UNIVERSITÄT
BERN

Predicting future locations of mobile users using hybrid features

Bachelor Thesis

Dominic Kohler
from
Worb BE, Switzerland

Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

15. Februar 2018

Prof. Dr. Torsten Braun
Dr. Zhongliang Zhao, Mostafa Karimzadeh

Communication and Distributed Systems Group
Institut für Informatik
University of Bern, Switzerland

Abstract

Predicting future locations of mobile users is beneficial for various applications. For example it can improve the relevance of location based services or forecast crowds. From a real-life data set, we extract mobile-phone users' data and establish categories based on the number of total visits and differences in their movement pattern. The type of a user's movement pattern is either homogeneous (simple,regular) or heterogeneous (complex, non-regular). A rich selection of hybrid features, extracted from the users' history movement data, is used to predict users' future places. To solve the prediction task, we use various machine learning algorithms, both individual and ensemble methods. For the users with a low number of total visits and homogeneous movement pattern an average accuracy of 86.7% is achieved. For users with a heterogeneous movement pattern and a low number of total visits, the average accuracy is 65.41%. Regarding users with a medium number of visits, the highest average prediction accuracy is 85.18% for users with homogeneous movement patterns and 65.02% for users with heterogeneous movement patterns. For users with many visits an average accuracy of 83.98% is achieved for the group with homogeneous movement patterns and an average accuracy of 67.75% for users with heterogeneous movement patterns.

Acknowledgement

I would like to express a special *Thank you* to Zhongliang Zhao and Mostafa Karimzadeh not only for supervising my thesis but also especially for the numerous discussions, explanations and their advice given to me on this work.

Contents

List of Figures

List of Tables

Abbreviations

ANN	**A**rtificial **N**eural **N**etwork
BN	**B**ayes **N**etwork
CDF	**C**umulative **D**istribution **F**unction
LDCC	**L**ausanne **D**ata **C**ollection **C**ampaign
MDC	**M**obile **D**ata **C**allenge
MLP	**M**ulti**l**ayer **P**erceptron
WEKA	**W**aikato **E**nvironment for **K**nowledge **A**nalysis

1 Introduction

Mobility is part of human's daily life. We move from one place to another for different reasons. For example we go to a certain place to work and to another place to meet people. Mobility patterns of users have been studied intensively. The ubiquity of mobile phones with their embedded sensors and available applications opened new possibilities both for academia and industry to study human mobility patterns. With the collected data, it is possible to forecast crowds or to improve the relevance of location based services such as context aware advertising. In this work, we define the future place prediction as a standard supervised learning task. A place visiting is represented by a set of features observed at that given location. These features build the input for the prediction task and future places are the targets. The models used for the prediction task are trained per user with its history data of visited places and the features corresponding to the visited location. A crucial subtask to obtain a high prediction accuracy is to define discriminative and informative features to identify and distinguish between locations a user visits. Furthermore, a proper set of features is important for machine learning algorithms to detect patterns in a user's movement. Given the features, we apply individual as well as ensemble learners to build predictive models. To evaluate the prediction task, we use WEKA [1], an open source data mining software with a wide collection of already implemented machine learning algorithms. We evaluate the performance of the algorithms in terms of accuracy - the number of correctly predicted instances divided by the total amount of computed predictions - and the execution time to solve the prediction task. Our models are validated using data extracted from a real-life data set of mobile users.

1.1 Related Work

Mobility prediction has been an active research field in the last years. Various approaches for solving the future place prediction problem exist. Tran *et al.* [2] use user-specific decision trees learned from temporal features of the each user's history. Etter *et al.* [3] combine multiple mobility predictors, such as graphical models, artificial neural networks or decision trees, by a strategy called blending. To adapt to changes in habits and thus resulting change in movement, they use a technique called aging to give more weight to recent user-data. However, they only use temporal features for the prediction. Other works are limited to use of temporal and spatial features such as Gao *et al.* [4] and Zhu *et al.* [5]. For example, Zhu *et al.* [5] lay a special focus on the acceleration feature. They use the variation in acceleration and average acceleration observed at a visited place. With this feature they want to identify locations related to transportation or a shopping center. Wang *et al.* [6] compute the visit frequency of places as well as the total time spent in places. Further, they try to observe regularity in user behavior and perform periodicity analysis for the top most visited places based on *Fourier Transformation* and *autocorrelation*. For some places, they determine the next place based on the majority voting whereas for other places they consider temporal information to predict the next place. In other works, Markov models are used to predict the next place a user visits [7, 8]. With probabilistic models, [9] and [10] estimate the probability of a user being at a certain location, given a specific time in the future.

Generally, the previous work is mostly limited to temporal or spatio-temporal features. Therefore, we try to exploit a rich type of hybrid features to improve the prediction accuracy of future places. In addition to the features and algorithms used to predict future places, the movement patterns of the users strongly influence the prediction accuracy. Tran *et al.* [2] classified users into several groups based on their movement patterns. A group of users, who have repeated movement patterns, leading to a prediction accuracy over 70%, a group of users whose movement patterns are heterogeneous and a resulting prediction accuracy of nearly 55%, and user groups which members change their behaviour (e.g. change their home) ending in an accuracy of less than 54%. Apart from their work, the classification of users' movement patterns are either limited to homogeneous (repeated, simple movement) or not performed.

1.2 Contributions

In this work, we define the prediction of future places of a mobile user as standard supervised learning task. In a first part, we divide the users based on the number of total visits during the experiment and categorize their movement pattern into homogeneous and heterogeneous. Then we extract a rich type of features to identify different places from a real-life data set of mobile-phone users. We propose to use hybrid features to build a predictive model for the prediction task. The proposed feature-set is tested using both single and ensemble predictors.

The main contributions summarized, are:

- Classification of user movement patterns
- A rich selection of *Hybrid features* to improve prediction accuracy over only *Temporal features*
- Improving the prediction accuracy for users with a low number of total visits

1.3 Overview

The remainder of this work is structured as follows. Chapter 2 presents the characteristics of the data set we use to study the mobility pattern of mobile-phone users. In Chapter 3, we classify the users' mobility pattern into two categories, homogeneous and heterogeneous, based on the number of visited places, and specify the set of features intended to use for the prediction task. Given the selected features, we evaluate and discuss the prediction of future locations using single and ensemble predictors, in Chapter 4. Finally, Chapter 5 concludes our work.

2
Data set

2.1 Lausanne Data Collection Campaign

In January 2009 Nokia Research Center Lausanne (NRC), Idiap Research Institute, and École Polytechnic Fédéral de Lausanne (EPFL) started an initiative to create large-scale mobile data research resources. This initiative included the design and implementation of Lausanne Data Collection Campaign (LDCC), in which mobile-phone data from about 185 volunteers were collected during the period of October 2009 until March 2011 (18 months) in the Lake Geneva region. The participants were provided with a Nokia N95 phone, which was equipped with an application to log activities 24 hours a day. The logged data was first stored in the device and later, as soon as a successful WLAN-connection could be established, transferred to a Simple Context server. The server then build a database with the received data. This architecture allowed the collection of data to be invisible to the users. This collection of data lead to the Mobile Data Challenge (MDC) data set, which provides the data used in this work.

2.2 Data Characteristics

The MDC data set consists of mainly 7 different categories of data, as visible in Figure 2.1. All data stored in this database correspond to a place a user has visited during the experiment.

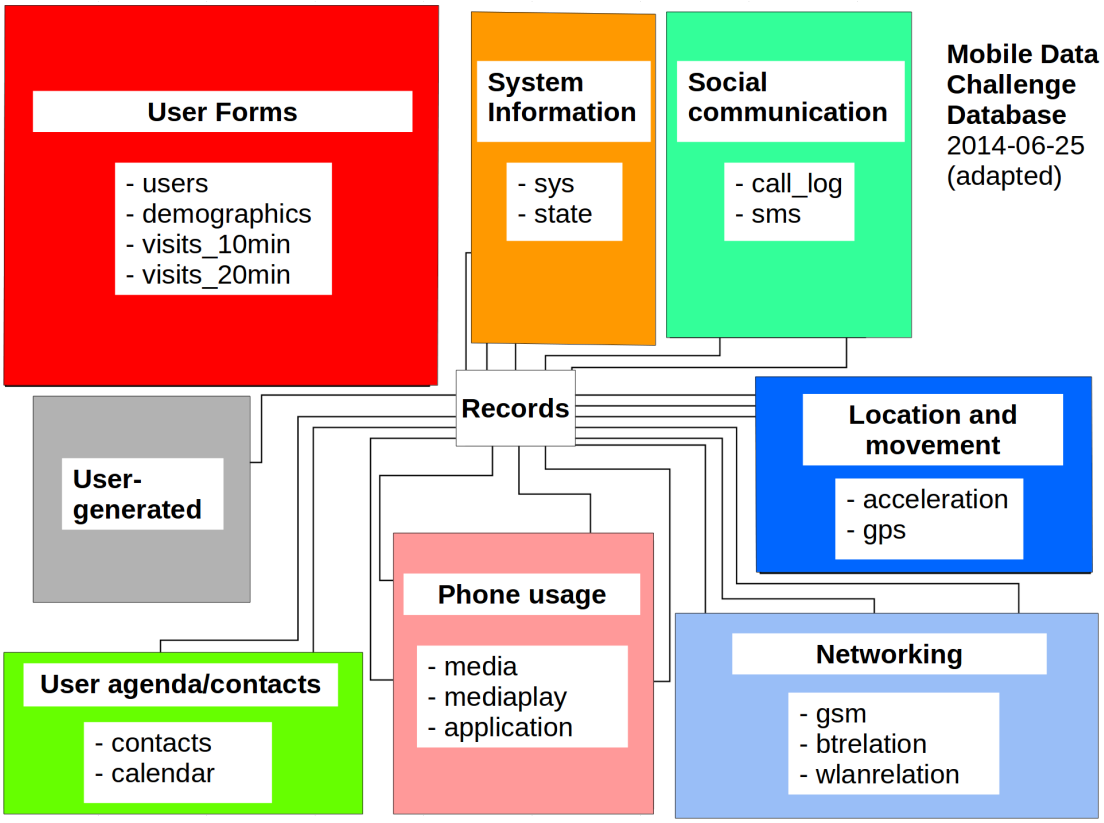

Figure 2.1: MDC data set overview of the most important tables

The red box, *user forms*, contains demographic descriptions about the user and temporal information about his/her visited places. Concretely, for the places users have visited, there exist two tables: the table *visits_10min* with all visits of a duration of at least 10 minutes and another table, *visits_20min*, with all the visits of a duration of at least 20 minutes. With the second table being a subset of the first. Both tables contain attributes like *user_id*, *place_id*,*time_start* and *time_end*. In our work, we use the table *visits_10min*, since in this table contains more records than the table *visits_20min*.

A further box, *system_information* (yellow), involves information about the system (=mobile-phone) and its state. The table *sys* consists of attributes like *phone-profile*, *battery*, *charging*-status and *freeram*. The attribute battery displays the percentage of remaining battery power. Charging-status shows if the phone is charging or not. Freeram indicates the amount of free RAM available.

Data about *social communication* are stored in the tables *call_log* and *sms*. The table *call_log* includes meta-data about phone calls. Crucial attributes, which build the columns

of the table, are the *time* a call started, *direction* of the call, whether it is incoming or outgoing, the *duration* of the call and the *number* calling or called.

In the royal blue box, data about locations and movement are aggregated. GPS and acceleration data are included. In the original competition of the MDC, GPS data was not given to the researchers. The table *accel* involves data relating the motion of a user and his/her phone, respectively. An entry in this table consists of a start and end time as well as the acceleration.

The box in the right corner at the bottom, *Networking*, the tables gsm, btrelation and wlanrelation are included. They provide information about network connections and available devices. In the table *gsm* the id of the GSM-antenna as well as the signal strength are stored. Similarly, the table *btrelation* contains the id and name of the network for bluethooth data. Data related to WLAN connections are stored in the table *wlanrelation*. The most important attributes in this table are the SSID and the security, whether the network is encrypted or not.

In the pink colored box, *Phone usage*, the tables *mediaplay* and *application* store data about media usage behavior. The table mediaplay covers attributes like title, artist and duration of the medium played. More general information is contained in the table *application*. Attributes like event and name describe which application has been opened, closed or is in use.

The last box, *user agenda/contacts*, aggregate data about the user's contacts and his or her personal calendar. In the table calendar, data about the begin, location and type of the event is stored.

With two thirds of the users between 22 and 33 years old, the data set is biased towards younger people and with 68% of male versus 32% female participants towards male users.

3

Approach: User Classification, Features & Algorithms

3.1 User classification

The quality of the user data greatly varies, which impacts prediction performance. For example the number of total visits during the experiment ranges from 90 to 1992. The number of total visits is of interest since users with a high number of total visits have generally more distinct visited places than users with a small total of visited places, wherein the prediction task gets harder. Therefore, in this work, the users are first split into groups based on the number of total visits during the experiment, with a duration of at least 10 minutes at a certain place. The group *few visits* is for users with a total of less than 800 visits, another group, *medium*, is for users with 800 - 1'500 visits, and the last group, *many visits*, involves users with more than 1'500 total visits. Within the group, the users are further categorized into homogeneous or heterogeneous movement pattern. A user's movement pattern is homogeneous if the visits are distributed over a few places or he/she does not move a lot. If the visits of a user are distributed over a lot of places or he/she moves a lot, without strong regularities in his/her visting of locations, the movement pattern is said to be heterogeneous.

This leads to the below listed categorization:

- Few visits (< 800)
  - Homogeneous movement
  - Heterogeneous movement
- Medium number of visits ($800 - 1'500$)
  - Homogeneous movement
  - Heterogeneous movement
- Many visits ($> 1'500$)
  - Homogeneous movement
  - Heterogeneous movement

In the following, we explain in more detail, how we categorize the movement pattern of the users in the different groups.

For the group with few visits the percentage of visits distributed over the top 10 most visited places is calculated. If the number of visits at the top 10 most visited places divided by the total visits is more than 90%, the user's movement pattern is classified as a homogeneous movement pattern. Otherwise, users with a value less than 70% are assigned a heterogeneous movement label. The movement pattern of users covering 70%-90% of their visits in the top 10 most visited places is difficult to classify. The method of the percentage of top 10 visited places is not accurate enough, as there are users with up to 20% difference in prediction accuracy, despite having about the same percentage of visits in the top 10 most visited places ($\pm 2\%$). To classify their movement pattern, more sophisticated techniques, which are beyond of the scope of this work, would be necessarily.

For the users with more than 800 visits the value of the Cumulative Distribution Function (CDF) of distinct visited places is calculated to classify the users' movement pattern. If the CDF value is low (< 0.5), the user has visited few distinct places and therefore a high prediction accuracy is expected. In contrast, users with a high CDF (> 0.5) value have a more complex movement pattern and therefore the prediction accuracy is expected to be lower. The number of distinct visited places ranges from 104 to 238 as visible in Figure 3.1.

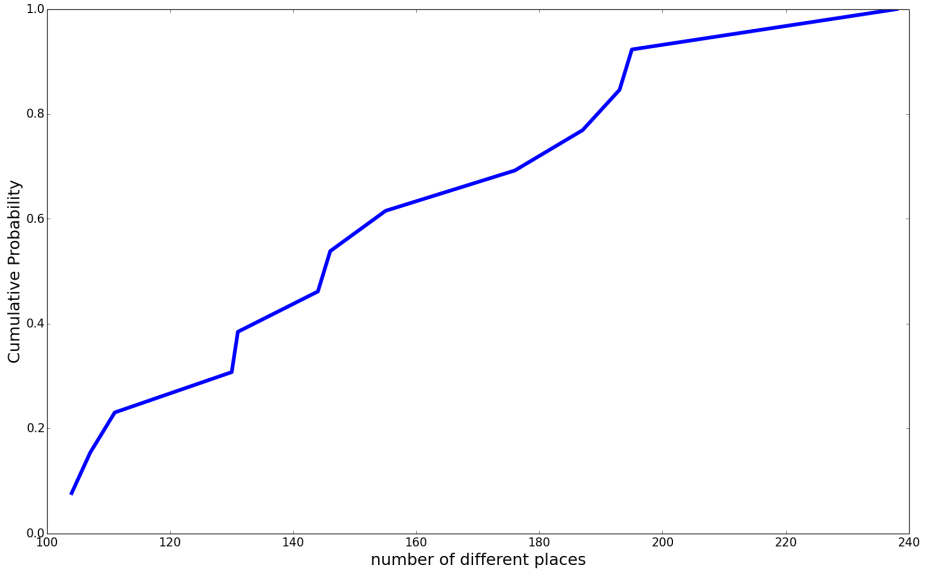

Figure 3.1: CDF of visited places for users with many visits

For some users, the CDF value is not an accurate metric because they have more randomness and less repetition in their movement patterns. Hence, if the accuracy of predicted place-IDs is low, despite a low CDF value (or vice versa, high CDF value and high accuracy), the radius of gyration is calculated to finer describe the movement pattern of the user. The radius of gyration [11, 12] describes how widely a user moves. Figure 3.2 graphically displays the radius of gyration. For a user the center of mass point (r_cm) of his/her trajectory is calculated. Then, for every visited location (r_i, the turquoise dots) the distance between the visited place and the center of mass point (r_i - r_cm) is computed to describe the displacement of the visited location. The displacement of the visited locations are averaged and result in the radius of gyration (r_g).

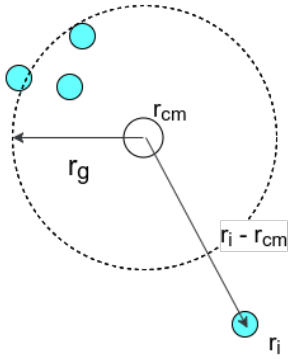

Figure 3.2: Radius of gyration

More generally, the radius of gyration is defined as,

$$r_g = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\vec{r_i} - \vec{r}_{cm})^2} \tag{3.1}$$

for $\vec{r_i}$ the $\{i = 1, 2, ..., n\}$ visited places for a given. $\vec{r}_{cm} = \frac{1}{n}\sum_{i=1}^{n} r_i$ the center of mass point of the user's trajectory. To calculate the distance between the center of mass point and a visited place, the $haversine$ [13] formula is used. The distances (r_i - r_cm) are summed up and averaged. Finally, the square root of the average distance is taken. We calculate the radius of gyration per day and build the average over the period of time the user participated in the experiment.

With the help of this method it can be explained that the prediction accuracy for the users despite a low CDF value is low because they have a wide radius of gyration. Contrary for the users with the prediction accuracy expected to be low due to their high CDF value, but a high actual prediction accuracy, it can be explained with their narrow radius of gyration.

3.2 Features for Place Definition

To achieve high prediction accuracy, it is essential to choose discriminative and informative features. The MDC data set offers a rich type of data. However, some categories suffer from sparsity. We focus on data which is of good quality and available for most of the users. Different types of features may provide different information. For example, features containing information related to the time a user visits a place can indicate whether a person is likely at a workplace. Yet, temporal information alone do not suffice to identify and distinguish all places. For example, if a user does not work on a regular basis, his home and his work place could easily be confused. Contextual data of the visited place, such as phone-profile or visible bluethooth devices, can provide further information about the location of a user. Therefore, we intend to use temporal, system, networking and movement features, which we describe in the following subsections. All features combined together (temporal + system + networking + movement) are regarded as hybrid features.

3.2.1 Temporal Features

Temporal features describe characteristics regarding the time of a visit. Users tend to spend longer durations at home and at their work place, whereas stays at a restaurant or transportation are of shorter duration. Furthermore, the mobility pattern during the week is generally different compared to the movement behavior on a weekend. For example, people usually work from Monday to Friday and spend time in a shopping center on a weekend.

Duration: How long a user stays at a given place.

Leavingtime: At what time the user leaves a given place

Weekday: At what day of the week, the user visits a place.

Weekend: If the visit is on Saturday or Sunday

3.2.2 System Features

System features include context information describing the state of the mobile-phone's system. In this work, we use the following two features:

Profile: The phone-profile the user has selected. With possible values being $general$, $silent, meeting$. The difference between the profile *silent* and *meeting* is that the user can configure the phone to ring if a certain person calls, whereas the the phone does not produce any sounds if the profile is set to *silent*.

Charging: If the user is charging his/her phone at the given place.

People tend to charge their phone at home or in the office, where they stay for a longer duration. The phone-profile may indicate if the user is not to be interrupted and likely working.

3.2.3 Networking Features

Networking features include context information relating visible devices and available connections. In our work, we make use of *GSM*, *Bluethooth* and *WLAN* data. Information regarding visible bluethooth devices can reveal how crowded a place is. For example the number of visible bluethooth devices is expected to be high in a shopping center, restaurant or transportation.

#Visible WLAN-SSIDs: The number of detected WLAN SSIDs at a given place

#Visible GSM-Cells: The number of visible GSM-Antennas from a given place

#Visible Bluethooth Devices: The number of visible bluethooth devices visible from a given place.

3.2.4 Movement Features

Movement features include information relating the user's movement, such as acceleration, at a given place.

Acceleration: If the average acceleration at a certain place is higher than a predefined threshold.

PlaceFeature	Leaving Time	Duration	Weekday	#Visible WLAN SSIDs	#GSM cells	#Bluethooth Devices	Charging	Phone profile	Acceleration
Home	19:00 - 09:00	$[480, 2880)$	MON - SUN	$[1, 5)$	$[1, 3)$	$[0, 6)$	$True$	Normal	low
Work	09:00 - 12:30	$[120, 480)$	MON - FRI	$[6, 10)$	$[3, 6)$	$[6.15)$	$True$	Silent Meeting	low
Restaurant	12:30 - 14:00 19:00 - 23.00	$[40, 120)$	MON - SAT	$[6, 15)$	$[3, 6)$	$[6,)$	$False$	Normal	low
Transportation	07:00 - 08:30 18:00 - 19:00	$[10, 40)$	MON - SUN	$[18,)$	$[6,)$	$[15,)$	$False$	Normal	high
Outdoor Sports	-	$[40, 120)$	SAT - SUN	-	$[1, 3)$	-	$False$	Normal	high
Indoor Sports	19:00 - 22:00	$[40, 120)$	MON - SUN	$[1, 5)$	$[1, 3)$	$[0, 6)$	$False$	Normal	high
Shopping Center	-	$[40, 120)$	FRI - SAT	$[15,)$	$[1, 3)$	$[15,)$	$False$	Normal	low
Friend Home	19:00 - 23:00	$[40, 480)$	MON - FRI	$[1, 5)$	$[1, 3)$	$[0, 6)$	$False$	Normal	low
Friend Office	-	$[10, 60)$	MON - FRI	$[6, 10)$	$[3, 6)$	$[6.15)$	$False$	Normal	low

Table 3.1: Place-Feature correlation

3.2.5 Feature Selection

Having extracted the features, we want to know if they are useful for the prediction task or not. Hence, we use the method InfoGainAttributeEval [14] from WEKA [14] for every feature to calculate its *Information Gain*. The information gain describes how much information a feature gives about the class we want to predict. It is a value greater than or equal to 0 and bounded from above by the class label. A value of 0 indicates that the feature does not provide any information to classify an instance and is therefore not useful. The bigger the information gain of a feature is, the more it reduces uncertainty about the classification. In table 3.2 the features and its corresponding information gain is ranked in decreasing order. Since all features have an information gain bigger than 0, we use all of them for the prediction task.

Feature	**Information Gain**
#visible WLAN SSIDs	1.536
#visible GSM-Cells	0.852
duration	0.835
leaving_time	0.700
#visible Bluethooth devices	0.611
charging	0.290
weekday	0.212
profile	0.209
acceleration	0.141
weekend	0.042

Table 3.2: Information Gain per feature

3.2.6 Features not Selected

As previously mentioned, not all data is of good quality and therefore not beneficial for the prediction task. The following describes which data we do not use.

Calendar data involve personal events a user stores in the mobile-phone calendar. These type of data suffers from sparsity. A lot of users do not have enough or not even have entries at all, in this table. Consequently, we do not consider to use this kind of data.

Sms and call_log build the social communication data in the MDC data set. Originally, we thought that communication activities could reveal important information about the visited places. For example, missed phone calls could indicate that a mobile-phone user is working, or when sending text messages that the user is traveling by

bus or train. Having those data inspected deeper we concluded not to make use of this data due to its sparsity and because we cold not detect regularities.

Applications used by the user at specific place. These type of data could reveal vital information about the places. Assuming that the user opens and runs an application to track his or her sport activities, that would help identifying places where a user practices sports. Eventually, we decided not to further extract data about the users' application since we use the feature acceleration to detect places related to sport activities and also because a lot of user have only standard system-apps running.

3.3 Algorithms

In the following, we shortly describe the different families of classifiers and describe the specified parameters used in the experiment. Each of the algorithms are trained and evaluated using 10-fold stratified cross-validation. This means that the constructed data set for a given user is partitioned into 10 evenly sized parts. One subsample is used for validation while the remaining 9 subsamples serve for training the classification model. This process is repeated 10 times, so that every subsample is once used for validation and serves 9 times as part of the training set.

3.3.1 Decision Trees

Decision trees are a tree like structure composed of a root node, branches and leaf nodes. With the root node containing all instances and representing the first split-attribute. Other non-leaf nodes of the tree represent tests based on attributes. For every possible test outcome, there exists a branch. The leaf nodes describe the class label [15]. From a given training set and its attributes the decision tree learns rules to split the instances into as pure as possible leafs. This means, the leaf should preferably only have instances of one distinct class label to make an accurate prediction. In this work, we have *J48* configured to use the class label that occurs most frequently, if a leaf node is impure - containing instances of more than one class - whereas the *Hoeffding Tree* is configured to use an adaptive *Naive Bayes* approach if the leaf node is impure. To reduce the models complexity by removing branches of the tree a technique called pruning is used, which should help prevent overfitting. WEKA uses statistical test for pruning to remove insignificant branches. The confidence factor is set to 0.25.

- *J48*, confidence factor = 0.25, min. number of objects = 2, leaf prediction strategy: majority class
- *Hoeffding Tree*, leaf prediction strategy: Naive Bayes adaptive, split Confidence 1.0E-7

- *Random Forest*, number of iterations = 100

3.3.2 Bayesian Approach

A Bayesian classifier is a probabilistic classifier that uses Bayes' theorem [16]

$$P(H|E) = P(E|H) * Prob(H)Prob(E)$$

with H being a future place and E being the current context (features). $Prob(H)$ is therefore the a priori probability of H and $Prob(E|H)$ is the a posteriori probability of H. In this work, the two following algorithms are used:

- *Naive Bayes*
- *Bayes Network*

Naive Bayes is called naive since it assumes independence between the features. In our formulated task, it is not important to calculate exact probabilities but to estimate the highest probability for a future place correctly [17]. This means that even if under the assumption of independence between the features the calculated probability is not the same as the real probability, but we can still obtain a correct prediction as long as calculated probability is the highest for the future place with real highest probability.

3.3.3 Functions

Multilayer Perceptron is a feedforward artificial neural network (ANN). An artificial neural network is a model used to solve computations for broad variety of tasks. It consists of connected units, so called artificial neurons. The neurons are grouped into a layer. The connections between layers can send signals (numbers) to other layers. In this context, feedforward means that no circular connections between different layers exist. Based on the signal strength and a threshold value, the neuron might be activated, process the signal and send it to other layers. Finally, there is an output function computing the output from the activation. [18]

- *MLP*, training time = 500 sec, hidden layers: automatic

3.3.4 Ensemble Methods

Instead of a single learner, ensemble methods use multiple learners to solve the prediction task [19]. Diverse machine learning algorithms are combined together to build one final model. The fact that different algorithms may produce different prediction outputs is exploited to improve prediction accuracy. We briefly explain the three different ensemble methods *boosting*, *bagging* and *stacking*.

Boosting is used to reduce bias and variance in a supervised learning task. So-called weak learners are combined into a strong classifier [20] with methods, such as for example weighted average or voting.. The version implemented in WEKA is *AdaBoostM1*, which stands for Adaptive Boosting [21].

- *AdaBoostM1*, number of iterations = 10

Bagging stands for Bootstrap Aggregation. It is a method to generate multiple versions of a predictor and using them to get an aggregated predictor. Multiple versions of the predictor are obtained by making bootstrap replicates of the training set. These replicates are the used as new trainings set. In the classification task, the aggregation does a plurality vote to predict the class. Bagging can improve the accuracy, if the replicated training sets can cause significant changes in the constructed predictor. [22]

- *Bagging*, number of iterations = 10

Stacking is an ensemble learning method to combine multiple algorithms. Several so-called base-learners are trained on a test set. With the obtained prediction results, a further algorithm, so called meta-learner then chooses for the instance to classify the best suited base-classifier [23].

4 Evaluation

In this chapter the evaluation of the future place prediction is presented. The metrics used for evaluating the prediction task are average prediction accuracy, which is defined as the number of correctly predicted instances divided by total predicted instances, as well as average prediction execution time. In a first part, the classifiers are evaluated without an ensemble method. In a second part, we make use of the ensemble learning methods boosting, bagging and stacking.

4.1 Hardware

For the evaluation of the selected algorithms the following computing resources are used:

- 5 cores Intel Xeon E312xx @ 2GHz
- 16 GB RAM

on Ubuntu 14.04 LTS.

4.2 Software

To evaluate the task of predicting future locations of a mobile user we use WEKA 3.8 [1] in this work. WEKA is an open source data-mining Software written in Java and provides a rich collection of standard machine learning algorithms. These algorithms

can either be used in WEKA itself or included in a Java program. We included WEKA in our Java code to evaluate the future place prediction of a user more comfortably.

4.3 Accuracy of Single Predictors

4.3.1 Temporal Features

If we only consider temporal features for the users with few visits and homogeneous movement, *J48* achieves the highest accuracy (66.35%) of all classifiers. For users with a heterogeneous movement pattern, the *Naive Bayes* performs best with 42.48% accuracy. Between the classifiers, there is not much variance in prediction accuracy ($\pm 4\%$ for homogeneous and heterogeneous movement).

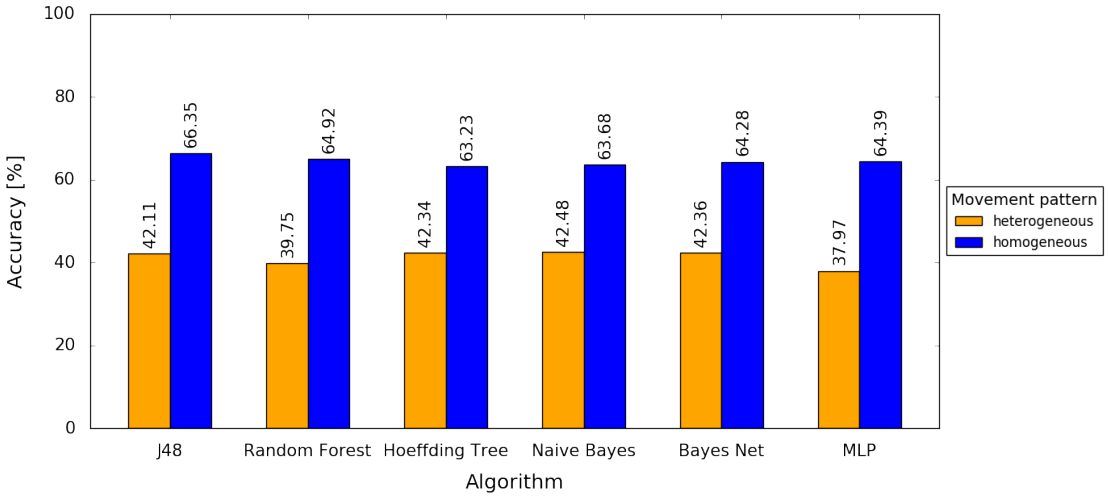

Figure 4.1: Average prediction accuracy using temporal features for users with few visits

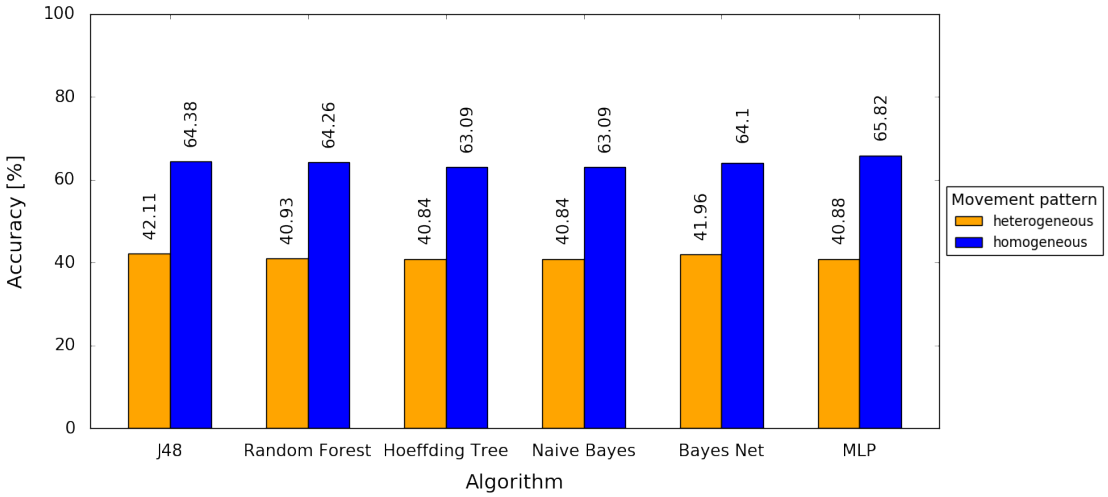

Figure 4.2: Average prediction accuracy using temporal features for users with a medium number of visits

The highest prediction accuracy for users with a medium number of visits is displayed in Figure 4.2. For users with homogeneous movement patterns the highest average accuracy is achieved by *MLP* with almost 66%. Considering heterogeneous movement patterns, *J48* achieves the highest average accuracy with 42%. The average prediction accuracy is very similar to the group of users with few visits.

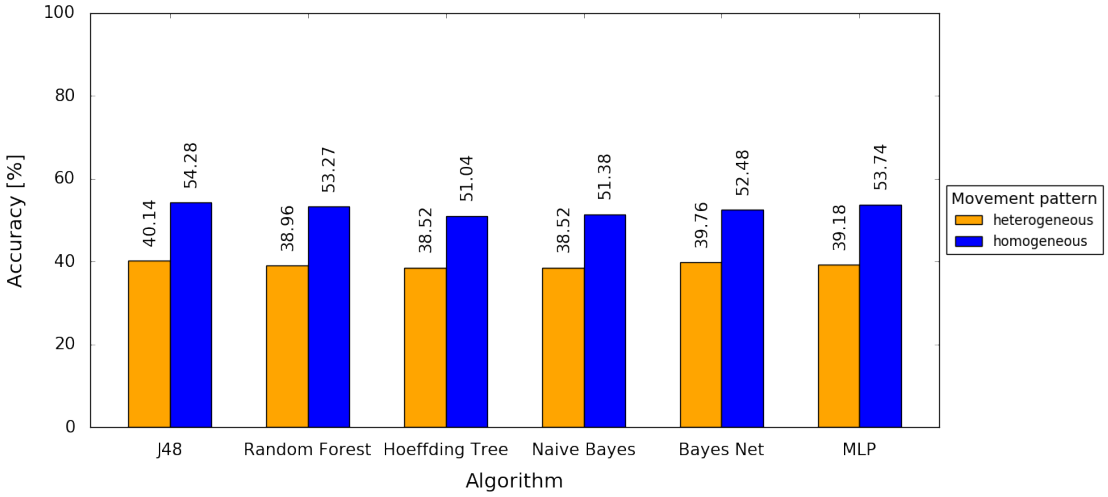

Figure 4.3: Average prediction accuracy using temporal features for users with many visits

In Figure 4.3 it is visible that *J48* scores the highest average prediction accuracy for both homogeneous (54.28%) and heterogeneous (40.14%) movement patterns, for

users with many visits. The average prediction accuracy is roughly 25% higher for users with homogeneous movement pattern compared to users with heterogeneous movement pattern. The average prediction accuracy for users with homogeneous movement patterns and many visits is roughly 10% lower compared to the groups of users with homogeneous movement and few or a medium number of visits.

4.3.2 Hybrid Features

In this subsection, we consider all hybrid features specified in subsection 3.2 for the prediction task.

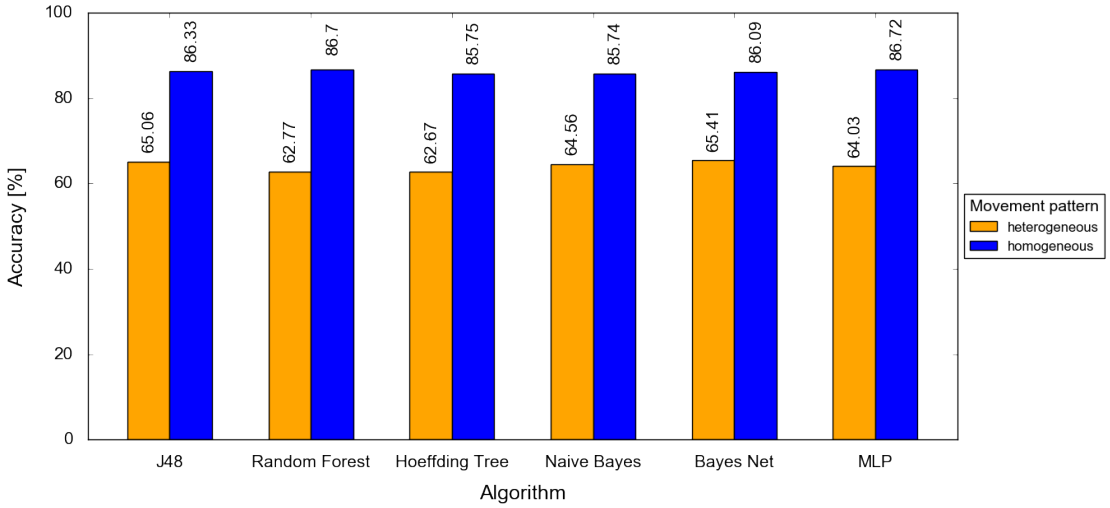

Figure 4.4: Average prediction accuracy using hybrid features for users with few visits

The average prediction accuracy for users with few visits and homogeneous movement gets as high as 86.72% (*MLP*). This is an improvement of 20% over using just temporal features. For the heterogeneous users, the highest accuracy is obtained with *Bayes Net*, 65.41%. For those users the accuracy improved up to 25% when considering hybrid features compared to using temporal features for prediction. When we compare the homogeneous movement pattern versus the heterogeneous movement pattern for users with few visits, the accuracy in the average 20% higher for the homogeneous users.

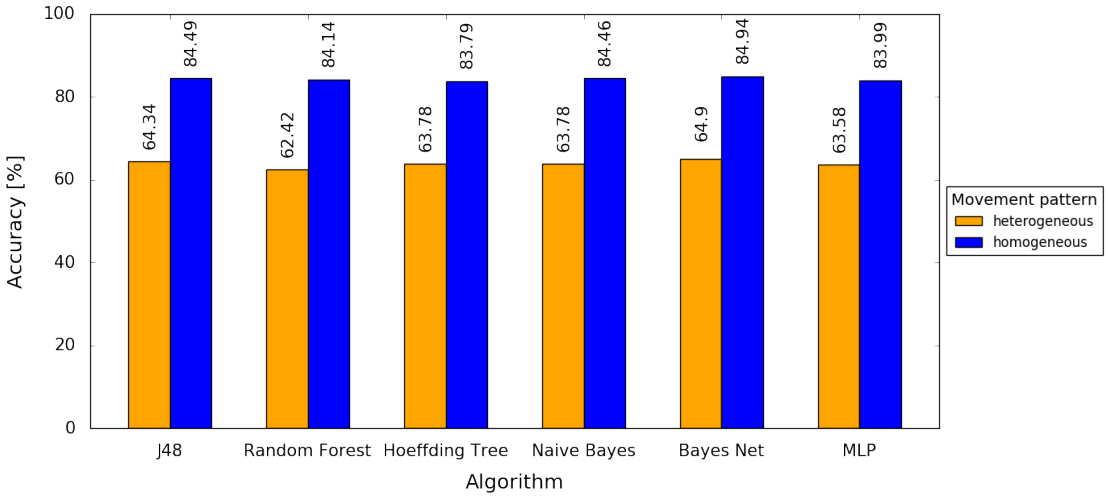

Figure 4.5: Average prediction accuracy using hybrid features for users with a medium number of visits

In Figure 4.5 the average prediction accuracy for users with a medium number of visits is visible. For users with homogeneous movement patterns *Bayes Net* performs best with an accuracy of nearly 85% closely followed by *Naive Bayes* and *J48*. When looking at users with heterogeneous movement patterns it is also *Bayes Net*, which performs best. The variance between the different algorithms is less than 2%.

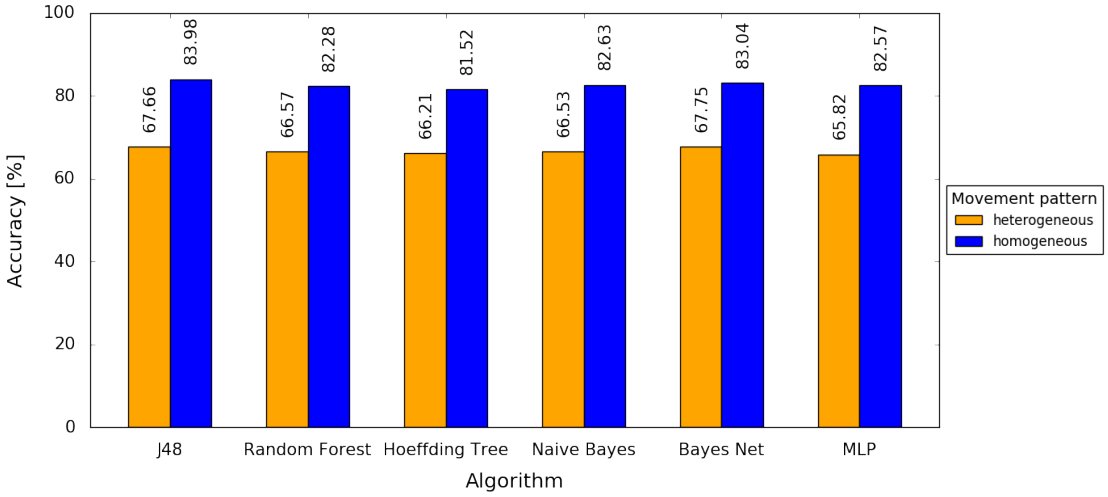

Figure 4.6: Average prediction accuracy using hybrid features for users with many visits

Regarding the users with many visits, the accuracy is 84% for users with a homogeneous movement pattern. For the heterogeneous group, the accuracy is almost 68%.

Again, the average accuracy for the users with homogeneous movement is almost 20% higher than the accuracy for users with heterogeneous movement. For the users with homogeneous movements the accuracy improves by 18% when considering hybrid and not just temporal features. Even more, i.e. by 28%, the average accuracy improves for users with heterogeneous movements when considering hybrid features over just temporal features.

4.4 Accuracy of Ensemble Predictors

In this section we analyze the average prediction accuracy when using hybrid features and the ensemble learning boosting, bagging and stacking applied to different algorithms. We do not further consider solely temporal features, since the accuracy is up to 30% higher when using hybrid features.

4.4.1 Hybrid Features

In Figure 4.7 the average prediction accuracy using boosting and different algorithms for users with few visits is presented. The highest average prediction accuracy for the group of users with heterogeneous movement patterns is 64.04% and obtained when *boosting* is applied to *MLP*. With an average of 87.03%, *boosting* applied to *J48* scores the highest accuracy for users with a homogeneous movement pattern.

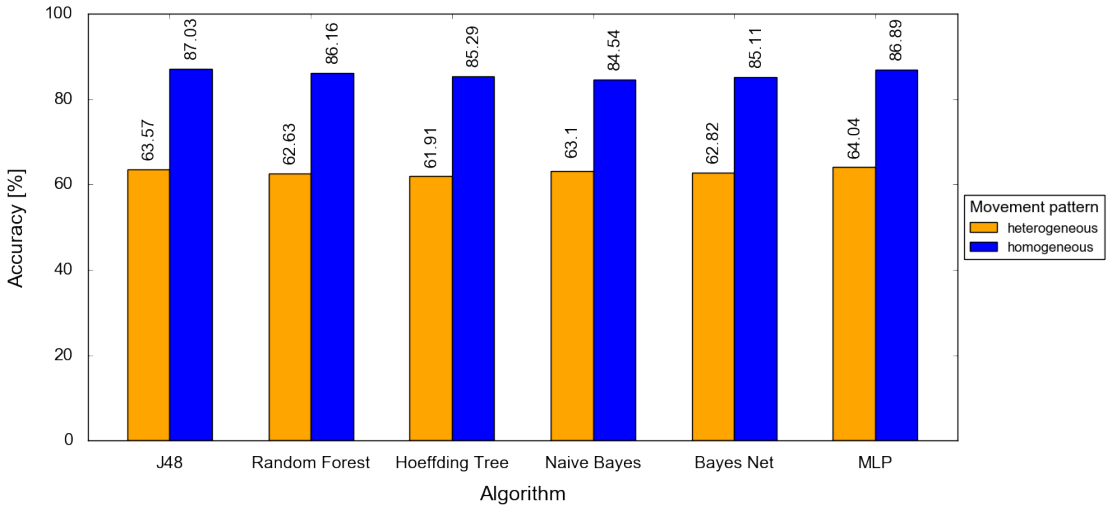

Figure 4.7: Average prediction accuracy using hybrid features and *boosting* for users with few visits

For users with a medium number of visits and homogeneous movement patterns, the ensemble method *boosting* performs best when applied to *J48*, with an average accuracy of 84.6%. Considering users with heterogeneous movement patterns, *boosting* performs best, with an average accuracy of 64.3%, when applied to *MLP*.

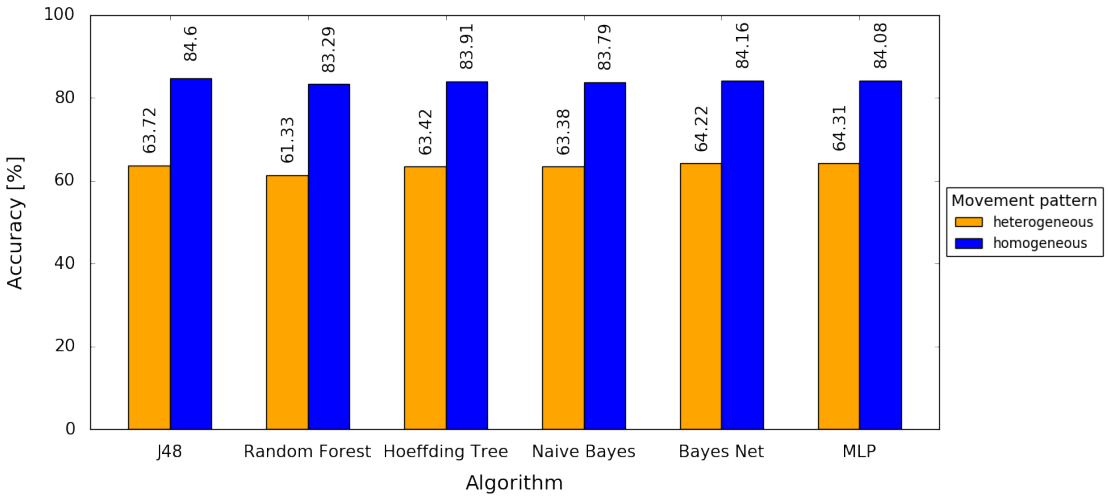

Figure 4.8: Average prediction accuracy using hybrid features and *boosting* for users with a medium number of visits

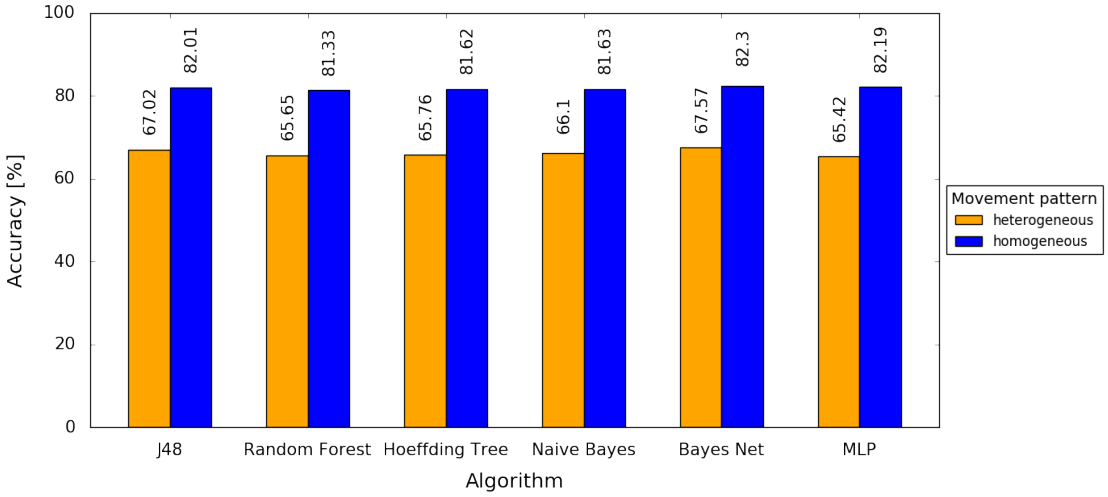

Figure 4.9: Average prediction accuracy using hybrid features and *boosting* for users with many visits

Regarding users with many visits (Figure 4.9), *boosting* applied to *Bayes Net* achieves the highest accuracy (82.3%) for the heterogeneous movement pattern group and 67.57%

accuracy for the heterogeneous movement group. Between the different algorithms combined with *boosting*, there is not a big difference in the average prediction accuracy, i.e. $\pm 2\%$.

In Figure 4.10 the average prediction accuracy for users with few visits is presented when bagging is applied to different algorithms. The highest accuracy is achieved by *MLP* with 87.04% for the homogeneous movement pattern and 65.71% for the heterogeneous movement group. Also when bagging is used, there is no big difference between the different classifiers.

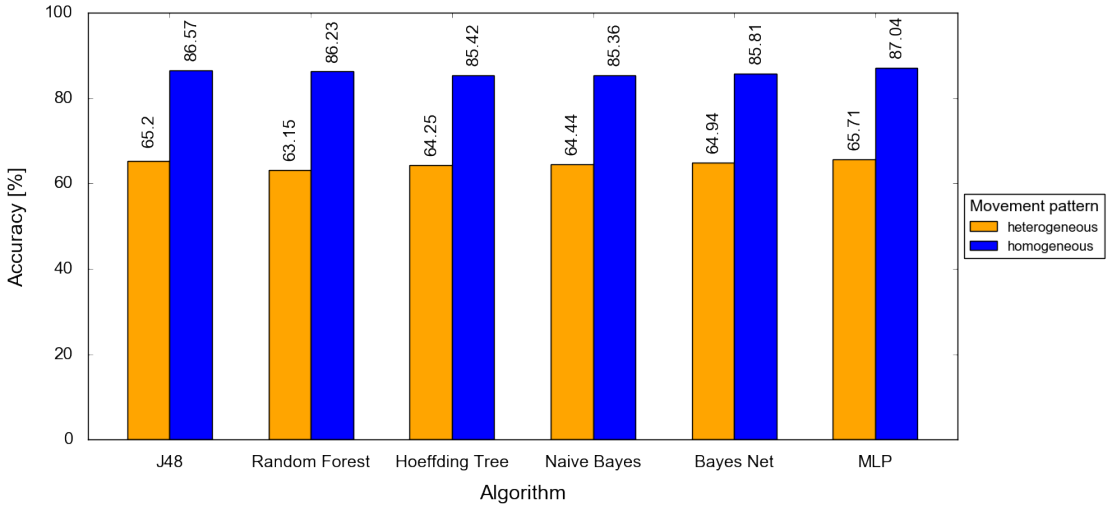

Figure 4.10: Average prediction accuracy using hybrid features and bagging for users with few visits

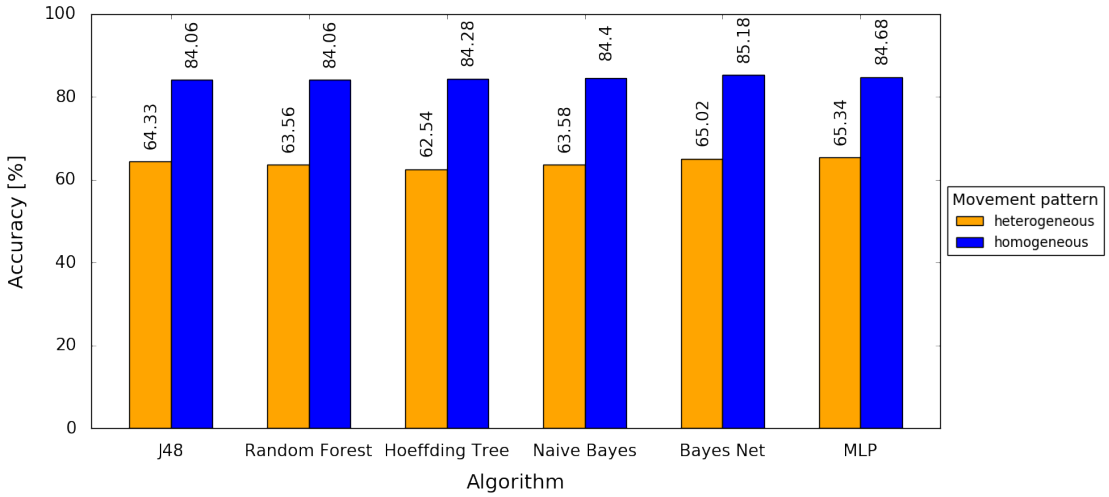

Figure 4.11: Average prediction accuracy using hybrid features and bagging for users with a medium number of visits

The ensemble method bagging performs best, for users with a medium number of visits and homogeneous movement patterns, when combined with *Bayes Net* with an average accuracy of more than 85% (see Fig. 4.11). Considering users with heterogeneous movement patterns, bagging performs best when applied to *MLP* (65.34% accuracy) closely followed by *Bayes Net* (65.02%).

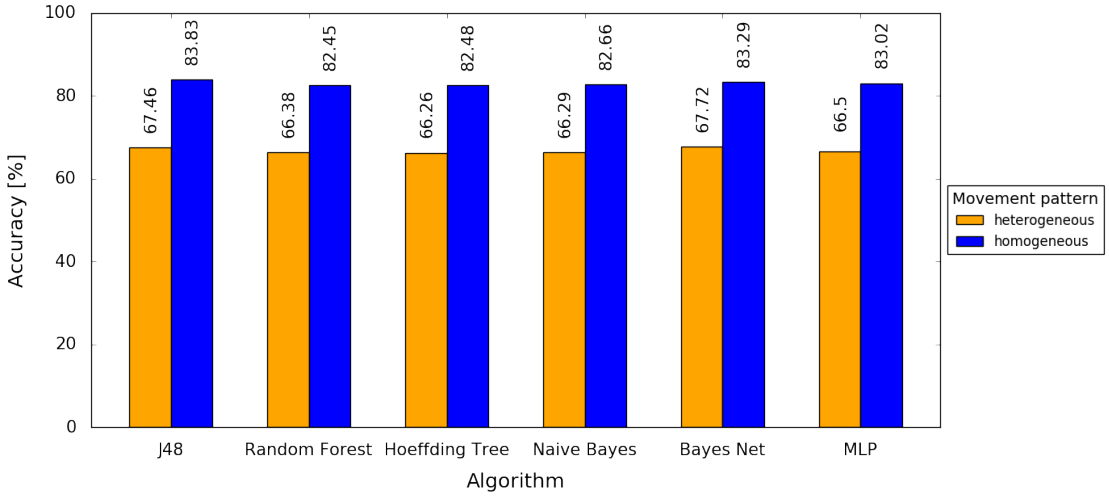

Figure 4.12: Average prediction accuracy using hybrid features and *bagging* for users with many visits

In Figure 4.12 the average prediction accuracy of bagging in combination with different algorithms for users with many visits is presented. For the group of users with

heterogeneous movement patterns *Bayes Net* performs best with an average accuracy of 67.72%, closely followed by *J48*. Bagging combined with *J48* achieves the highest accuracy, almost 84%, for users with homogeneous movement patterns.

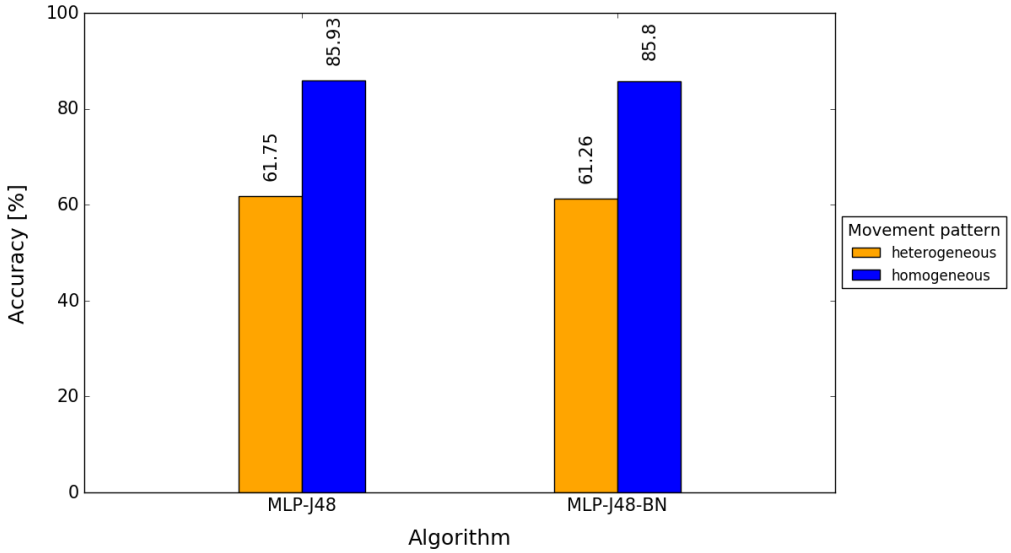

Figure 4.13: Average prediction accuracy using hybrid features and *stacking* for users with few visits

The average accuracy for users with few visits and heterogeneous movement is about 62% when using using *stacking* to combine *MLP-J48-BN* or *MLP-J48*, as visible in Figure 4.13. The average accuracy is 24% higher for the users with a homogeneous movement pattern. Overall, the average accuracy does neither improve for the heterogeneous movement pattern group nor for the homogeneous one, compared to using a single classifier.

Regarding users with a medium number of visits and the ensemble method stacking, the average prediction for users with homogeneous movement patterns is almost 2% lower compared to the users with few visits and homogeneous movement patterns. Conversely, the average prediction accuracy using *stacking* for users with heterogeneous movement patterns is slightly higher compared to the users with few visits and heterogeneous movement patterns, when *MLP*, *J48* and *Bayes Net* are stacked.

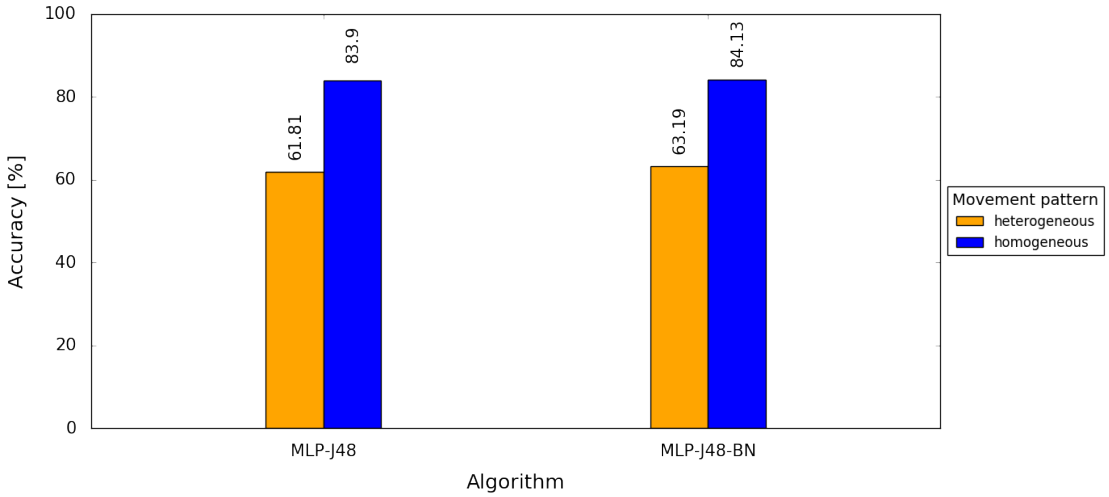

Figure 4.14: Average prediction accuracy using hybrid features and *stacking* for users with a medium number of visits

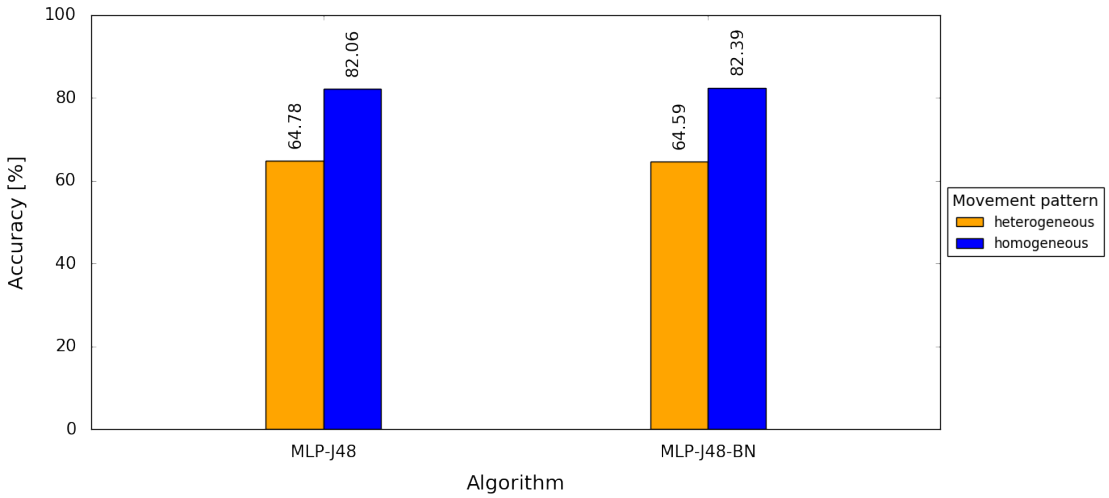

Figure 4.15: Average prediction accuracy using hybrid features and *stacking* for users with many visits

Regarding users with many visits and heterogeneous movement, the average accuracy is nearly 65% when using *stacking* to combine *MLP-J48-BN* or *MLP-J48*. For those users with homogeneous movement, the average accuracy is 82%, as visible in Figure 4.15.

4.5 Execution Time of Single Predictors

In this subsection we present the execution time when using temporal features and hybrid features, different algorithms and for the different user groups. The execution time is composed of the time taken to build the model and the time taken to classify the instances.

4.5.1 Temporal Features

When considering only temporal features, the built model is less complex than the model for hybrid features. Therefore, lower prediction execution time is expected when just using temporal features.

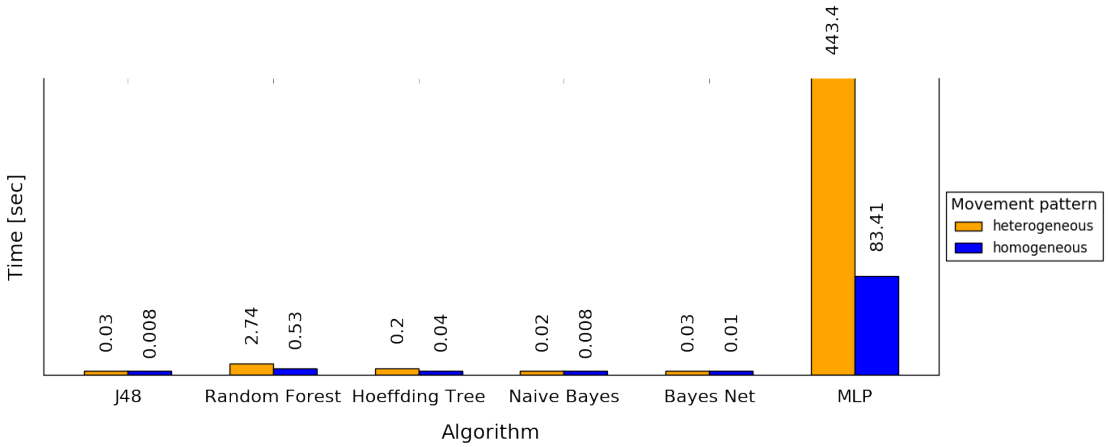

Figure 4.16: Average execution time using temporal features for users with few visits

In Figure 4.16 the average execution time when using only temporal features for users with few visits is presented. We can see that the *MLP* algorithm takes more than 30x more time than all other algorithms. *J48*, *Hoeffding Tree*, *Naive Bayes* and *Bayes Net* all take less than 2 seconds for both the heterogeneous and homogeneous movement group. With 10 seconds for the user group of heterogeneous movement and 6.5 seconds for the users with homogeneous movement, random forest takes significantly more time than the previously mentioned algorithms.

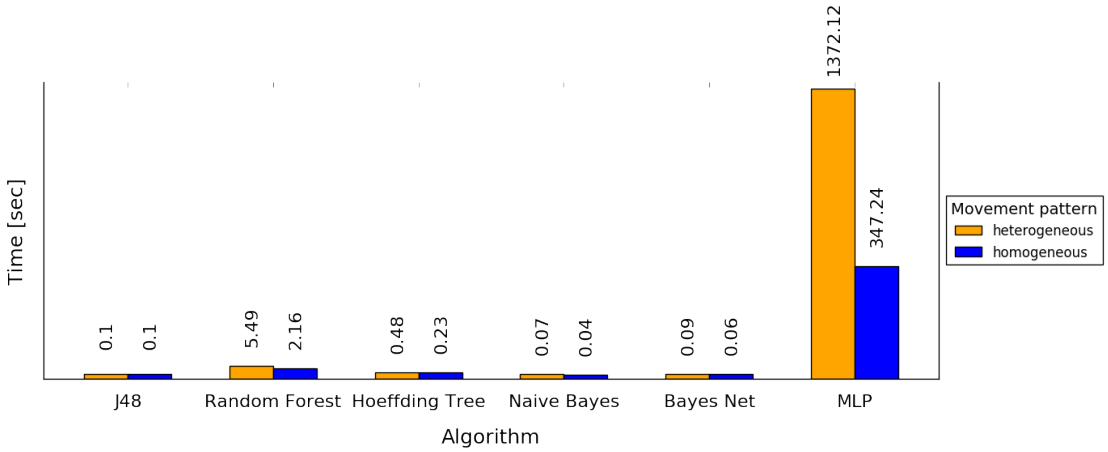

Figure 4.17: Average execution time using temporal features for users with a medium number of visits

The execution time to solve the prediction task with temporal features for the group of users with a medium number of visits is similar compared to the users with few visits: All algorithms except *Random Forest* and *MLP* take less than 1 second. However, the execution time for *Random Forest* and *MLP* increases by a factor of 2 respectively 4, for users with a medium number of visits (see Fig. 4.16 and 4.17).

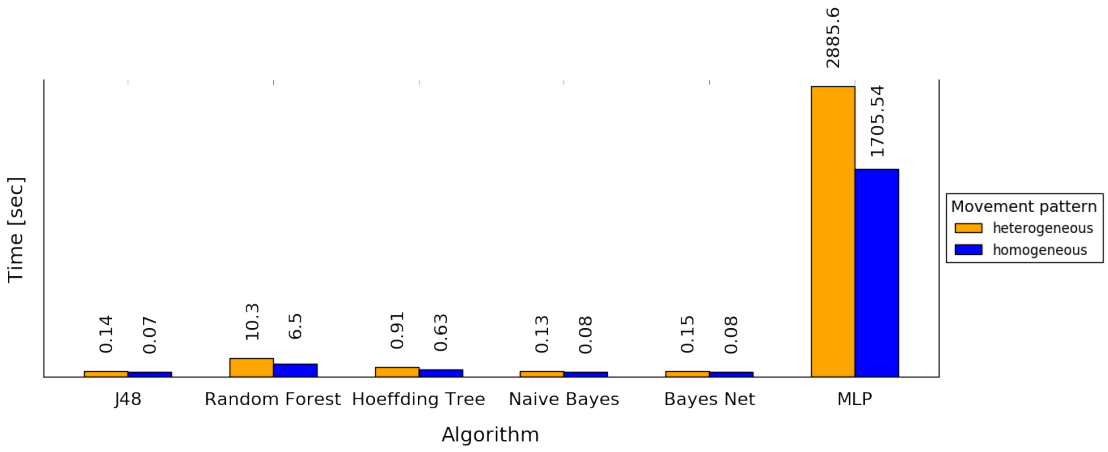

Figure 4.18: Average execution time using temporal features for users with many visits

For the users many visits, all classifiers except *Random Forest* and *MLP* take on average less than 1 seconds to predict the future places of all instances, independent of the movement type, as visible in Figure 4.18. Generally the difference in the execution time between the users with homogeneous and heterogeneous movement is very small

($<$ 4 sec) except for the *MLP*. The execution time takes significantly more time (+1'000 sec) for the heterogeneous movement group compared to the homogeneous one.

4.5.2 Hybrid Features

In the following Figures 4.19, 4.20, 4.21, the prediction execution time using hybrid features with different classifiers is presented.

When considering all types of features the execution time for the prediction only marginally increases compared to when using just temporal features for most of the algorithms except for *MLP*.

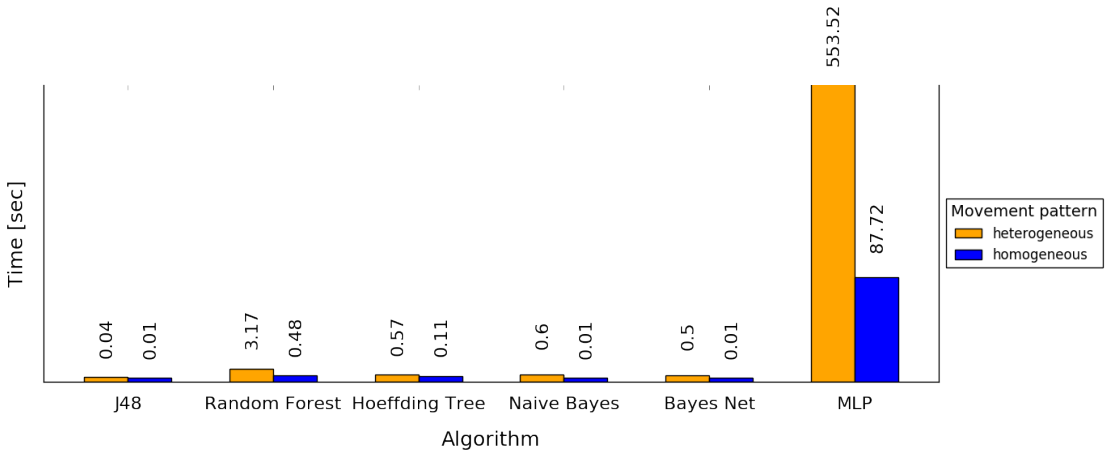

Figure 4.19: Average execution time using hybrid features for users with few visits

Figure 4.19 shows the average prediction execution time for users with few visits. The classifiers *J48*, *Hoeffding Tree*, *Naive Bayes* and Bayes net execute this prediction task on average in less than 4 seconds for both movement types. If the execution time using hybrid features is compared to execution time using just temporal features (such as in Fig. 4.16, it increases substantially only for the heterogeneous movement users when using *MLP* as predictor by roughly 130 seconds.

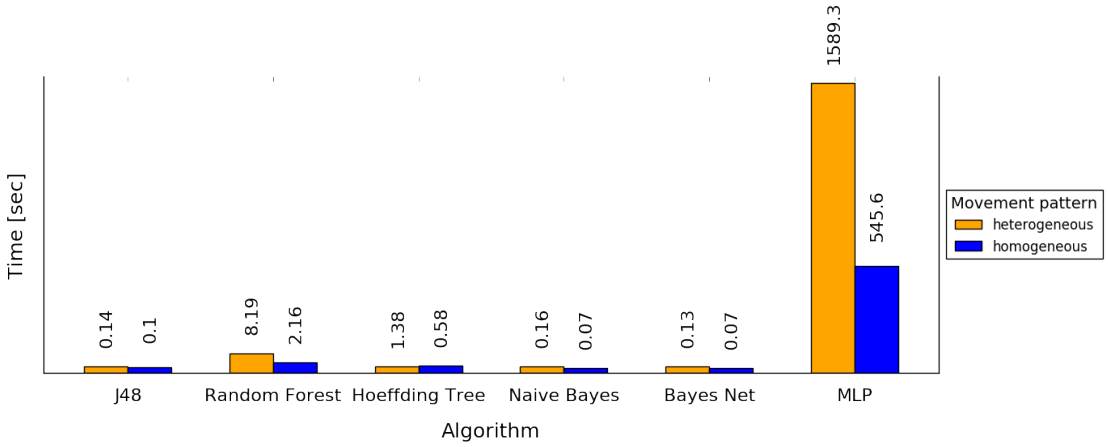

Figure 4.20: Average execution time using hybrid features for users with a medium number of visits

When using hybrid features for the prediction task, the execution time for users with a medium number of visits increases by about 200 seconds for the group of users with homogeneous movement patterns and about 500 seconds for users with heterogeneous movement patterns. The execution time for the other algorithms remains roughly the same.

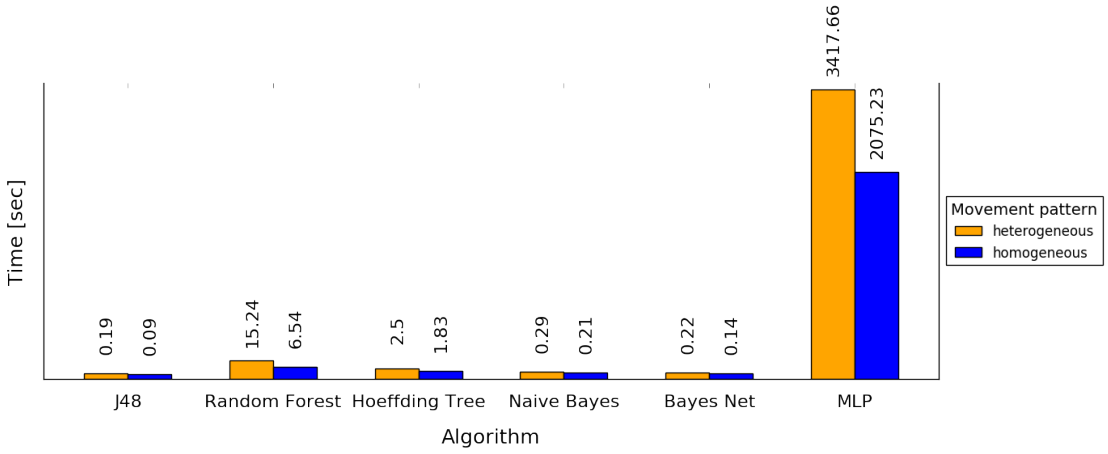

Figure 4.21: Average execution time using hybrid features for users with many visits

As visible in Figure 4.21 the execution time is less than 1 second when using *J48*, *Naive Bayes* or *Bayes Net*. For *Random Forest* the execution takes about 15 seconds for the heterogeneous movement group and on average about 7 seconds for users with homogeneous movement patterns. By far the highest execution time is perceived when using *MLP*, 3'417 vs 2'075 seconds for heterogeneous and homogeneous movement

type. Compared to when just using temporal features, *MLP* using hybrid features takes significantly more time.

4.6 Execution Time of Ensemble Predictors

In this section, we present the time taken to execute the prediction task, when we apply ensemble methods and consider hybrid features.

4.6.1 Hybrid Features

When considering the ensemble learning method *boosting* and users with few visits, the execution time increases by at least a factor of 3. Nevertheless, *J48*, *Hoeffding Tree*, *Naive Bayes* and *Bayes Net* take less than 10 seconds to execute.

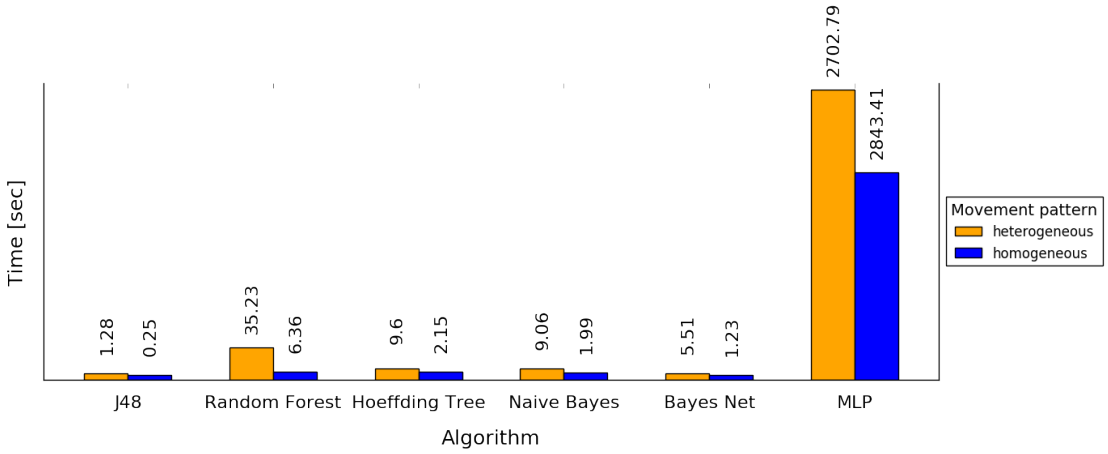

Figure 4.22: Average execution time using hybrid features and *boosting* for users with few visits

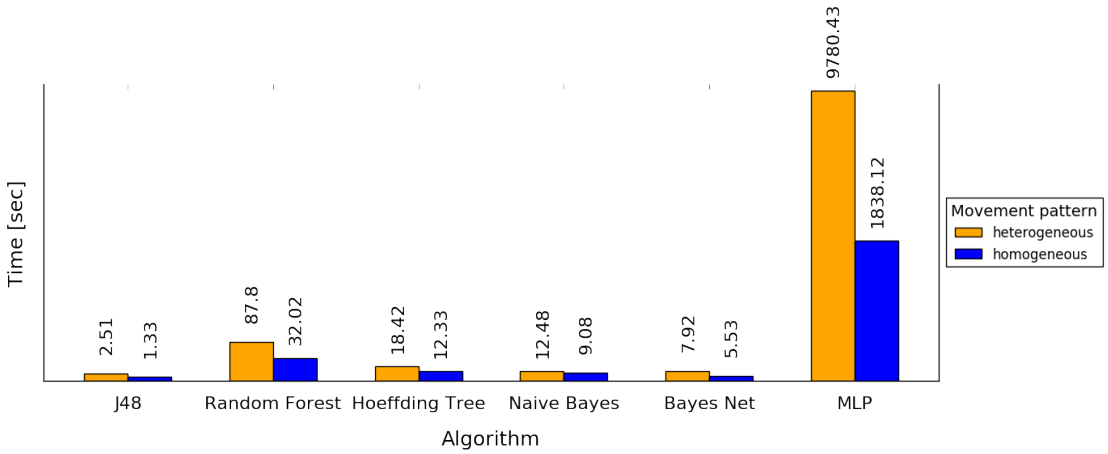

Figure 4.23: Average execution time using hybrid features and *boosting* for users with a medium number of visits

In Figure 4.23 is the execution time presented for users with a medium number of visits. *J48* and the Bayesian classifiers have an execution time of less than 2 seconds. For users with heterogeneous movement patterns, *Random Forest* takes 4x more time and *MLP* 3.5x more time to execute the prediction task compared to execution time for the users with homogeneous movement.

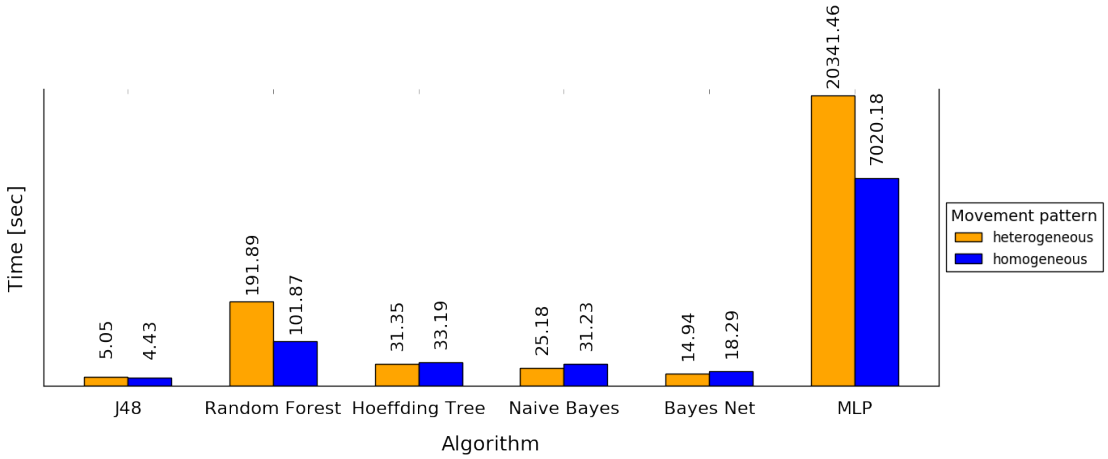

Figure 4.24: Average execution time using hybrid features and *boosting* for users with many visits

For the users with many visits the execution time is only for *J48* less than 10 seconds, regardless of the movement type. *Hoeffding Tree Naive Bayes* and *Bayes Net* perform the task in about 15 to 30 seconds, as visible in Figure 4.24. As seen previously, *MLP* has the highest execution time with 20'341 seconds for heterogeneous and 7'020 seconds for homogeneous users.

Applying the ensemble method *bagging* to *J48*, *Naive Bayes* and *Bayes Net* does only marginally affect the execution time (+2 to seconds) as visible in Figures 4.25,4.26 and 4.27. For the *MLP* the execution time increases by a factor of 10 when *boosting* is used with it, compared to using *MLP* as a single classifier.

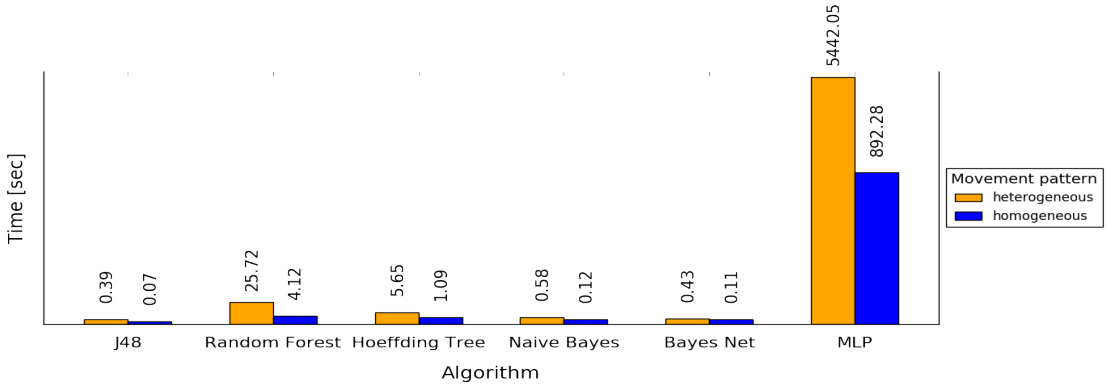

Figure 4.25: Average execution time using hybrid features and bagging for users with few visits

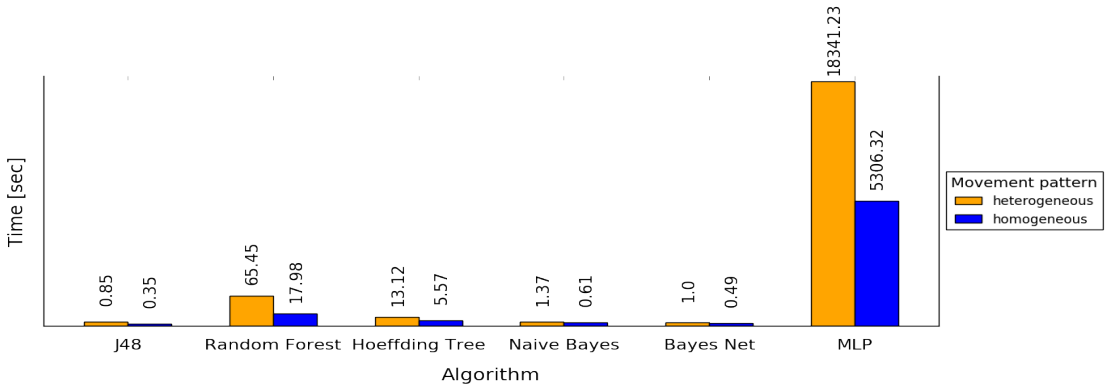

Figure 4.26: Average execution time using hybrid features and bagging for users with a medium number of visits

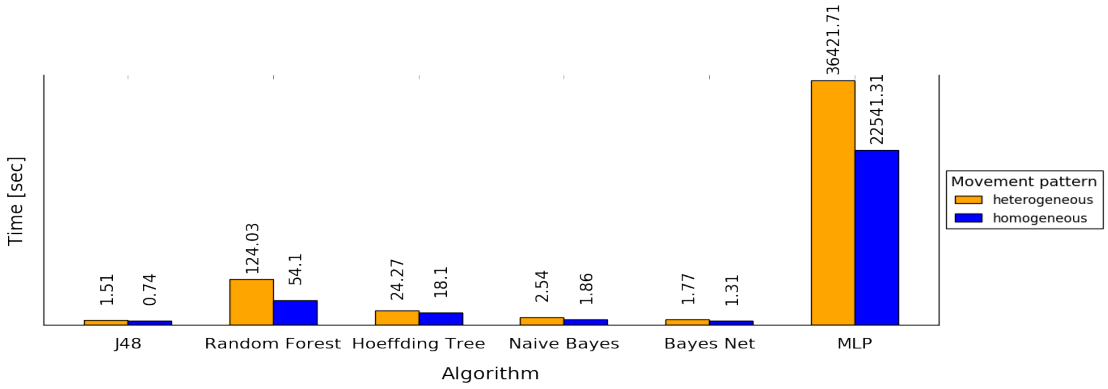

Figure 4.27: Average execution time using hybrid features and bagging for users with many visits

The ensemble method of stacking multiple classifiers takes the highest execution time, compared to all other methods used in this work. This is true for all group of users independent of the number of visits or the movement patterns. When using stacking the execution time, considering users with few visits, is about 6'000 seconds for those with a heterogeneous movement and about 1'000 seconds for users with homogeneous movements.

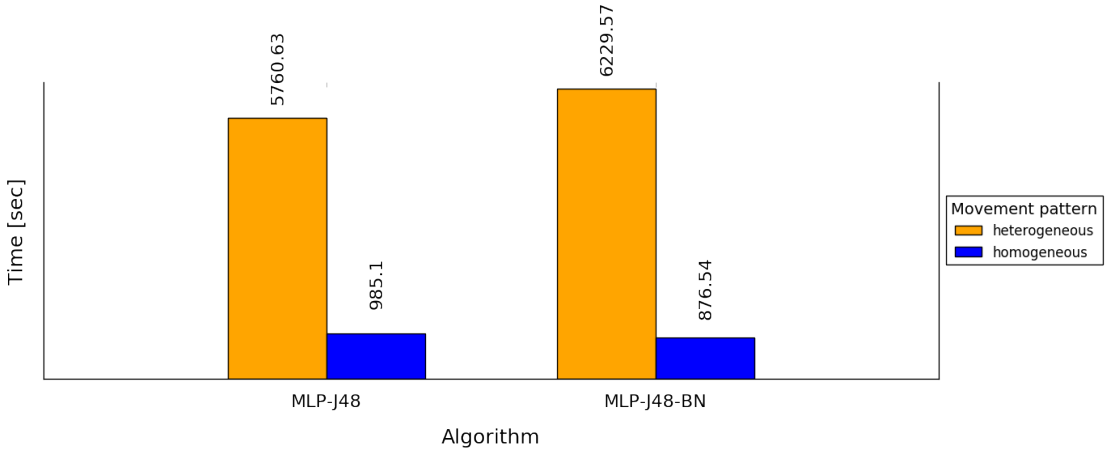

Figure 4.28: Average execution time using hybrid features and *stacking* for users with few visits

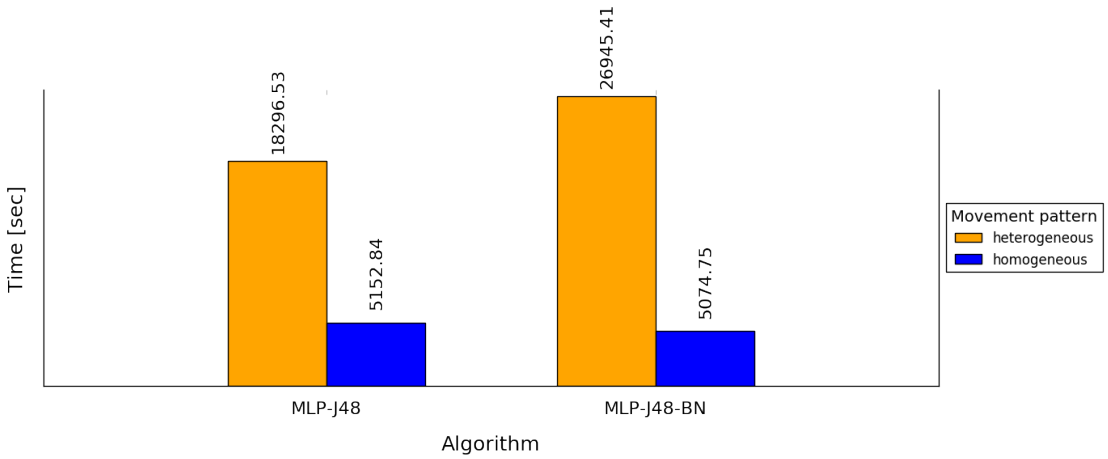

Figure 4.29: Average execution time using hybrid features and *stacking* for users with a medium number of visits

Regarding users with a medium number of visits, the stacked algorithms in Figure 4.29 require about 5'000 seconds to execute the prediction task for users with homoge-

neous movement patterns. The execution time of the stacked algorithms is higher than 18'000 seconds for users with heterogeneous movement patterns.

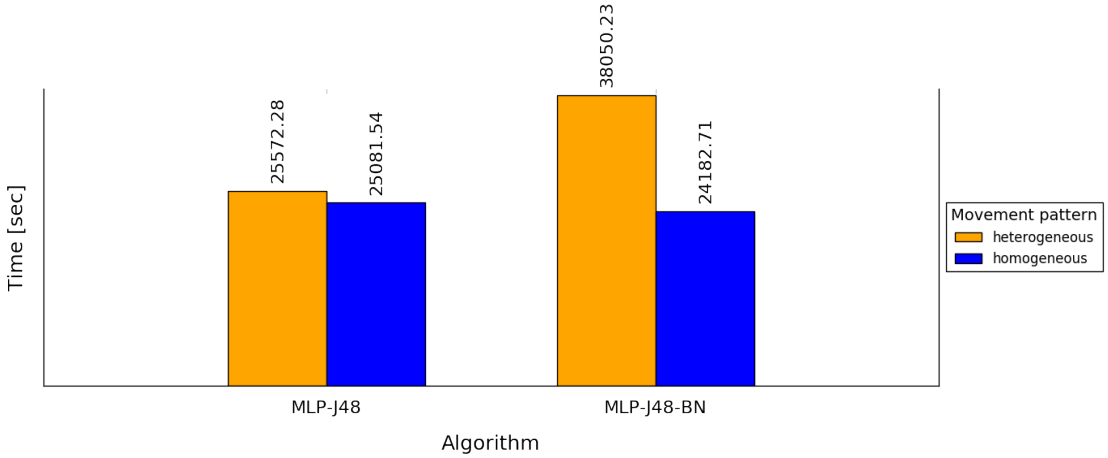

Figure 4.30: Average execution time using hybrid features and *stacking* for users with many visits

For the users with many visits and heterogeneous movements the average execution time is at most 38'050 seconds when using stacking. The execution time for users with homogeneous movements is about 25'000 seconds when using stacking.

4.7 Performance Analysis

As seen before, hybrid features can increase the average accuracy by up to 30%. To explain the improvement in the higher accuracy when using hybrid features - and not only temporal features for the prediction task - we present the confusion matrix. The confusion matrix is a tool to visualize the performance of an algorithm. In Figure 4.31 the confusion matrix is presented when using temporal features for a user with many visits and homogeneous movements. On the vertical axis of the confusion matrix are the actual place-IDs presented. The horizontal axis represents the predicted place-ID. This view helps understand which places are mistaken for another place. In Figure 4.31 it is visible, that roughly 6 different places are identified out of a total of 107. For example, 181 instances with an actual place-ID of 1 are correctly predicted as place-ID 1. Yet, there are a lot of misclassifications: For example, 34 instances with an actual place-ID of 1 were misclassified as placed-ID 2. Temporal features do not provide enough information to identify or distinguish more places.

```
  1   2   3   4   5   6   7   8   9  10  <-- classified as
----------------------------------------
181  34  21  19  71   0   0   1   0   2  |   1
 23 104  40   6  15   0   0   0   0   9  |   2
 21  35  59   6  14   0   0   1   0   5  |   3
 44  43   7  32  48   0   0   0   0   0  |   4
110  34  16  28  99   0   0   0   0   2  |   5
  2   1   0   0   4   0   0   0   0   0  |   6
  3   6   2   0  16   0   0   0   0   0  |   7
 23   5   2   1  12   0   0   0   0   0  |   8
  2   0   0   3   1   0   0   0   0   0  |   9
  8  29  10   2  17   0   0   0   0   4  |  10
```

Figure 4.31: Confusion matrix when using temporal features and *J48* as classifier

```
  1   2   3   4   5   6   7   8   9  10  <-- classified as
----------------------------------------
255   0  29   5  63   0   0   0   0   0  |   1
  0 202   0   0   0   0   0   0   0   0  |   2
  0   0 133   0   5   0   0   0   0   0  |   3
 92   0   2  20  64   0   0   0   0   0  |   4
152   0  17   8 119   0   0   0   0   0  |   5
  0   0   0   0   0   0   0   0   0   2  |   6
  0   0   0   0   0   0  10   0   0   5  |   7
 41   0   1   0   1   0   0   0   0   0  |   8
  0   6   0   0   0   0   0   0   0   0  |   9
  0   0   0   0   0   0   0   0   0  74  |  10
```

Figure 4.32: Confusion matrix when using temporal features, bluethooth, profile and charging with *J48* as classifier

```
  1   2   3   4   5   6   7   8   9  10  <-- classified as
----------------------------------------
354   0   0   0   0   0   0   0   0   0  |   1
  0 202   0   0   0   0   0   0   0   0  |   2
  0   0 144   0   0   0   0   0   0   0  |   3
  0   0   0 178   0   0   0   0   0   0  |   4
  0   0   0   0 298   0   0   0   0   0  |   5
  0   0   0   0   0   7   0   0   0   0  |   6
  0   0   0   0   0   0  19   0   0   0  |   7
  0   0   0   0   0   0   0  42   0   0  |   8
  0   0   0   0   0   0   0   0   4   0  |   9
  0   0   0   0   0   0   0   0   0  76  |  10
```

Figure 4.33: Confusion matrix when using hybrid features and *J48* as classifier

Therefore, we added more features and of other types. In Figure 4.32 the confusion matrix is displayed when using temporal features + bluethooth + charging and profile feature. With this feature set, the model no longer mistakes placed-ID 1 with place-ID 2 and vice versa. Place-ID 1 is most probably the user's home. The phone-profile is most of the time set on *general* and the number of visible bluethooth devices is small. When the user visits the place with ID 2 is phone profile is set to silent or general and the number of detected bluethooth devices is higher than at place-ID 1. Consequently, place-ID 2 is most probably the workplace of the user. Figure 4.33 shows the confusion matrix when using all in section 3.2 defined features. The model is able to properly discriminate at least 10 different places.

In this work, the ensemble learning method could not significantly improve the prediction accuracy. In general, ensemble methods are beneficial to the accuracy if there is significant diversity between the different models [24]. A seen in chapter 4 there is very little variance in the average prediction accuracy between the different algorithms, when using hybrid features ($\pm 2\%$). Furthermore, there is low variance in prediction accuracy when comparing different algorithms user by user. The features selected for the prediction task are of good quality even for the users with a low number of visits. Therefore, there was no possibility for ensemble methods to compensate for sparsity of data. Finally, WLAN, GSM, duration, leavingtime and Bluethooth data turned out to be good and reliable features to correctly predict future places.

4.8 Performance Comparison

In this subsection, we compare our results with those of previously conducted experiments.

Work	Algorithm	Features	Movement Pattern	Accuracy
Our work	J48	Hybrid	Homogeneous	83.98%
	Bagging-MLP	Hybrid	Heterogeneous	65.02%
Tran *et al.* [2]	J48 with Home and	Temporal	Homogeneous	71.85%
	Holiday Detection	Temporal	Heterogeneous	54.27%
Zhao *et al.* [25]	Stacking	Hybrid	Homogeneous	83.37%

Table 4.1: Accuracy comparison of different works

For users with homogeneous movement patterns, the prediction accuracy could not be improved and is with almost 84% as good as the work of Zhao *et al.* [25]. All users have places in their data set, which they have only visited once. The prediction of such places is rather difficult. Consequently, the obtained accuracy is quite high and difficult to improve, as one had to deal with the correct prediction of such places. Regarding users with heterogeneous movement patterns, our proposed features improved the overall accuracy by roughly 10% compared to the work of *Tran et al.* [2]. In their work, Tran *et al.* use temporal features such as leaving time, duration and weekday/weekend, and an algorithm to detect if the user is on holiday. Based on our results, we suggest to exploit hybrid features for the prediction task to improve the accuracy for both, users with homogeneous and users with heterogeneous movement patterns.

5
Conclusions

In this work, we model the future place prediction as a standard supervised learning task. The mobile-phone users are divided into groups based on the number of total visits and whether their movement pattern is homogeneous or heterogeneous. We use a rich selection of features to distinguish places of a user. Then, we use various algorithms to predict future locations of a user. We observe that using hybrid features increases the accuracy by up to 22% compared to temporal features, for users with few or a medium number of visits, having heterogeneous or homogeneous movement patterns. For the users with many visits, using hybrid features can increase the accuracy about 27%, for users with heterogeneous movement patterns and almost 30% for users with homogeneous movement patterns compared to using only temporal features. For users with homogeneous movement patterns, future places can be predicted with a high accuracy ($>$83%) and reasonably fast. The correct prediction of future places for users with heterogeneous movement patterns is still difficult.

Bibliography

[1] Weka 3: Data mining software in java. https://www.cs.waikato.ac.nz/ml/weka/. Accessed: 2018-01-16.

[2] Le-Hung Tran, Michele Catasta, Luke McDowell, and Karl Aberer. Next place prediction using mobile data. 2012.

[3] Vincent Etter, Mohamed Kafsi, Ehsan Kazemi, Matthias Grossglauser, and Patrick Thiran. Where to go from here? mobility prediction from instantaneous information. *Pervasive Mob. Comput.*, 9(6):784–797, December 2013. ISSN 1574-1192. doi: 10.1016/j.pmcj.2013.07.006. URL http://:dx.doi.org/10.1016/j.pmcj.2013.07.006.

[4] Huiji Gao, Jiliang Tang, and Huan Liu. Mobile location prediction in spatio-temporal context. In *Nokia mobile data challenge workshop*, volume 41, pages 1–4, 2012.

[5] Ying Zhu, Yong Sun, and Yu Wang. Nokia mobile data challenge: Predicting semantic place and next place via mobile data. 01 2012.

[6] Jingjing Wang and Bhaskar Prabhala. Periodicity based next place prediction.

[7] Daniel Ashbrook and Thad Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, 7(5):275–286, October 2003. ISSN 1617-4909. doi: 10.1007/s00779-003-0240-0. URL http://dx.doi.org/10.1007/s00779-003-0240-0.

[8] Haiyang He, Yuanyuan Qiao, Sheng Gao, Jie Yang, and Jun Guo. Prediction of user mobility pattern on a network traffic analysis platform. In *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture*, MobiArch '15, pages 39–44, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3695-6. doi: 10.1145/2795381.2795389. URL http://doi.acm.org/10.1145/2795381.2795389.

[9] Trinh Minh Tri Do, Olivier Dousse, Markus Miettinen, and Daniel Gatica-Perez. A probabilistic kernel method for human mobility prediction with smartphones.

Pervasive Mob. Comput., 20(C):13–28, July 2015. ISSN 1574-1192. doi: 10.1016/j.pmcj.2014.09.001. URL https://doi.org/10.1016/j.pmcj.2014.09.001.

[10] M. Karimzadeh, Z. Zhao, L. Hendriks, R. de O. Schmidt, S. la Fleur, H. van den Berg, A. Pras, T. Braun, and M. J. Corici. Mobility and bandwidth prediction as a service in virtualized lte systems. In *2015 IEEE 4th International Conference on Cloud Networking (CloudNet)*, pages 132–138, Oct 2015. doi: 10.1109/CloudNet.2015.7335295.

[11] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das. Understanding traffic dynamics in cellular data networks. In *2011 Proceedings IEEE INFOCOM*, pages 882–890, April 2011. doi: 10.1109/INFCOM.2011.5935313.

[12] Marta C. Gonzalez, Cesar Hidalgo, and Albert-Laszlo Barabasi. Understanding individual human mobility patterns. 453:779–82, 07 2008.

[13] Wikipedia: Haversine formula. https://en.wikipedia.org/wiki/Haversine_formula. Accessed: 2018-01-16.

[14] Weka class infogainattributeeval. http://weka.sourceforge.net/doc.stable-3-8/weka/attributeSelection/InfoGainAttributeEval.html. Accessed: 2018-01-16.

[15] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986. ISSN 0885-6125. doi: 10.1023/A:1022643204877. URL http://dx.doi.org/10.1023/A:1022643204877.

[16] Wikipedia: Bayes' theorem. https://en.wikipedia.org/wiki/Bayes%27_theorem. Accessed: 2018-01-16.

[17] Vikramkumar, Vijaykumar B, and Trilochan. Bayes and naive bayes classifier. *CoRR*, abs/1404.0933, 2014. URL http://arxiv.org/abs/1404.0933.

[18] Wikipedia: Artificial neural network. https://en.wikipedia.org/wiki/Artificial_neural_network. Accessed: 2018-01-16.

[19] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6(3):21–45, Third 2006. ISSN 1531-636X. doi: 10.1109/MCAS.2006.1688199.

[20] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012. ISBN 1439830037, 9781439830031.

[21] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, EuroCOLT '95, pages 23–37, London, UK, UK, 1995. Springer-Verlag. ISBN 3-540-59119-2. URL http://dl.acm.org/citation.cfm?id=646943.712093.

[22] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, Aug 1996. ISSN 1573-0565. doi: 10.1023/A:1018054314350. URL https://doi.org/10.1023/A:1018054314350.

[23] Stacked ensembles. http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/stacked-ensembles.html. Accessed: 2018-01-16.

[24] Cross validated: How do ensemble methods outperform all their constituents? https://stats.stackexchange.com/questions/255230/how-do-ensemble-methods-outperform-all-their-constituents. Accessed: 2018-01-16.

[25] Zhongliang Zhao, Mostafa Karimdazeh Motallebiazar, and Torsten Braun. Technical report: Mobile crowd location prediction with hybrid features using ensemble learning. 2017.