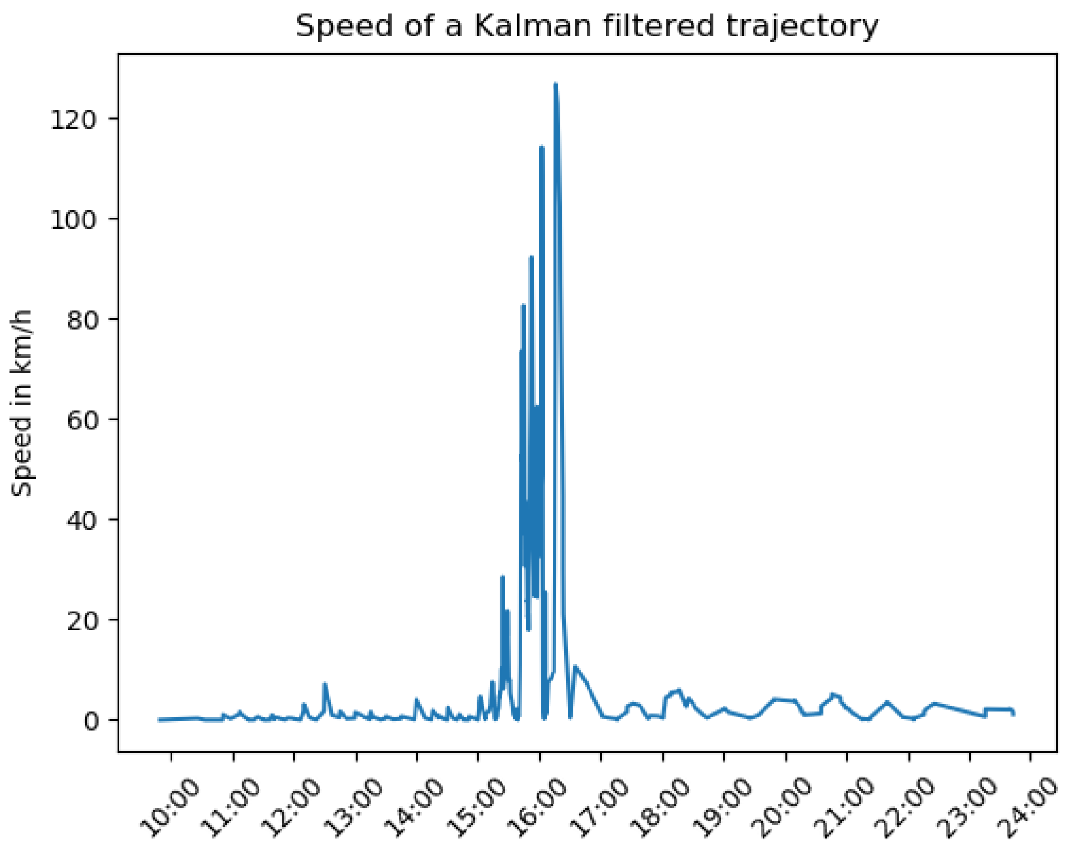

Figure 2.6: Speed profile created from the internal state v from $\hat{\mathbf{x}}$ of the applied Kalman filter (2.2.1). 21.03.2018, subject 2

To correct this imbalance, a balanced accuracy measure was used. In a binary context such as this, it is the mean of sensitivity (true positive rate) and specificity (true negative rate), or the area under the ROC curve. For the optimization a 5-fold cross-validation was used.

2.2.3 Trip purpose imputation

Feature extraction

As with all classical machine learning algorithms, the raw data, in this case the segments, have to be converted to features for a machine to be able to process the information. The creation of those features is crucial for the ability of the algorithm to learn and is more of a creative process. It is also the most critical step, as with the wrong features no patterns can be discovered.

Table 2.5: Comparison of features used in previous works. Translated from [19].

Year	Author	Features	Reference
2001	Wolf et al.	Duration, Start time	[38]
2003	Axhausen et al.	Duration, Start time, Activity, Visit fequency, Day of week	[39]
2005	Griffin & Huang	Duration, Start time, Cluster specific parameters	[12]
2007	McGowen & McNally	Duration, Start time, Repetition duration, Repetition location, Change in household	[23]
2009	Huang et al.	Start time, Activity order, Day of week	[16]
2010	Deng & Ji	Start time, Trip duration, Trip distance, Trip speed, Day of week, Transportation mode	[6]
2010	Chen et al.	Start time, Activity Order	[4]
2013	Shen & Stopher	Start time, duration, activity order	[31]
2013	Montini et al.	Start time, Duration, Day of week, Mode of transport, Clusterspecific parameters	[26]
2013	Lu et al.	Start and End time of trips (before and after activity), Duration (of current, previous, following segment), Activity order (first, last or in-between), Day of week, Mode of transport	[22]
2015	Feng & Timmermans	Start time, Duration, Trip duration, Mode of transport	[8]
2015	Alexander et al.	Start time, Visiting frequency, Day of week	[1]
2015	Widhalm et al.	Start time, Duration, Activity order, Visiting frequency	[37]
2016	Xiao et al.	Start time, Duration, Day of week, Mode of transport	[40]
2016	Han & Sohn	Start time, Duration	[13]
2016	Gong et al.	Start time, Duration, End time, Trip duration, Day of week	[10]
2017	Usyukov	Start time, Duration	[36]
2017	Janzen et al.	Trip duration, Trip distance, Month, Weekend part	[17]

With trip purpose imputation being a very established topic already, certain types of features have been proposed in previous studies. An overview of those can be found in Table 2.5.

The features used in this work were mainly dictated by their availability in the dataset used for training the classifier (see 2.2.3). It is, however, crucial to consider that the classifier was not trained with the data recorded for this study, as it would have been biased by the unrepresentative nature of it. More information about the training data ('Österreich unterwegs' (OEU) dataset) for the classifier can be found in the next section. Where other works were using spatial features, such as landuse or distances to Point of Interests (POIs), this was not possible here as the training data from OEU did only include spatial information on a municipality level. The data does however include sociodemographic information such as an age range and the home municipality.

Some of this demographic information is also available in the data coming from the net provider, as customers have an opt-in option to share their data when signing their contract. As not everybody opts-in, the classifier has to be prepared in a way that some of the data is missing.

In general, if the dataset is big enough, all possible features can be added. The idea is to let the algorithm itself figure out which are useful and which are not. In this work the following features were extracted:

Start time The start time of the segment in hours relative to midnight. The distribution of the start time split by the trip purpose can be seen in Figure 2.7 and Figure 2.8. Numerical.

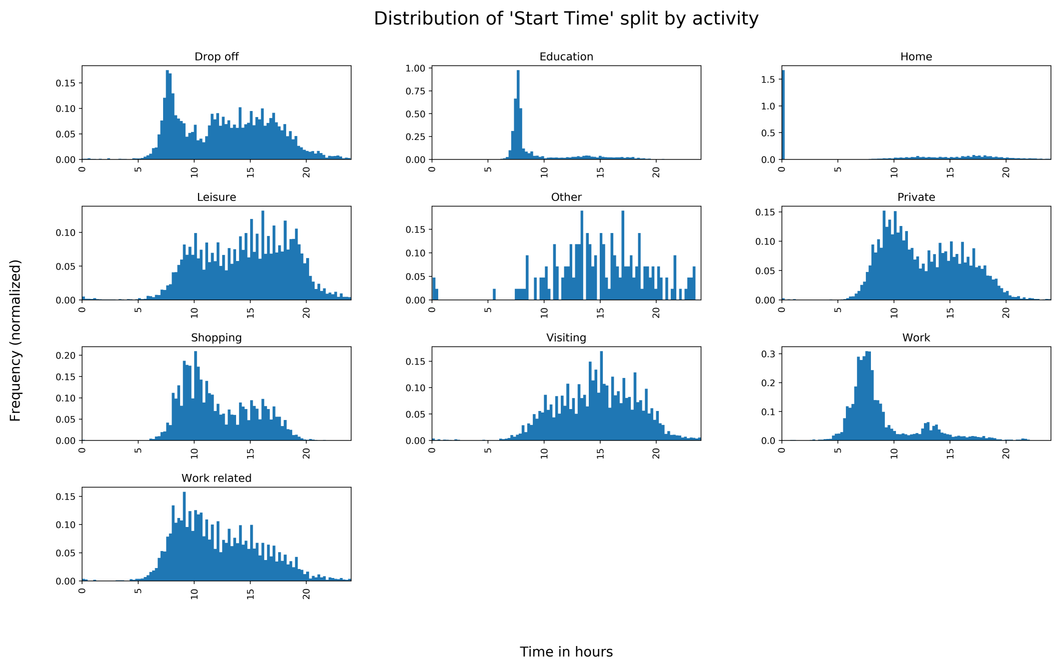

Figure 2.7: Distribution of activity start times relative to midnight (Local time). As most `Home` segments start at midnight, a second plot (Figure 2.8) is provided to show its distribution during the day.

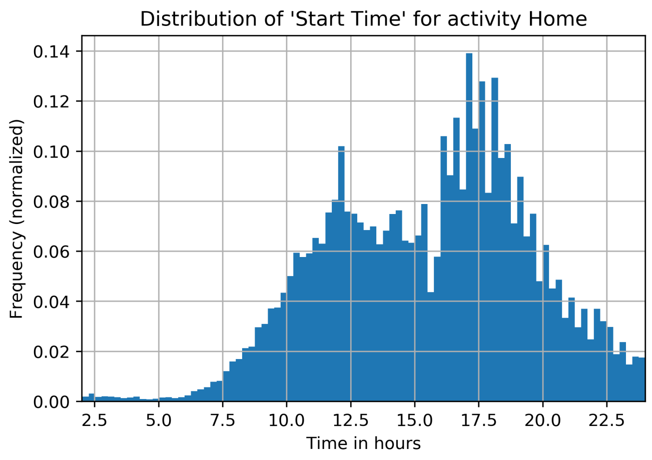

Figure 2.8: Distribution of 'Start Time' for activity `Home`*. Only segments starting after 2 am (Local time), to show the distribution without the peak around midnight (see Figure 2.7 for the complete distribution).*

Duration The duration of a segment in hours. An overview of the distribution split by activities is plotted in Figure 2.9. Numerical.

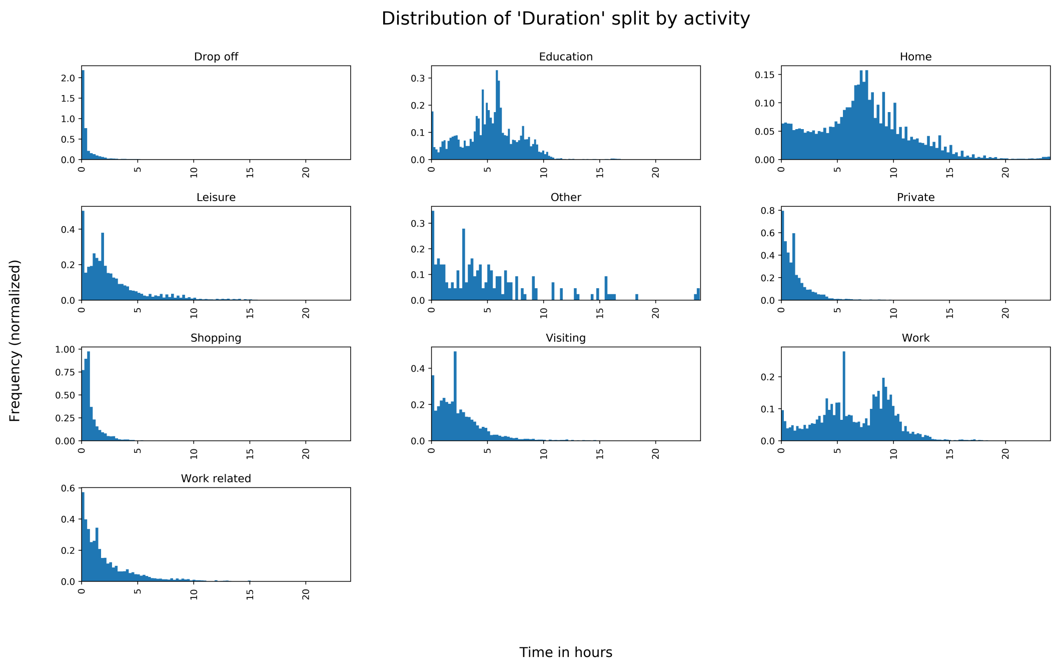

Figure 2.9: Distribution of the duration of different activities, extracted from the OEU dataset.

Activity Order The order of the activity during the day. Similar to the approach in [26] acceptable values were 'first', 'middle' or 'last' corresponding to the position of the segment in a trajectory. The distribution of activity order can be seen in Figure 2.10. Categorical.

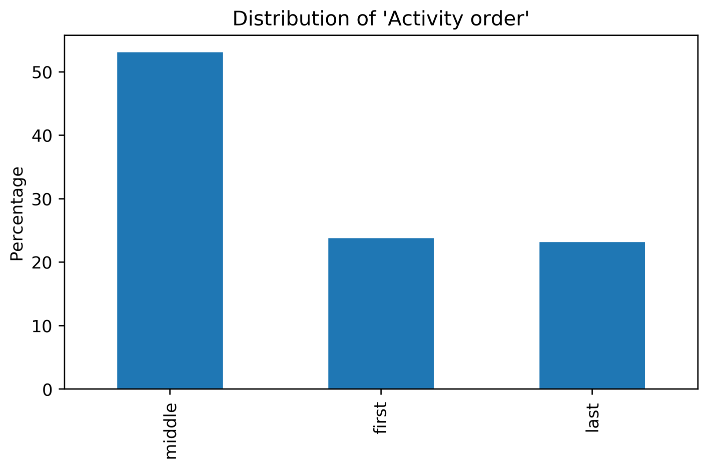

Figure 2.10: Distribution of activity order in the OEU dataset.

Day type Measure for the type of day. Categories are 'Working day', 'Saturday', 'Sunday', where holidays are included in 'Sunday'. The distribution in test dataset can be seen in Figure 2.11. Categorical.

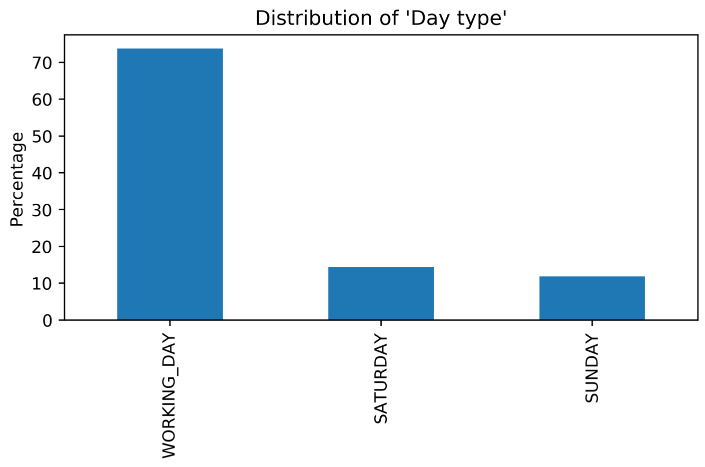

Figure 2.11: Distribution of day types in the OEU dataset.

Location Type Each district and therefore each municipality can be assigned one of the following:

- 'Vienna'
- 'Major city without Vienna'
- 'Central district'
- 'Peripheral district'

An overview of which district belongs to which location type can be found in Figure 2.12. Categorical.

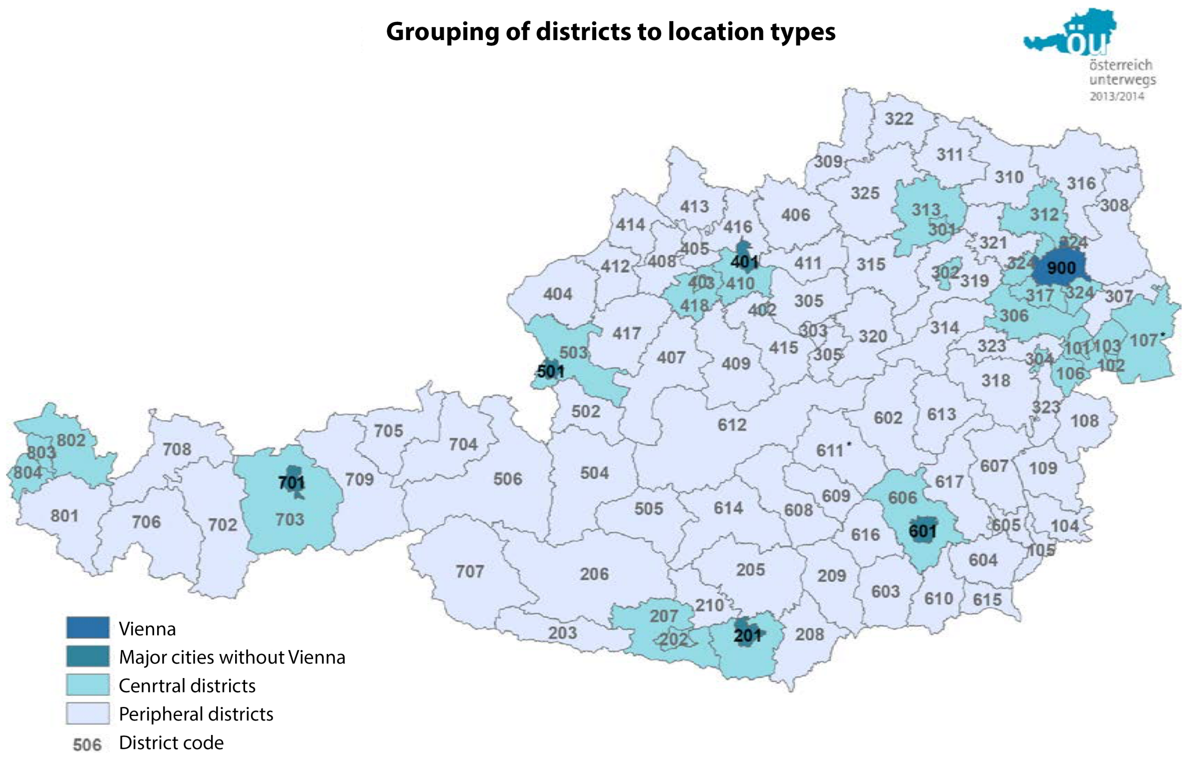
Grouping of districts to location types
österreich unterwegs 2013/2014
Vienna
Major cities without Vienna
Cenrtral districts
Peripheral districts
506 District code

Figure 2.12: Overview of the location type, translated from [34]. Location type is based on how remote the district is (ÖROK 2005). Districts with an asterisk are different ÖROK 2005, based on suggestions from the states.

Age Split into the following categories:

- 0-17 years
- 18-29 years
- 30-39 years

  ⋮

- 70-79 years
- older than 80
- NOT_SET

This is also the same format as provided by the cell phone company; however, only around 30% of all trajectories are enriched with sociodemographic information. The age distribution of the test dataset can be seen in Figure 2.13. Categorical.

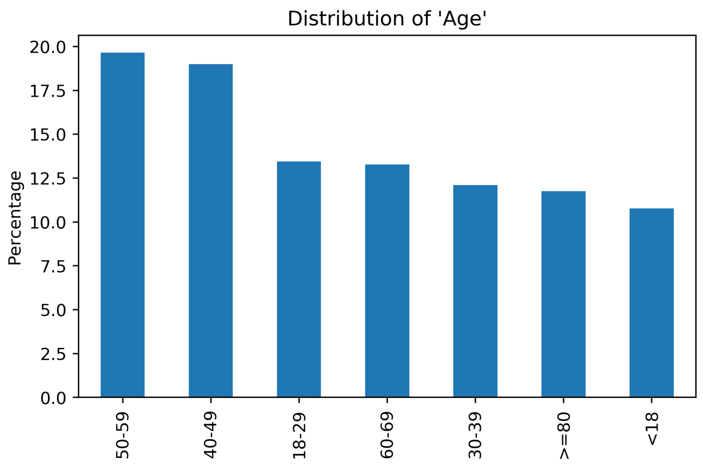

Figure 2.13: Distribution of age groups in the training dataset.

Gender Possible values are 'Male', 'Female' or 'NOT_SET'. This is also only set if is sociodemographic information is available for a trajectory. The age distribution of the test dataset can be seen in Figure 2.14. Categorical.

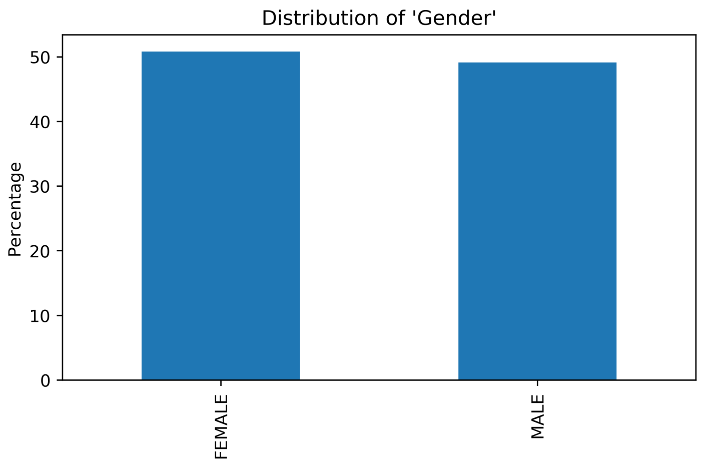

Figure 2.14: Distribution of gender in the OEU dataset.

Annotation

As the recorded dataset for this thesis (2.1) is not representative for the Austrian population, an algorithm trained on this data would not be able to generalize well and be biased towards the daily routines in the collected dataset.

For this purpose, data collected for the survey OEU [35] was used. The study was conducted on behalf of the Austrian government in 2013/2014 and incorporated 38,220 participants. The focus, however, was not on the trip purpose, but on the ways between activities. In total 196,604 ways, were recorded and, when looking at the stationary segments between those ways, this results in 307,890 annotated segments.

The complete cleaned dataset used for all the analysis is available to download online. Each of the recorded ways includes detailed information such as the time, municipality, purpose, season the trip start as well as the trip end municipality. Additionally, each trip is associated with an anonymized person, offering sociodemographic information that can be used for analysis.

Table 2.6: Number of stationary segments grouped by the trip purpose for the OEU data. For an elaborate description of the different activities please consult Table 2.1

	Number of Segments	Percent
Trip purpose		
Drop off	9104	3.66
Education	7591	3.05
Home	130981	52.65
Leisure	25807	10.37
Other	173	0.07
Private	16031	6.44
Shopping	18721	7.53
Visiting	12086	4.86
Work	21987	8.84
Work related	6276	2.52

As can be seen when comparing Table 2.1 to Table 2.6, the smaller dataset collected for this thesis shows a different distribution. This is due to the fact that the sample in the collected dataset is smaller and was recorded mostly by men between the age of 25 and 40 in the area of Graz and is therefore not representative of the Austrian population. When one extracts the same target group of young men living near Graz from the OEU dataset, segment wise feature statistics from the dataset collected by this study match the extracted data.

As it usually not feasible to run a grid search for tuning all hyperparameters in the ANN, optimizing is performed through trial and error using educated ranges. Some work has already been done on which parameters to optimize first [11], but as they are all interconnected there is no gold standard. In a first step, the optimizer was set to Adam to allow for faster convergence and be less dependent on the learning rate. Also, the batch size was fixed to 32 as it mostly affects convergence performance and usually people use a power of two to better fit computer hardware. The first test network trained was intentionally too big with 1000 neurons in one hidden layer. After learning, and of course overfitting, was observed, the network was regularized by decreasing the keep probability for dropout until overfitting could not be seen any longer. The network was then decreased in size without loosing too much accuracy on the test set to speed up training and evaluation performance. The final hyperparameters of the model then were:

- *Network layout*: One hidden layer with 500 hidden neurons each
- *Activation function*: ReLU

- *Optimization algorithm*: Adam
- *Learning rate*: 0.001. This was kept at the default value of the TensorFlow™ `AdamOptimizer` as it is not decisive because the algorithm performs an automatic decay.
- *Batch size*: Was set to 32. This is a compromise as a batchsize of one would be desirable, however, the added computational effort and therefore the increased computation time allow this for a dataset with this many samples and limited computing hardware.
- *Number of epochs*: Was set to 50. No overfitting was observed, but the loss did not increase anymore afterwards.
- *Dropout*: The *keep probability* was set to 0.5, dropping 50% of all neurons at every batch.

3 Results

The following sections show results of the different processing blocs seen in Figure 2.3.

3.1 Processing pipeline

3.1.1 Speed filter

Figure 3.1 shows the outcome of a trajectory filtered with the developed speed filter (Algorithm 1). As discussed in section 2.2.2, a cutoff of 200 km/h was chosen. The corresponding speed profile of the same trajectory can be seen in Figure 3.2.

3.1.2 Kalman filter

The outcome of the reference trajectory being filtered with the Kalman filter developed in 2.2.1 is shown in Figure 3.3. Having parts in the countryside as well as parts in the city, this trajectory serves as a good example. However, as one trajectory cannot be representative for the whole testset, the negative example (Figure 2.2) was also filtered with the Kalman filter (Figure 3.4). All trajectories shown in plots in this section have been preprocessed using the speed filter.

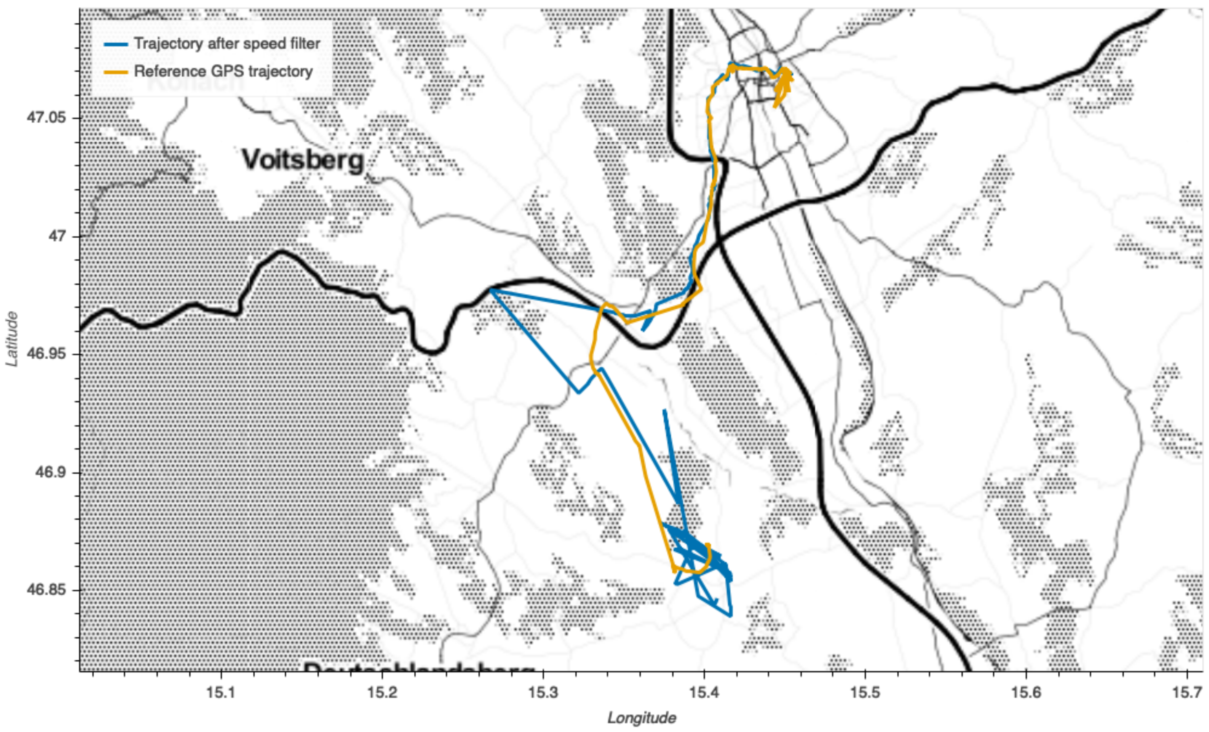

Figure 3.1: Trajectory filtered with the speed filter (Algorithm 1) with a cutoff v_{th} of 200 km/h. 21.03.2018, subject 2

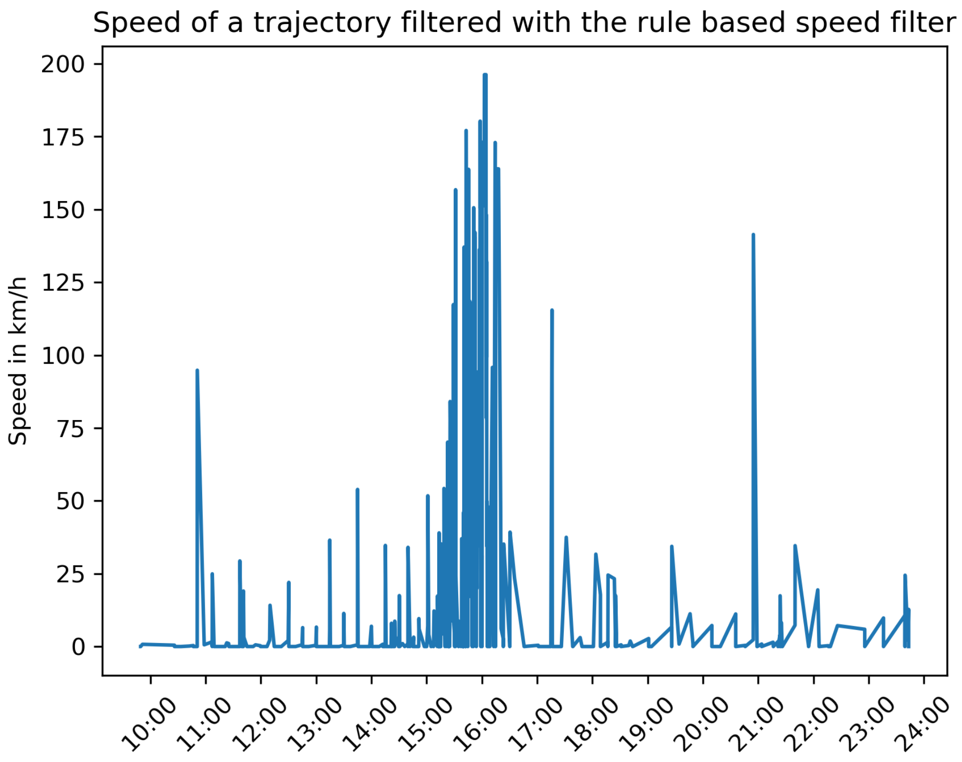

Figure 3.2: Speed profile after the trajectory was filtered with the speed filter (Algorithm 1, $v_{th} = 230km/h$). As expected, there are no velocities higher than 230 km/h anymore. 21.03.2018, subject 2

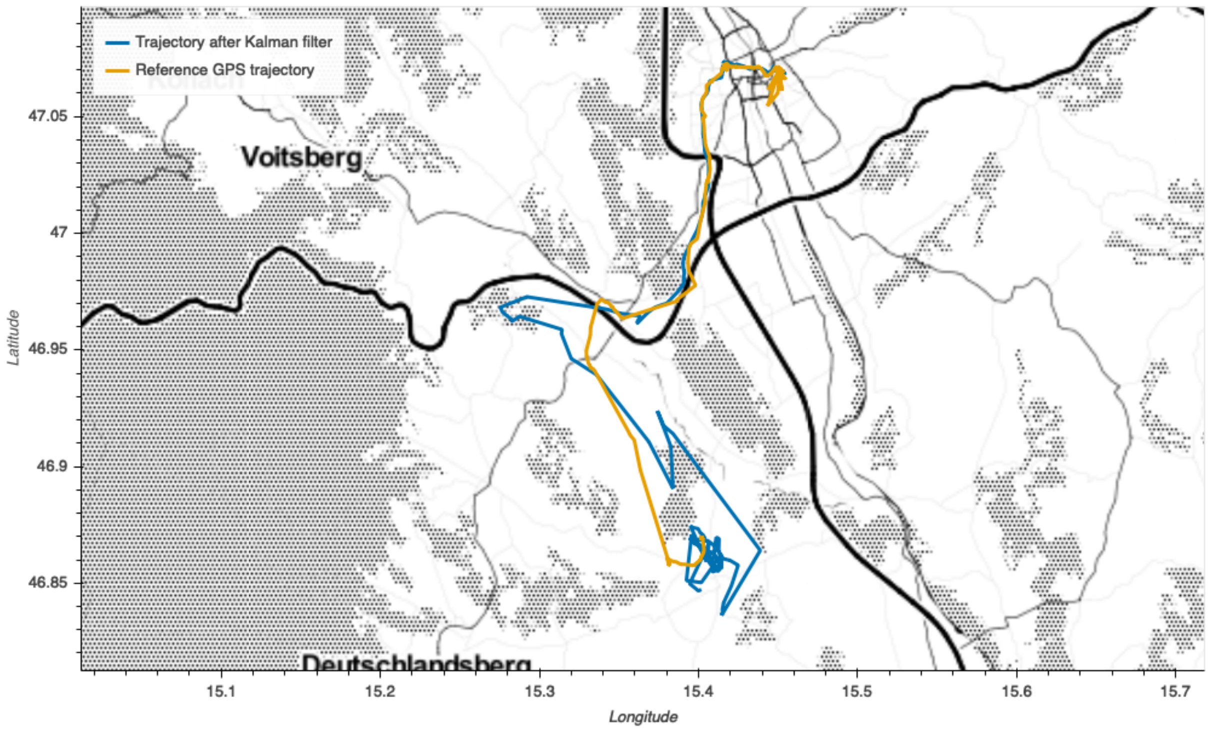

Figure 3.3: Raw trajectory from filtered with speed filter (Algorithm 1, $v_{th} = 230km/h$). 21.03.2018, subject 2

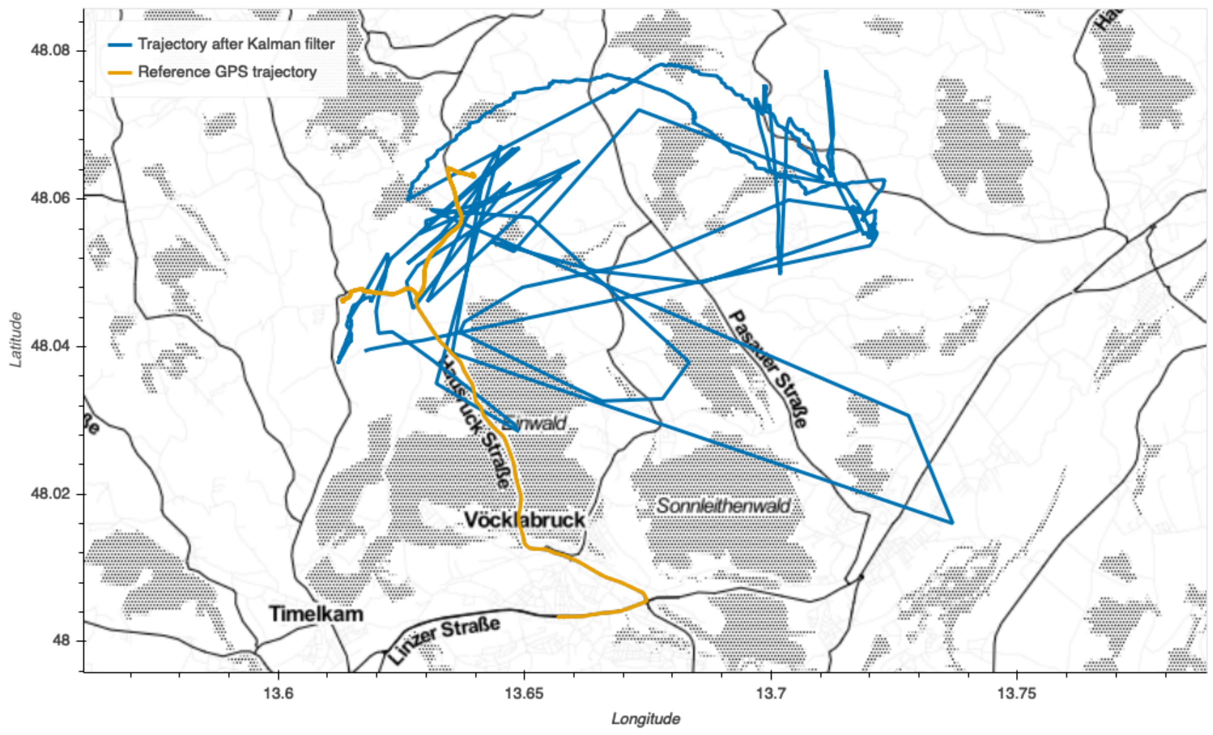

Figure 3.4: Raw trajectory from the countryside filtered with speed filter (Algorithm 1, $v_{th} = 230km/h$) and Kalman filter. 01.11.2018, subject 15

3.1.3 Segmentation

Using the metric described in 2.2.2, the optimized values found were:

- $t_{min\_stationary} = 15$ min
- $\Delta x_{min\_stationary} = 700m$

With these values, the best balanced accuracy was 77%. The general accuracy (number of correct observations divided by all observations) was also calculated, as it gives a good measure of how well a whole day can be split. With the setting above, an accuracy of 91% was achieved, meaning that all 207 trajectories considered, on average 91% of the day, the segmentation was correct.

One can see an example of a segmented trajectory in Figure 3.6. As a comparison, the same trajectory was segmented based on the trip journal (Table 2.2); the result can be seen in (Figure 3.6). In the example a balanced accuracy of 99.68% as well as a regular accuracy of 99.40% was reached.

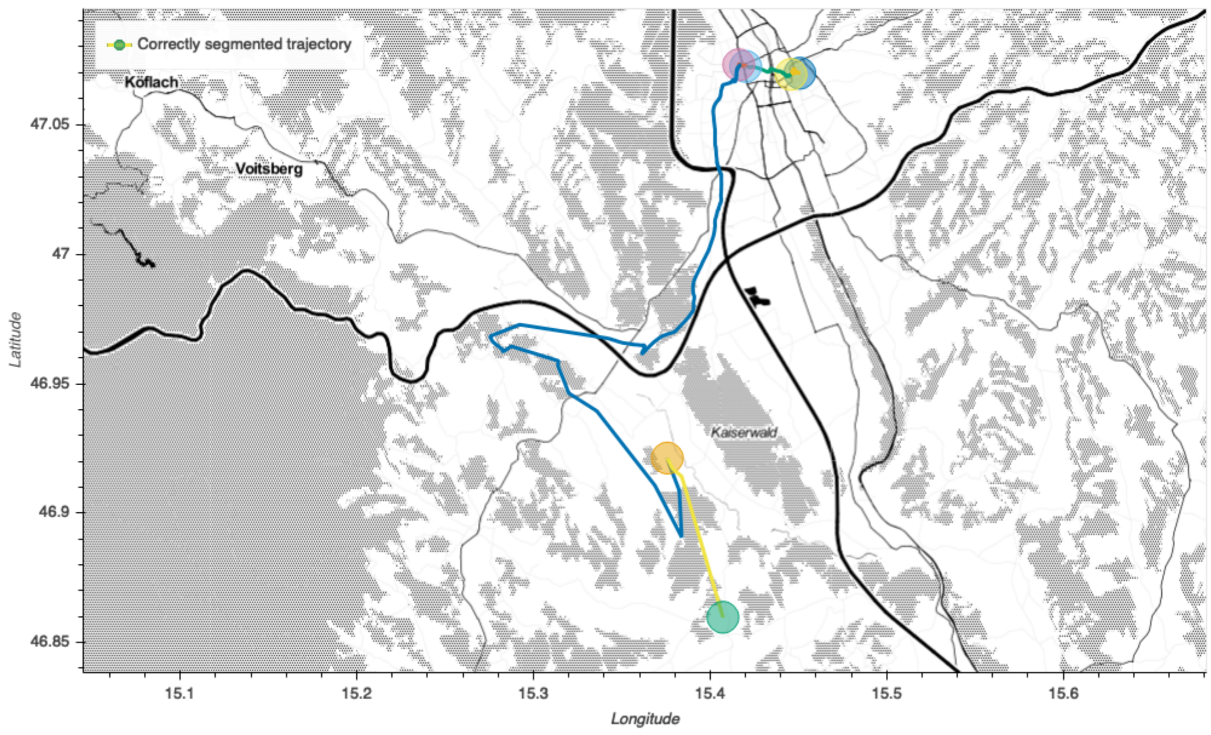

Figure 3.5: Trajectory segmented using the collected trip journal (Ground truth). Stationary segments were replaced with a circle, with the middle being in the centroid of all observations. 21.03.2018, subject 2

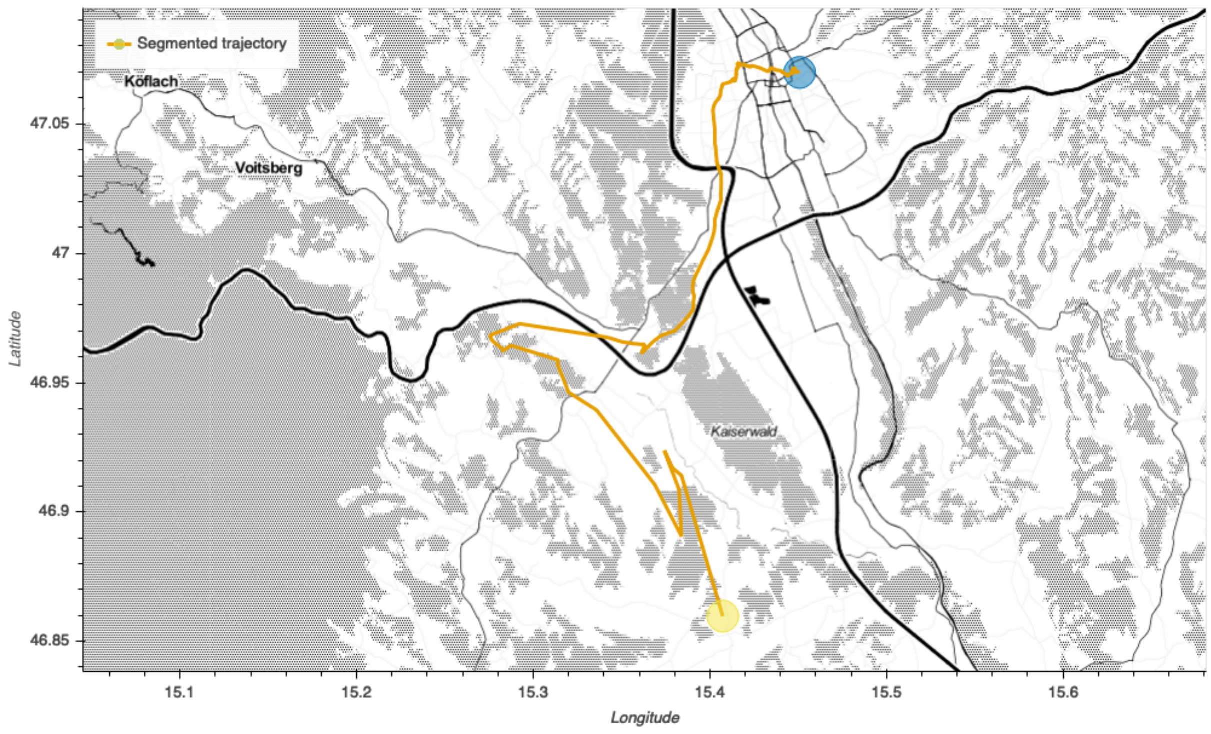

Figure 3.6: Trajectory segmented with the method presented in 2.2.2. Stationary segments were replaced with a circle, with the middle being in the centroid of all observations. 21.03.2018, subject 2

3.1.4 Trip purpose imputation

The overall classification accuracy on the OEU test dataset was 67.84%. However as this dataset is highly imbalanced, this is not a good measure of the classifiers' performance. An overview of the class imbalance is given in Table 2.6. For a better understanding of the classification, a confusion matrix of the results is provided in Figure 3.7.

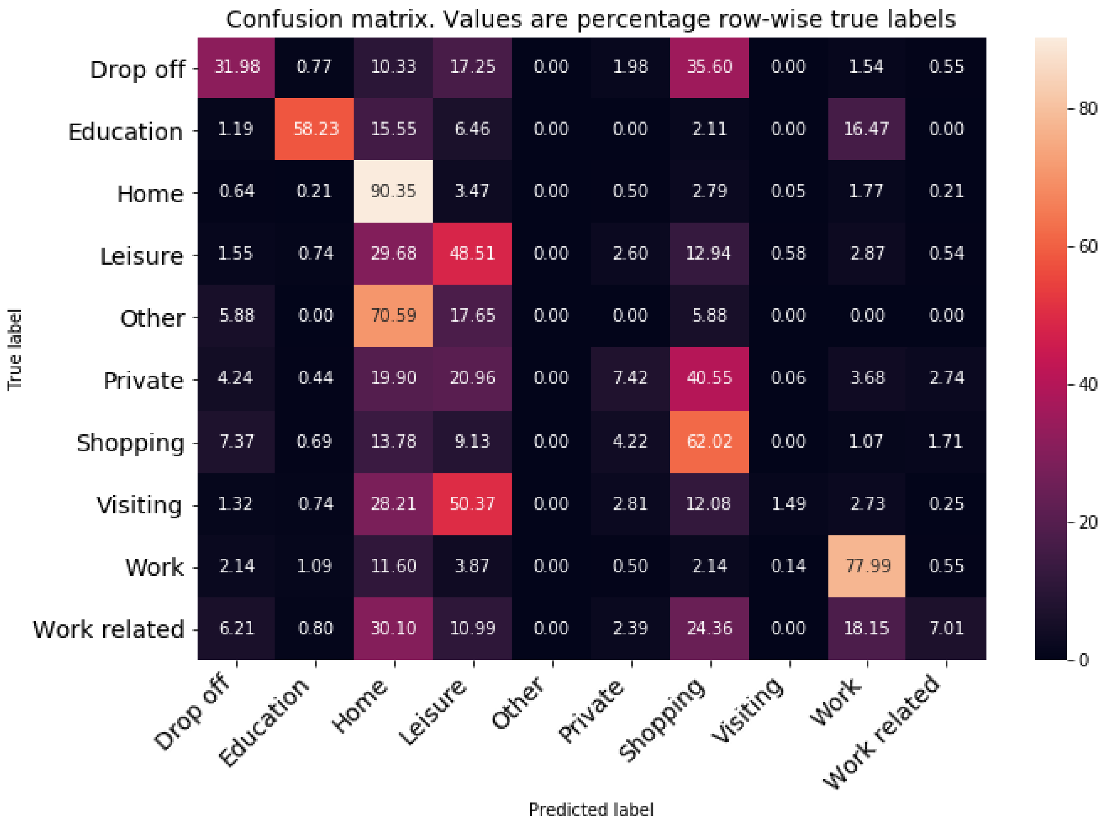

Figure 3.7: Confusion matrix after classifying the test set. Cell values are row-wise percentages of all true labels. For a class balance overview see Table 2.6

3.2 Application Example: One day of data

To test the complete system, the whole pipeline was applied to data created by all customers of the phone provider for a day. In total, one day consists of about 3,000,000 trajectories. The phone company also has an optional opt-in upon contract signing, allowing them to add basic demographics to each user; as a result so some trajectories have additional information such as age-group or gender attached. The size of this dataset proves a further challenge, as one day consists of about 70 GB, creating the need for optimized algorithms to process the data in a feasible time.

The following plots show simple metrics of all activities created by applying the here developed methods. Figure 3.8 shows the number of activities per trajectory, Figure 3.9 gives an overview of how the detected activities are distributed.

The other three Figures (3.10, 3.11, 3.12) are plots of one activity class, shown for Graz, Styria. To create these plots, the location of all stationary segments of the given activity were plotted on a map. The brightness of each pixel correlates with the number of underlying segments of the given activity. Major employers and shopping centers in Graz were identified.

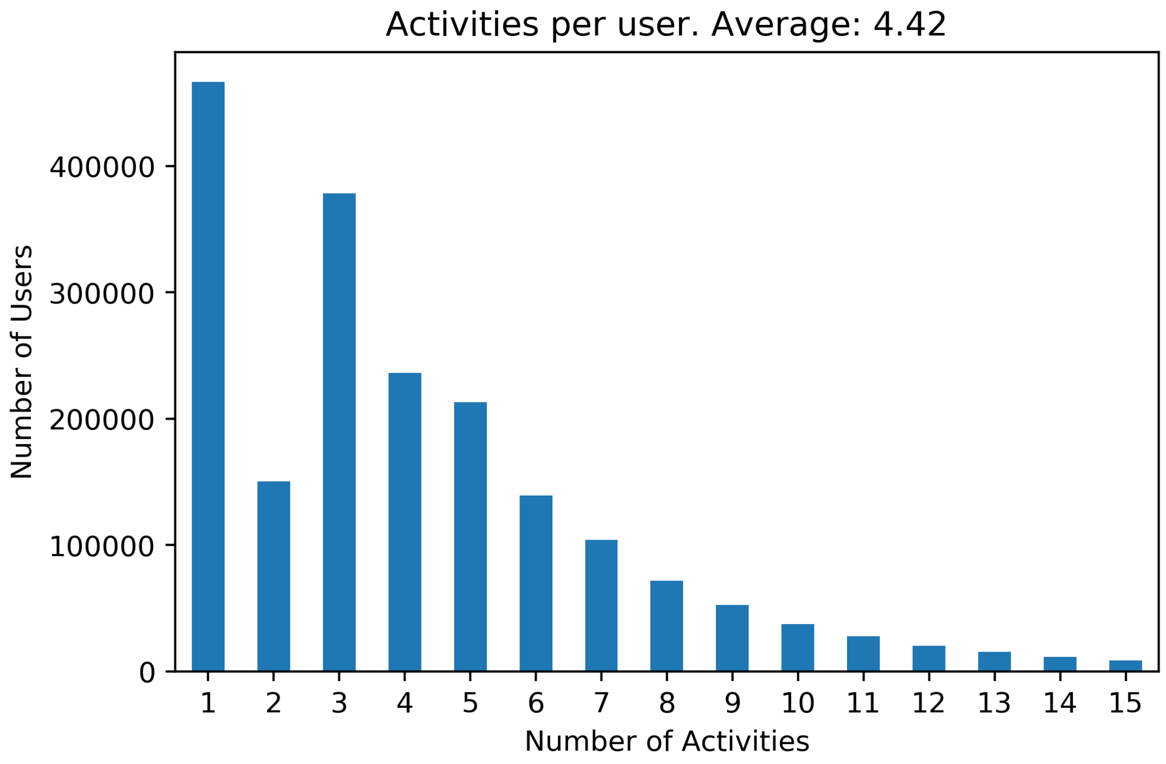

Figure 3.8: Activity per user, cutoff at 10 activities per user to better fit the data

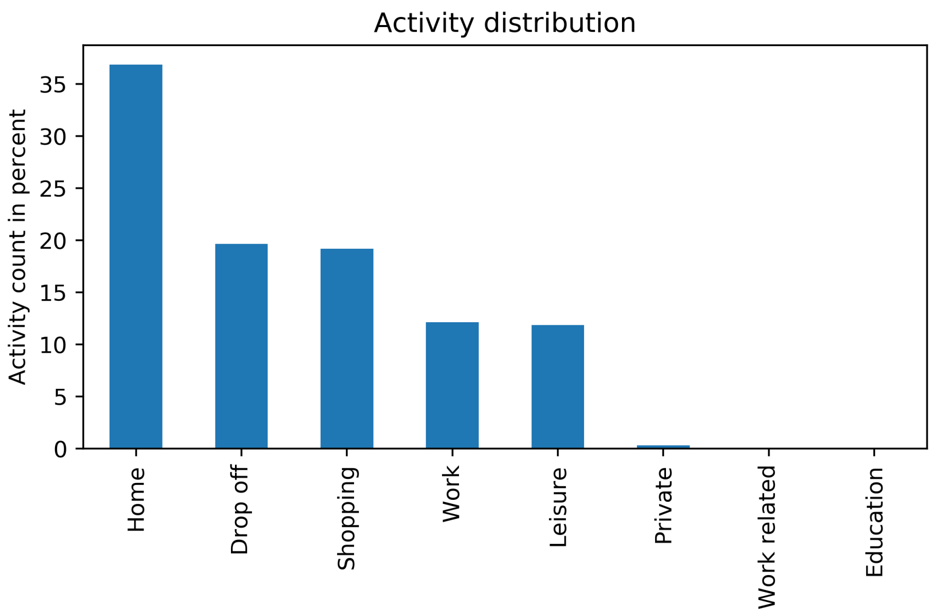

Figure 3.9: Distribution of all detected activities

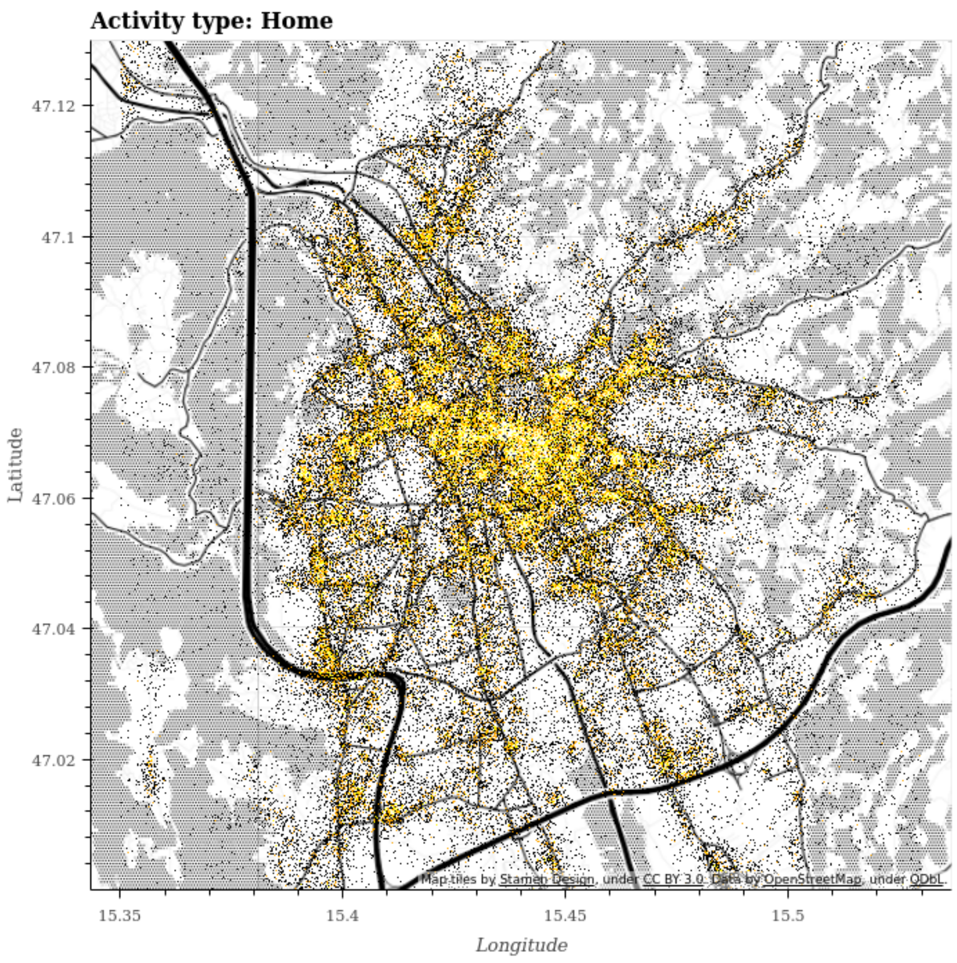

Figure 3.10: Distribution of all detected stationary `Home` segments in the area of Graz

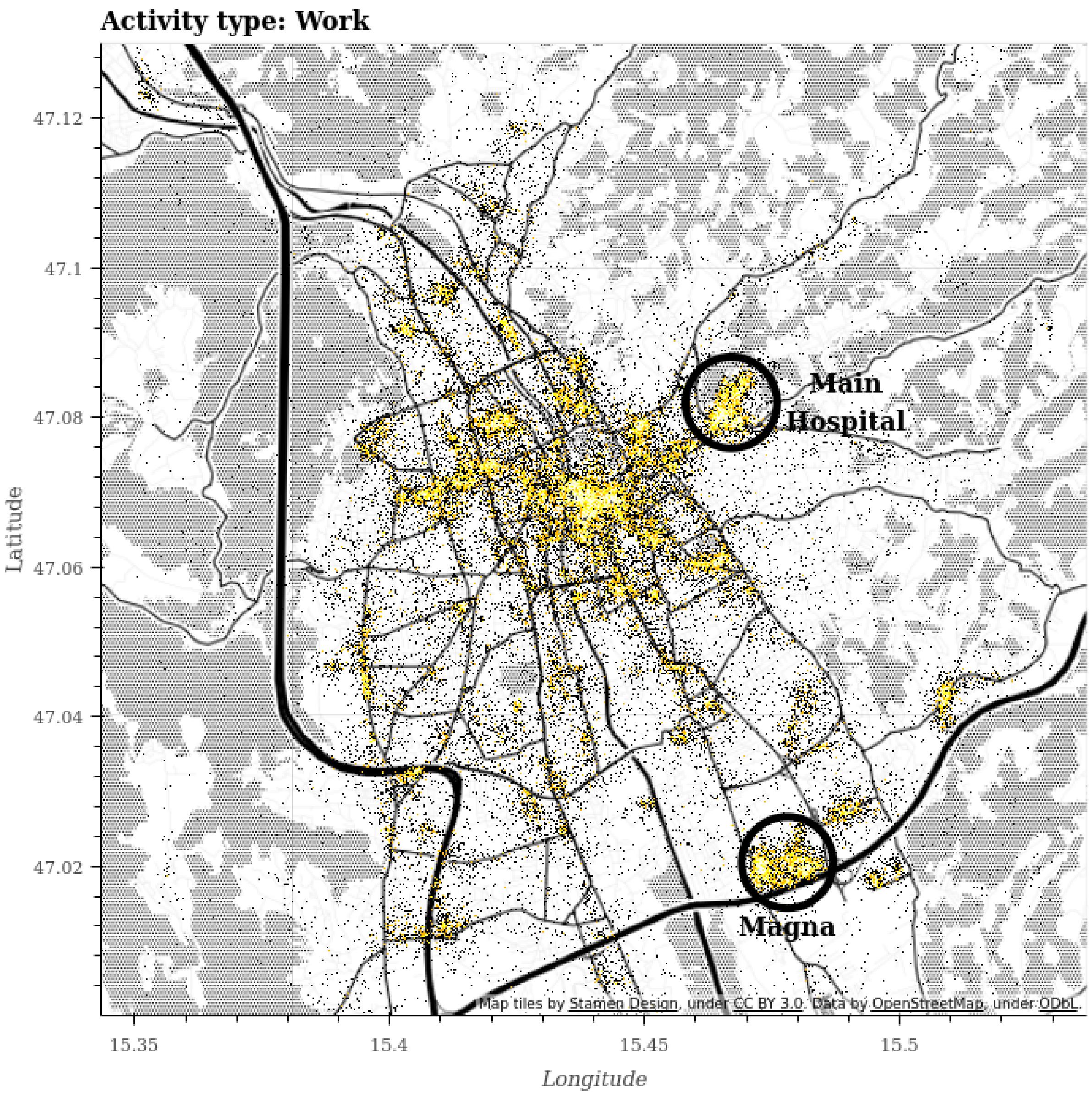

Figure 3.11: Distribution of all detected stationary `Work` *segments in the area of Graz. Magna is an automotive company with 12,000 employees worldwide*

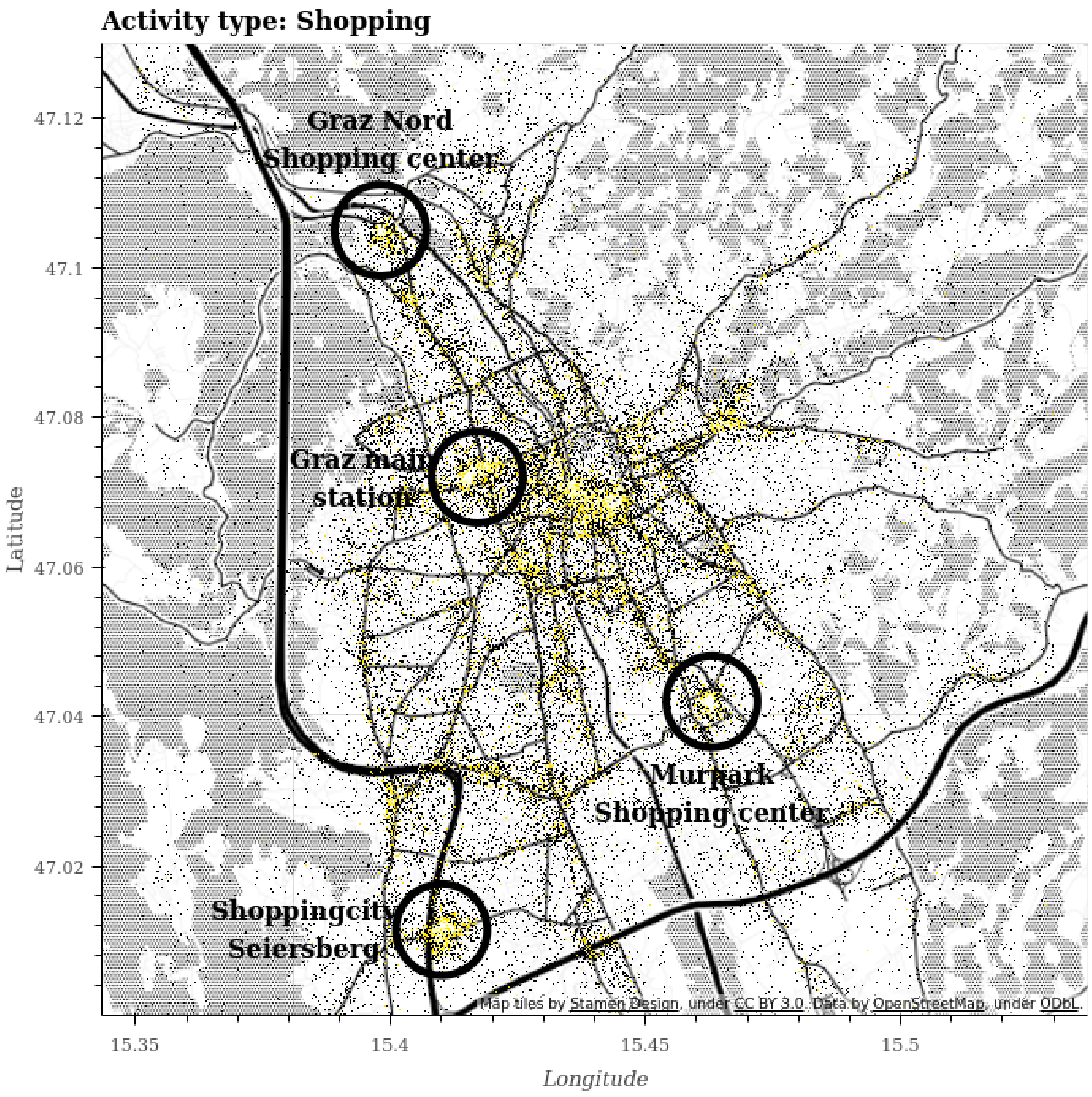

Figure 3.12: Distribution of all detected stationary `Shopping` *segments in the area of Graz*

4 Discussion

4.1 Processing pipeline

4.1.1 Speed filter

As one can see from the filtered speed profile (Figure 3.2, compare to unfiltered, Figure 2.5), all observations with a speed greater than 230 km/h are removed, but the rest of the speed profile is still very noisy.

When comparing the spatial plots before (Figure 1.1) and after (Figure 3.1), a great reduction in outliers, especially those with a big magnitude, can be observed. Which is important, as the current implementation of the Kalman filter assumes that the measurement noise is normally distributed.

4.1.2 Kalman filter

The example of the trajectory filtered with the Kalman filter (Figure 3.3, compare to unfiltered Figure 1.1) shows that the noise is greatly reduced by the filter. In contrast, if the raw data is fairly noisy (Figure 2.2), the Kalman filter could not improve the result much. This is possibly due to the fact that the measurement noise in this case is not anywhere near normally distributed. Analysis has shown, that while the person was at home in the morning, most position estimates have been made a few kilometers to the east. As there is no workaround to this problem as long as the bias is not known, this was communicated to VIAVI (San Jose, CA, USA), the company providing the estimation. According to them they are currently working on a method that provides a greater accuracy while estimating.

Also, when looking at the filtered version of the standard trajectory in the current work (Figure 3.3, the participant was going from north to south on his way `Home` from `Work`), with the selected parameters, the filter seems to be somewhat inert, leading to an overshoot after long straights. This is because the speed v and its corresponding angle ϕ of the internal filter $\hat{\mathbf{x}}$ state can only change due to the process noise $\mathbf{Q}$ in the model. Manual optimization showed that there had to be some trade-off. If one would allow for faster changes reducing the overshoot, this would also lead to additional noise on other trajectories.

Also, manual optimization in itself is a problem as it takes a lot of time and as on optimal solution may not be found. One could think of improving this, for example by trying different error measures such as the ones proposed by Deng et al. [5]. One hast to ensure that the measure also takes the temporal component into account and does not just compare spatial relations.

The current work only performs a qualitative analysis of the filter, as finding an appropriate error measure has proven to be challenging. Analyses to quantify the results are currently being performed.

4.1.3 Segmentation

The values for $\Delta x_{min_stationary} = 700$m and $t_{min\_stationary} = 15$ min found by optimization are similar to the ones suggested in the works of Widhalm et al. [37] (1km, 15 minutes) as well as Alexander et al. [1] (500m, 10min). Again, these boundaries are usually set to smaller values when dealing with GPS data (ca. 100m, also about 10min [9, 25]), but due to the poor spatial and temporal resolution of mobile phone trajectories, looser thresholds ones have to be allowed.

In general, when comparing the correctly segmented trajectory (Figure 3.5, trip journal in Table 2.2) to the one segmented by the developed algorithm (Figure 3.6), it can see that, as expected, `Work` (northmost activity) and `Home` (southernmost activity) were detected, whereas stationary segments result from a change in the mode of transportation were not recognized. This is due to a minimum stationary duration $t_{min\_stationary}$ of 15 minutes.

Looking at the speed profile after Kalman filtering, however, (Figure 2.6) and comparing this to the recorded trip journal of participant 2 (Table 2.2), one can clearly see that the information about the `Tram` trip as well as the stop after the `Train` ride is in the data and could be extracted using a smaller value for $t_{min\_stationary}$.

One might have noticed that the accuracy of the example trajectory is greater than 99% (Figure 3.6) even though four stationary segments have not been detected. This is because all of these segments were shorter than one minute as they denote a change in mode of transportation (see Table 2.2). Due to the nature of the scoring, missing short stationary segments does not reduce the accuracy by much.

4.1.4 Trip purpose imputation

As one can see in the confusion matrix (see Figure 3.7), the classification performance is best for `Home` and `Work` with 90.35% and 77.99% accuracy respectively. This is possibly due to the nature of these activities being usually very long (see Figure 2.9).

With 58.23% the classification of `Education` also works quite well. As one would suspect, its feature values are quite similar to those of `Work` with the main difference being in the `Age`. Further investigation using the WIT showed that for most `Education` samples misclassified as `Work` (16.47%), the age was not in group '$<$ 17'. The expected performance on a real dataset will therefore be lower as only about 30% of all real trajectories have sociodemographic information.

The classes `Private`, `Visiting`, `Personal` and `Leisure` are often mixed. This is the case as the used features do not provide enough information to distinguish between them in most cases (see for example Figures 2.7 and 2.9).

For the `Other` class the main problem lies in the fact that only 0.07% of the classifier training dataset is this class. This also shows that almost all activities can be put into one of the other categories, as especially `Leisure` as well as `Personal` are not so strictly defined.

As a conclusion, one can say that `Home` and `Work` can be distinguished quite best the rest of the activities. The others can most probably not be distinguished with the currently used features as they are too similar.

4.2 Application Example: One day of data

When one first compares the 4.42 average number of observations on the predicted data (Figure 3.8) with the 4.2 in classifier training data (coming from OEU), the result looks quite promising. On the other hand, a big part of the detected trajectories has only one segment, which cannot be seen in the OEU dataset. There only about 2% of all people having only one activity per day. This is most probably due to the poor dataquality in the countryside, resulting in the segmentation algorithm not being able to find more than one stationary segment as the speed of the filtered trajectory is too high.

When looking at the single activity plots of Graz (Figures 3.10, 3.11 and 3.12), one can clearly see a difference in their distributions. In the plots for `Work` and `Shopping` some POIs have been highlighted and will be discussed below.

As a first example, when one compares the plots for `Home` and `Work` for the POI Magna (a big car company), a clear discrepancy can be observed. When looking at hotspots for `Shopping`, the major shopping centers (Murpark, Shoppingcity Seiersberg, Shopping nord) light up as expected. However, also Graz's main train station seems to be very dense. Further investigation, not only restricted to Graz, shows that a lot of activities at public transport hubs are classified as `Shopping`. This is probably due to the lack of a distinct `Transit` class as this is missing from the classifier training data. It would also give an explanation for the over-representation of `Shopping` in the overall activity distribution (Figure 3.9).

One can also observe that no segments were classified `Other` and `Visiting`. For `Other` the answer can simply be that it was underrepresented in the training dataset (0.07%). For `Visiting` no obvious reason was found.

4.3 Improvements

There is still a big potential for improvement in every part of the processing pipeline. For example, at the moment the estimated measurement variance $\mathbf{R}$ is guessed using only the density of the cell tower locations. A more sophisticated model, using also the information about the type and configuration of the used antenna as well as its beam characteristic, could provide a much better and physically accurate value.

Also, at the moment, the filter uses only past information to calculate the current position. If there is no need for the filter to work in real time, one could include future observations to improve the classification, using for example the RTS algorithm [28].

The segmentation algorithm is already quite sufficient and an improvement in Kalman filtering; and as a result, better velocity profile should further increase its accuracy.

By far biggest improvement could presumably be made with regard to classification. Even though the current classifier seems to be at the limit given the current features, other classifiers could be tested. One could think of building a hierarchical classifier, splitting the current multiclass-classifier into separate 'One-vs-rest' classifiers. This would include the benefit that in each stage, different features could be used. Also, after each stage, different checks could be performed. For example, after `Home`-vs-rest, one could check if all detected `Home` segments are within a given radius from another, as most people have only one home. Additionally, if sociodemographic information is added to a trajectory this can also the home municipality, adding even more semantic information. As seen in the previous section, short segments such as shopping or personal cannot easily be distinguished by the current classifier. Therefore, one could use the data from the dataset collected in this thesis to train classifiers lower down in the hierarchy, having the benefit of an accurate location in the training data. This would enable the use of more accurate spatial features such as distance to the nearest shop or educational institution.

Also, instead of the class output, one could look the output probabilities of the softmax layer, introduce a threshold and use only samples with a classification probability above that threshold.

4.4 Outlook

As seen from the results of the real-world example, the proposed system can be used to help modeling commuting traffic in cities, which is one important step in the direction of planning new infrastructure.

Also, in future years, the change from 4G cell networks to 5G will bring a tremendous accuracy improvement. For once because the nature of 5G requires the cell towers to be at most a kilometer apart, but also because the antenna technology changes, enabling better location estimation.

Bibliography

[1] L. Alexander, S. Jiang, M. Murga, and M. C. González. Origin–destination trips by purpose and time of day inferred from mobile phone data. *Transportation research part c: emerging technologies*, 58:240–250, 2015.

[2] A. Allström, I. Kristoffersson, and Y. Susilo. Smartphone based travel diary collection: Experiences from a field trial in stockholm. *Transportation Research Procedia*, 26:32–38, 2017.

[3] W. Bohte and K. Maat. Deriving and validating trip purposes and travel modes for multi-day gps-based travel surveys: A large-scale application in the netherlands. *Transportation Research Part C: Emerging Technologies*, 17(3):285–297, 2009.

[4] C. Chen, H. Gong, C. Lawson, and E. Bialostozky. Evaluating the feasibility of a passive travel survey collection in a complex urban environment: Lessons learned from the new york city case study. *Transportation Research Part A: Policy and Practice*, 44(10):830–840, 2010.

[5] K. Deng, K. Xie, K. Zheng, and X. Zhou. Trajectory indexing and retrieval. In *Computing with spatial trajectories*, pages 35–60. Springer, 2011.

[6] Z. Deng and M. Ji. Deriving rules for trip purpose identification from gps travel survey data and land use data: A machine learning approach. In *Traffic and Transportation Studies 2010*, pages 768–777. 2010.

[7] T. Feng and H. J. Timmermans. Extracting activity-travel diaries from gps data: towards integrated semi-automatic imputation. *Procedia Environmental Sciences*, 22:178–185, 2014.

[8] T. Feng and H. J. Timmermans. Detecting activity type from gps traces using spatial and temporal information. *European Journal of Transport and Infrastructure Research*, 15(4):662–674, 2015.

[9] B. Furletti, P. Cintia, C. Renso, and L. Spinsanti. Inferring human activities from gps tracks. In *Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing*, page 5. ACM, 2013.

[10] L. GONG, T. YAMAMOTO, and T. MORIKAWA. Comparison of activity type identification from mobile phone gps data using various machine learning methods. *Asian Transport Studies*, 4(1):114–128, 2016.

[11] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.

[12] T. Griffin and Y. Huang. A decision tree classification model to automate trip purpose derivation. In *The Proceedings of the ISCA 18th International Conference on Computer Applications in Industry and Engineering*, pages 44–49. Citeseer, 2005.

[13] G. Han and K. Sohn. Activity imputation for trip-chains elicited from smart-card data using a continuous hidden markov model. *Transportation Research Part B: Methodological*, 83:121–135, 2016.

[14] C. Horn, S. Klampfl, M. Cik, and T. Reiter. Detecting outliers in cell phone data: correcting trajectories to improve traffic modeling. *Transportation Research Record: Journal of the Transportation Research Board*, (2405):49–56, 2014.

[15] E. Howe, R. Schönduwe, A. Graff, L. Damrau, and J. Kükenshöner. Smartphones unterstützen die mobilitätsforschung.

[16] L. Huang and Q. Li. Automatic activity identification from raw gps vehicle tracking data, 2010. In *Proceedings of the Canadian Geomatics Conference*, 2010.

[17] M. Janzen, M. Vanhoof, and K. W. Axhausen. Purpose imputation for long-distance tours without personal information. *Arbeitsberichte Verkehrs-und Raumplanung*, 1181, 2016.

[18] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[19] A. Lechner. Aktivitätenerkennung aus mobilfunkdaten-events. Master's thesis, Technical University of Graz, Rechbauerstraße 12, 8010 Graz, 2018.

[20] F. Liu, D. Janssens, G. Wets, and M. Cools. Annotating mobile phone location data with activity purposes using machine learning algorithms. *Expert Systems with Applications*, 40(8):3299–3311, 2013.

[21] Y. Lu, S. Zhu, and L. Zhang. A machine learning approach to trip purpose imputation in gps-based travel surveys. In *4th Conference on Innovations in Travel Modeling, Tampa, Fla*, 2012.

[22] Y. Lu, S. Zhu, and L. Zhang. Imputing trip purpose based on gps travel survey data and machine learning methods. Technical report, 2013.

[23] P. McGowen and M. McNally. Evaluating the potential to predict activity types from gps and gis data. In *Transportation Research Board 86th Annual Meeting, Washington*. Citeseer, 2007.

[24] M. Minsky and S. A. Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 1969.

[25] L. Montini, S. Prost, J. Schrammel, N. Rieser-Schüssler, and K. W. Axhausen. Comparison of travel diaries generated from smartphone data and dedicated gps devices. *Transportation Research Procedia*, 11:227–241, 2015.

[26] L. Montini, N. Rieser-Schüssler, A. Horni, and K. W. Axhausen. Trip purpose identification from gps tracks. *Transportation Research Record*, 2405(1):16–23, 2014.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[28] H. E. Rauch, C. Striebel, and F. Tung. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[29] F. Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[31] L. Shen and P. Stopher. Should we change the rules for trip identification for gps travel records. In *Proceedings of the 36th Australasian Transport Research Forum ATRF, Brisbane, Australia*, pages 2–4, 2013.

[32] L. Shen and P. R. Stopher. A process for trip purpose imputation from global positioning system data. *Transportation Research Part C: Emerging Technologies*, 36:261–267, 2013.

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[34] R. Tomisch and M. Herry. Österreich unterwegs 2013/2014: Methodenbericht zum arbeitspaket „datenverarbeitung, hochrechnung und analyse". 2016.

[35] R. Tomschy, M. Herry, G. Sammer, R. Klementschitz, S. Riegler, R. Follmer, D. Gruschwitz, F. Josef, S. Gensasz, R. Kirnbauer, et al. Oesterreich unterwegs 2013/2014: Ergebnisbericht zur oesterreichweiten mobilitaetserhebung „oesterreich unterwegs 2013/2014 ". 2016.

[36] V. Usyukov. Methodology for identifying activities from gps data streams. *Procedia Computer Science*, 109:10–17, 2017.

[37] P. Widhalm, Y. Yang, M. Ulm, S. Athavale, and M. C. González. Discovering urban activity patterns in cell phone data. *Transportation*, 42(4):597–623, 2015.

[38] J. Wolf, R. Guensler, and W. Bachman. Elimination of the travel diary: Experiment to derive trip purpose from global positioning system travel data. *Transportation Research Record: Journal of the Transportation Research Board*, (1768):125–134, 2001.

[39] J. Wolf, S. Schönfelder, U. Samaga, M. Oliveira, and K. Axhausen. Eighty weeks of global positioning system traces: approaches to enriching trip information. *Transportation Research Record: Journal of the Transportation Research Board*, (1870):46–54, 2004.

[40] G. Xiao, Z. Juan, and C. Zhang. Detecting trip purposes from smartphone-based travel surveys with artificial neural networks and particle swarm optimization. *Transportation Research Part C: Emerging Technologies*, 71:447–463, 2016.