Journal of Internet Services and Applications

RESEARCH Open Access

Ensemble mobility predictor based on random forest and Markovian property using LBSN data

Felipe Araújo[1*], Fábio Araújo[1], Kássio Machado[2], Denis Rosário[1], Eduardo Cerqueira[1] and Leandro A. Villas[3]

Abstract
The ubiquitous connectivity of Location-Based Systems (LBS) allows people to share individual location-related data anytime. In this sense, Location-Based Social Networks (LBSN) provides valuable information to be available in large-scale and low-cost fashion via traditional data collection methods. Moreover, this data contains spatial, temporal, and social features of user activity, enabling a system to predict user mobility. In this sense, mobility prediction plays crucial roles in urban planning, traffic forecasting, advertising, and recommendations, and has thus attracted lots of attention in the past decade. In this article, we introduce the Ensemble Random Forest-Markov (ERFM) mobility prediction model, a two-layer ensemble learner approach, in which the base learners are also ensemble learning models. In the inner layer, ERFM considers the Markovian property (memoryless) to build trajectories of different lengths, and the Random Forest algorithm to predict the user's next location for each trajectory set. In the outer layer, the outputs from the first layer are aggregated based on the classification performance of each weak learner. The experimental results on the real user trajectory dataset highlight a higher accuracy and f1-score of ERFM compared to five state-of-the-art predictors.

Keywords: Mobility prediction, Ensemble learning, Location-based social network

1 Introduction

Over the past decade, an overwhelming number of location-aware services and applications have profoundly changed the way people live [1]. The ubiquitous connectivity of Location-Based Systems (LBS) allows people to share individual location-related data anytime [2]. In this sense, Location-Based Social Networks (LBSN) [3], such as Foursquare and Instagram, became popular to provide public data capable of mapping people through status, check-ins, and photos shared online, leading to a new urban computing era [4]. The availability of massive human location tracking datasets was enabled by mobile technologies, such as big-data technologies in mobile telecommunication networks and large scale deployment of GPS technologies. In this context, LBSNs users stopped being only consumers to become data producers, offering various research opportunities, such as mobility prediction and recommendation systems [5]. Hence, location data bridges the gap between the physical and digital worlds, enabling a deeper understanding of users' preferences and behavior.

LBSN data distinguish from traditional GPS data and Call Data Records (CDR) mainly in social, spatial, and temporal resolutions, which can be used to model movement patterns and infer similar movements [6]. LBSN provides valuable information that is currently available in large-scale and low-cost fashion via any traditional data collection methods [7]. In this sense, social media is an important tool in urban computing to provide urban data with social features, such as the user's preferences and

*Correspondence: felipearaujo@ufpa.br
[1]Federal University of Pará, Rua Augusto Corrêa, 01, 66075-110 Belém, Pará, Brazil
Full list of author information is available at the end of the article

routine. It allows us to understand user patterns, city dynamics, and social, economic, and cultural aspects [8]. For instance, LBSN data can be used for extracting user mobility patterns to understand when and where a user commonly goes (location prediction). Also, it can capture user preferences and location profiles to investigate where and when a user wants to explore (location recommendation).

In this context, mobility prediction plays important roles in urban planning, traffic forecasting, advertising, and recommendations, and has thus attracted lots of attention in the past decade [9]. For example, it can be used to improve Device-to-Device (D2D) communications in Opportunistic Networks, where user location is required to make mobile data offloading. Besides, mobility prediction can be applied to proactive caching, alleviating back-haul traffic, and mitigating latency caused by handovers [10]. On the other hand, due to cold start and sparsity problems, LBSN imposes some challenges when predicting user mobility, requiring more complex data mining techniques compared to other mobility data, such as GPS and CDR.

The study of human movement patterns shows that people's actions are repetitive since they visit specific locations at a relatively fixed time every day [11]. Also, people tend to visit the same places that their friends visited, enabling the investigation of social features [12]. In contrast, social information is less effective in predicting a user's repetitive mobility behavior compared to spatial and temporal information, since a user's repetitive mobility behavior is more affected by his interests than his friends' preferences [6]. On the other hand, social correlation, called user similarity, can be considered to assist spatial and temporal information for mobility prediction. For instance, a user's trajectory can be in some way correlated with the trajectory of other users. Therefore, mobility prediction can be made considering the combination of all possible locations of both users.

Several methods have been proposed for mobility prediction based on mobility data, where most of them use the historical trajectories to identify user and group movement patterns [13]. For instance, Markov models are widely used in prediction algorithms, due to their efficiency, simplicity, and low computing costs. For example, a Markov Chain (MC) predictor considers the sequence of last locations visited by a user to predict his next location. The length k of that sequence of locations represents the order of the Markov chain, and we refer to this model as an order-k Markov Chain model. In this sense, the model assumes that the prediction is based on the location transitions, computing the number of times the user moved from a location to another. For instance, in the order-1 MC (1-MC), in which sequence length $k = 1$, the next location of a given user is only influenced by his last visited location. In contrast, in the order-2 MC (2-MC), the next location is not only dependent on the last visited location but also on the previous one.

Due to the advance of technologies and the big challenges faced by mobility prediction problems, several Machine Learning (ML) methods have been used to predict the user's next location [14–16]. In this context, aggregated models, also known as Ensemble predictors, have become popular as long as they have shown excellent results. For instance, suppose you pose a complex question to thousands of random people, then aggregate their answers. In many cases, you will find that the aggregated answer is better than an expert's answer. Similarly, if one aggregates the predictions of a group of predictors, he will get, in most cases, better predictions than with the best individual predictor [17]. Hence, a combination of several models tends to improve mobility prediction.

In this article, we extend the previous work [18] by introducing an Ensemble Random Forest-Markov predictor, called ERFM. In [18], we propose the TEmporal Markov Mobility predictor based on User Similarity (TEMMUS), a Markov-Chain mobility predictor which leverages the days of the week and the user similarity to predict the next location. On the other hand, ERFM is a two-layer ensemble predictor and ranks a set of possible locations that a given user could be by combining Random Forest (RF) models based on the trajectory of different lengths. Based on the Markov property, also known as memoryless property, it assumes that the user's next location depends only on the last locations he visited. Moreover, ERFM considers the locations coordinates (latitude and longitude), the bearing angles, and the distances between the locations for each trajectory to predict the next location. Besides, we also introduced an extended evaluation in a more challenging and realistic user mobility scenario, consisting of more than 400 thousand users' records over a period of 22 months.

The contributions of this work can be summarized as follows:

i) Two-layer Ensemble mobility predictor using the combination of Random Forest models based on sequences of locations of different lengths.
ii) Extended evaluation in a more realistic scenario, new metrics, and evaluated models.

The remainder of this article is organized as follows. In Section 2, we review relevant related work about mobility prediction. In Section 3, we introduce the proposed ERFM mobility predictor and describe our data collection procedures and evaluation metrics. In Section 4, we describe the results. In Section 5, we introduce the conclusion, limitations, and future work of this article.

2 Related work

To identify the important locations of the target user from trajectory data, Wang et al. [19] proposed a novel division method for pre-processing trajectory data. Also, to predict the next location of mobile users, the authors proposed a multi-order fusion Markov model based on the Adaboost algorithm. The model order k is adaptively determined, and the weight coefficients of the 1-to-k order models are given by the Adaboost algorithm according to the importance of various order models. As a result, a multi-order fusion Markov model is generated to predict the next important location of the user. According to the authors, experimental results on the real user trajectory data set Geolife, prove that the proposed method overcomes the prediction accuracy of the low-order Markov model and the high sparse rate of the high-order Markov model to some extent, and makes full use of the user's prefix trajectory information.

Gebrie et al. [16] performed a comparative analysis of four mobility predictors: Deep Neural Network, Extreme Gradient Boosting Trees, Semi-Markov, and Support Vector Machines (SVM). The authors evaluated the effectiveness of each model not only based on the model's ability to predict the future location of mobile users but also the time each algorithm takes to be fully trained and perform such prediction. Their investigation was based on a realistic synthetic dataset of eighty-four mobile users generated through a realistic Self-similar Least Action Walk (SLAW) mobility model. Their experimental results prove the Extreme Gradient Boost Trees algorithm stands out as a clear winner among all predictors considered. Besides, its high prediction accuracy enables high energy saving gain of above 80% when it is employed for driving proactive energy Self-Organizing Networks solution.

Abani et al. [10] proposed a proactive caching strategy for reducing the latency of retrieving predictable content requests in a vehicular network. This proposal considers the individual strategy for mobility prediction since it is based on the history of the object itself. However, this approach is limited by the locations visited by the node, failing in predicting future locations of non-systematic objects due to the individuality of each object. Nguyen et al. [20] proposed a prediction-based routing algorithm, which considers both spatial and temporal contact dimensions. In this sense, the source knows when and where to start the routing process, which minimizes the network delay and overhead.

Existing mobility prediction models consider the historical record of users. For instance, Jiang et al. [21] proposed a method to extract the Region-of-Interest (ROI) from the historical data location. On the other hand, other authors consider not only the history of the user but also the spatial-temporal context to improve the accuracy of the mobility prediction model. For instance, Wang et al. [1] modeled the spatial and temporal activity preferences separately and combined them for preference inference. Gao et al. [22] proposed a general framework to exploit and model temporal cyclic patterns and their relationships with spatial and social data. The experimental results on two real-world LBSN data-sets that validate the importance of temporal effects in capturing user mobile behavior.

Some researchers study the social property on LBSNs to extract user movement and preference patterns. Cheng et al. [23] included the social information, and combined the geographical influence into a generalized matrix factorization framework to provide more accurate and efficient Points Of Interests (POI) recommendation. Silveira et al. [12] proposed a model to predict human mobility, called MobDatU, which considers data from mobile calls and LBSN data. MobDataU includes social interactions between users as an important factor to predict the next region. Munjal et al. [24] proposed SMOOTH, which is a simple and realistic model that leverages several known features of human movement to model human mobility. Dong et al. [25] introduced Leap Graph, which considers base station location information available in a CDR to a service provider to perform mobility prediction.

Markov Model is one of the statistical models used in predictive analytics. In this way, Chen et al. [13] introduced three Markov-based models, namely, Personal Markov Model (PMM), General Markov Model (GMM), and Next Location Predictor with Markov Modeling (NLPMM). PMM considers only the mobility of a specific user, i.e., its past trajectories, to build the mobility prediction model. On the other hand, GMM takes into account the collective aspects of the mobility, i.e., considering not only the movement of a specific node but of all the nodes since they often share similar movement patterns. NLPMM combines PMM and GMM models using linear regression to explore the individual and collective aspects of mobility.

In our previous work [18], we proposed TEMMUS, a Markov-Chain-based mobility predictor. It considers the day of the week (weekday or weekend) and the user similarity to enhance the user's next location prediction. Moreover, TEMMUS considers a Fallback-Markov approach, in which the order of the trajectory is reduced if the user transition does not exist in the Markov-Chain (e.g, the user has never made this trajectory). It is important to notice that while ERFM and TEMMUS are based on the Markov property (memoryless property), the former is not a Markov-Chain model. Table 1 summarizes the analyzed mobility predictors.

3 Ensemble random forest-Markov

In this section, we provide the mobility problem formalization, the main concepts related to it, and introduce

Table 1 Summary of Mobility Prediction Models

	Characteristics							
	Factors			Data Type		Collected/Generated From		
	Temporal	Spatial	Social	Real	Synthetic	Location-Based Social Network	Call Data Record	Other
Wang et al. [19]		✓		✓				✓
Gebrie et al. [16]	✓	✓			✓	✓		
Abani et al. [10]		✓		✓				✓
Nguyen et al. [20]	✓	✓			✓			✓
Jiang et al. [21]	✓	✓		✓				✓
Wang et al. [1]	✓	✓		✓		✓		
Gao et al. [22]	✓	✓	✓	✓		✓		
Cheng et al. [23]		✓	✓	✓		✓		
Silveira et al. [12]		✓	✓	✓		✓	✓	
Munjal et al. [24]		✓			✓			✓
Dong et al. [25]		✓			✓		✓	
Chen et al. [13]		✓	✓	✓		✓		
Felipe et al. [18]	✓	✓	✓	✓		✓		
ERFM	✓	✓	✓	✓		✓		

the Ensemble Random Forest-Markov predictor (ERFM). It mainly consists of the following steps, each of them is detailed in the following subsections.

1 **Data Acquisition/Preparation:** In the first step, we collect data and split it into two subsets: train and test.
2 **Features Engineering:** In the second step, we build trajectories of different lengths based on the model order k. Also, extract features from the data, such as bearing and Haversine distance between every two locations.
3 **Model Building/Training/Aggregation:** It is the main. Here, we build the base models for the ensemble predictor, tune the hyperparameters for each model based on the Grid Search approach, train each model using the selected parameters, and aggregate them based on the Out-Of-Bag error.
4 **Model Evaluation:** In the last step, we evaluate the ensemble predictor (ERFM).

3.1 System model

In this article, we introduce an ensemble model based on LBSN data to predict the user's next location. Therefore, for a better understanding, we provide a brief definition of the principal concepts related to it, including the mobility prediction problem formalization.

Definition 1 (check-ins) *The check-in is defined as a 5-tuple* $c = \{id, lat, lon, loc, t\}$, *where 'id' represents the user id; 'lat' and 'lng' denotes the location coordinates and is defined by latitude and longitude, respectively, 'loc' is the location id and 't' represents the timestamp. We denote the set of check-ins of all users as* $\mathbb{C}$ *and the set of check-ins for a specific user as* $\mathbb{C}_{id}$, *where the index is the user id. For instance,* $\mathbb{C}_i$ *is the check-ins set for the user i.*

Definition 2 (trajectory) *The trajectory* $tr_m(i)$ *is defined as the m-th time-ordered sequence of locations that the user 'i' just passed. For instance, for a sequence of length three* ($k = 3$), $tr_1(u) = \{loc_1, loc_2, loc_3\}$ *is the frist trajectory of user 'u', indicating he just checked-in at these locations in such order. The set of all trajectories of all users is defined as* $\mathbb{T}$ *while the set of all trajectories of a specific user is defined as* $\mathbb{T}_{id}$, *where the index is the user id.*

Definition 3 (mobility prediction) *We formalize the mobility prediction problem as follows. Given a user u whose current check-in is* $c = \{u, lat, lon, t\}$, *we aim to rank the set of possible locations so that the next location to be visited will be ranked at the highest possible position in the list. Therefore, the mobility prediction problem is essentially a ranking task, where we compute a ranking score for all venues in* $\mathbb{L}$.

3.2 Data acquisition/preparation

We used the United States region from Global-scale Check-in Dataset [26]. It has over 12 million check-ins by about 400 thousand users at about 2 million locations over a period of 22 months (from Apr. 2012 to Jan. 2014). This dataset consists of the following fields: (i) User ID (anonymized); (ii) Latitude; (iii) Longitude; (iv) Timestamp/DateTime; (v) Location ID; (vi) category. Even though this dataset has a high number of users, only a few ($< 1\%$) was used. It occurs due to the number of check-ins per location or the total number of check-ins per user. In this sense, we considered only users that checked-in at least 10 different locations and 5 times on each. Also, we filtered users with a total of check-ins of less than 500.

Figure 1 illustrates ERFM pipeline, in which the first process is the data splitting. There are many ways to split the data into training and testing sets. The most common approach is to use some version of random sampling since it is a straightforward strategy to implement and usually protects the process from being biased towards any characteristic of the data. However, this approach can be problematic when the response is not evenly distributed across the outcome. In this context, a less risky splitting strategy would be to use a stratified random sample based on the outcome. Therefore, for classification models, this is accomplished by randomly selecting samples within each class. It ensures that the frequency distribution of the outcome is approximately equal within the training and test sets.

Also, the data can be sliced sequentially, in which the first p% data is the training set and the remainder data is the testing set. However, sequential data such as mobility trajectories is subjected to auto-correlation, where the assumption made by the currently splitting approaches of i.i.d observations does not hold. Therefore, techniques such as random sampling are not applied to time series data, since they do not consider its main aspect: time. Moreover, for large datasets, such as Global-scale Check-in, splitting the whole data sequentially is not a good option, since the testing set may not be correlated with the training set. In this sense, ERFM is based on the Block-rolling Time Series split (BRTS). It leverages the time dependence by splitting the data into N small partitions (folds), and for each one, it applies a sequential split given a training and testing data proportion (see Fig. 1, item 1).

3.3 Features engineering

In many cases, the assumption that "the next place that is going to be visited is only dependent on the current location" becomes unsuitable or even false because it can be not enough to extract the patterns. For instance, the mobility pattern may be associated with several consecutive user movements than low-order transitions. On the other hand, building higher-order transitions may lead to long trajectories that are not directly related to the user's next location and a reduced number of samples, making the mobility prediction difficult. In this context, we used a varied-order approach, where for a defined model order k, we build trajectories ranging from size 1 to k. For instance, for a $k = 5$, we also build trajectories of sizes from 1 to 4, totaling trajectories of different sizes, each responsible for extracting a different pattern.

In this sense, the user trajectories were built based on two aspects: *(i) Individual* and *(ii) General*. The former assumes that user mobility is only influenced by his behavior while the general aspect assumes that behaviors of different users can be someway correlated. Firstly, we cluster the sequence of locations according to the day of the week. Then, for each cluster, we group the check-ins based on the timestamp difference between two consecutive check-ins from the same user. If it is lower or equal than a threshold β, we just add to the same group, otherwise, we create a new one. After that, assuming the memoryless property and the maximum trajectory length k, we iterate the groups up to k times using an overlapping rolling window with variable size (from 2 to $k+1$). It is important to notice that the rolling window length is fixed for each iteration. As a result, we split each group into other overlapping subgroups of size from 2 to $k+1$, where the first locations are the trajectory and the last location is the destination.

In the context of general aspect, it is also categorized into other two classes: *(i) Collective* and *(ii) Hybrid*. In the collective approach, all the individual trajectories set are merged into unique collective trajectories set. Hence, it assumes that the trajectories are the same for all users. For instance, let $\mathbb{T}_i$ and $\mathbb{T}_j$ be the individual trajectories set for the users i and j, the collective trajectories set is given by $\mathbb{T} = \mathbb{T}_i \cup \mathbb{T}_j$. The main advantage of this approach compared to the individual one is the number of possible next locations. For instance, Markov-based algorithms fail to correctly predict future movements if the new location has never been visited by a user. On the other hand, in the collective approach, the chances of the location has never been visited is lower. In contrast, this approach may lead to incorrect predictions, since it does not take into account the individuality movement of each user.

In the hybrid approach, user similarity enhances the spatial and temporal information for mobility prediction since the mobility from a user could be correlated with some user but not all. In this way, we find users with similar routines for mobility prediction. As in Araujo et al. [18], we computed the similarity based on the spatial factor. Therefore, first, we calculated the normalized frequency (f) for each user based on the number of times he visited each location. Hence, the normalized frequency is given by Eq. (1):

$$f_{uid} = \frac{\#\text{ user } uid \text{ visited } loc}{\text{total visits of user } uid} \; \forall \, loc \in \mathbb{L} \tag{1}$$

where uid is the user, loc is the location, and $\mathbb{L}$ is the locations set. After that, since the output of the normalized frequency of each user uid is a probability distribution, we computed the similarity between any two users i and j $\left(i \neq j\right)$, denoted as $SRE\left(i,j\right)$, based on Kullback-Leibler divergene (D_{KL}), more specifically on Jensen-Shannon divergence $\left(D_{JS}\right)$. In this context, both measures (D_{KL} and

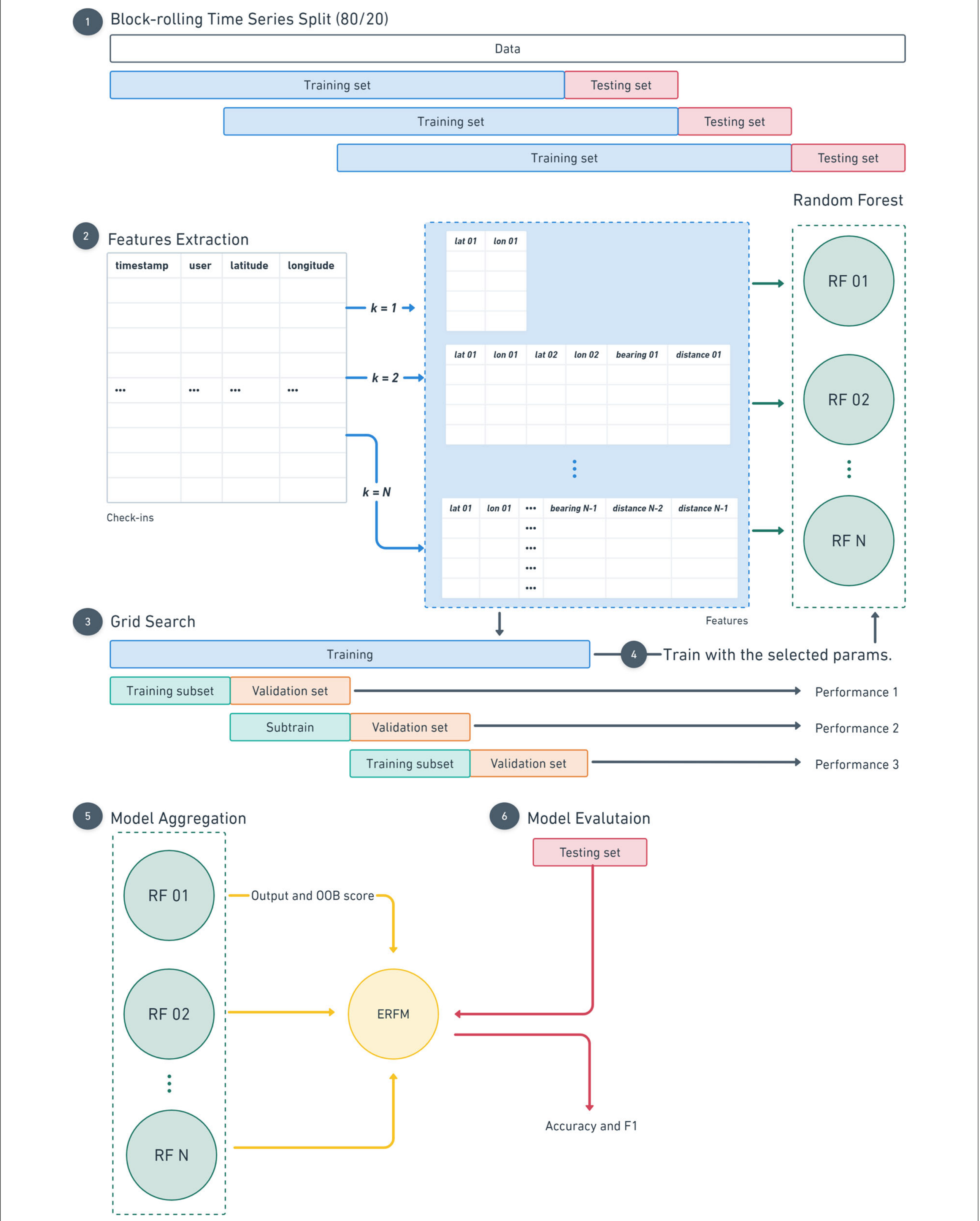

Fig. 1 Ensemble Random Forest-Markov Pipeline

D_{JS}) are usually used to measure the divergence (or similarity) between any two probability distributions. However, differently from D_{KL}, Jensen-Shannon is symmetric and has a normalized value (ranges from 0 to 1). Therefore, we considered similar users those whose *SRE* metric was above a given threshold γ, where $\gamma = 0.7$. We computed the threshold γ by rouding the average of all SRE values. The user similarity is given by Eq. (2):

$$SRE\left(i,j\right) = 1 - D_{JS}\left(f_i,f_j\right) \tag{2}$$

$$D_{JS}\left(f_i,f_j\right) = \frac{D_{KL}\left(f_i,M\right) + D_{KL}\left(f_j,M\right)}{2} \tag{3}$$

$$D_{KL}\left(i,j\right) = \left[\sum_{loc} f_{i,loc}\, log\left(\frac{f_{i,loc}}{f_{j,loc}}\right)\right] \tag{4}$$

where $M = 0.5\left(f_i + f_j\right)$ while $f_{i,loc}$ and $f_{j,loc}$ are the normalized frequencies of the users i and j, respectively, for the location *loc*. Therefore, in the hybrid approach, there will be a trajectory set for each user as in the individual approach. However, each hybrid trajectory set contains own user's individual trajectories and trajectories from the similar users.

Figure 1 (item 2) illustrates the process of extracting features. Hence, in order to build a more sophisticated ML model, besides the coordinates of the sequence of locations, we added two more features for every two subsequent locations: *bearing* and the *distance*. The *bearing* feature (θ) is the angle measured clockwise from the north direction from a location to another and the calculation is given by the Eq. (5). The *distance* feature is the geodesic distance in kilometers between two locations and it is given by the Haversine formula, since we are working with latitude and longitude values and it is usually used for computing the distance. Therefore, for trajectories with length $k \geq 2$ the features are extracted. For instance, for a trajectory with length $k = 3$, two bearing features and two distance features are added, each representing the angle and distance of each user movement.

$$\begin{aligned} A &= \sin \Delta\lambda \cdot \cos \varphi_2 \\ B &= \cos \varphi_1 \cdot \sin \varphi_2 - \sin \varphi_1 \cdot \cos \varphi_2 \cdot \cos \Delta\lambda \\ \theta &= \arctan\left(A,B\right) \end{aligned} \tag{5}$$

3.4 Model building/training/aggregation

Ensemble learning is an ML technique where multiple predictors (often called "weak learners" or "basic models") are trained to solve the same problem and combined to get better results [17]. These basics models often perform not so well by themselves either because they have a high bias, such as low degree of freedom models or because they have too much variance (e.g., a high degree of freedom models). Then, the idea of ensemble methods is to try reducing bias and/or variance of such weak learners by combining several of them to create an aggregated learner (or ensemble model) that achieves better performances.

Traditional ensemble learning approaches only have one layer, i.e., they use ensemble learning once. In this article, we propose ERFM, a two-layer ensemble learning model, in which the weak learners are ensemble learning models. Therefore, in the inner layer, we combine collections of Decision Trees (DT) to create Random Forest models, each of which is based on a different trajectory set according to the trajectory length *k*. Hence, for an order-*k* model, there will be *k* different Random Forest models. In the outer layer, the outputs from the previous layer are aggregated based on the classification performance of each weak learner.

RF performs better than an individual DT on two aspects: overfitting and anomaly isolation. During the RF training process, the outliers are in some of the trees but not in all of them, and thus the aggregation system guarantees the anomalies will be isolated. Also, RF uses the Bagging (Bootstrap Aggregation) approach, which allows each tree to randomly sample from the training dataset with replacement (bootstrap sample), resulting in different trees. Therefore, the voting system minimizes the effect of overfitting concerning the individual decision tree. Also, since each DT takes a different set of training data as input, the deviations in the original training dataset do not impact the final result obtained from the aggregation of DT. Therefore, bagging as a concept reduces variance without changing the bias of the complete ensemble. Moreover, Random Forest can be evaluated using the Out-Of-Bag error (OOB). In this sense, the OOB error is the average error for each training sample z_i calculated using predictions from the trees that do not contain z_i in their respective bootstrap sample.

In the context of hyperparameters optimization, we used a Grid Search approach. Therefore, we split the training set using BRTS strategy into two equally subsets: training and validation and for a given parameter, it chooses the best parameters for a model based on the validation classification performance (see Fig. 1, item 3). In this article, we used the following parameters:

n_estimator: It specifies the number of trees in the forest of the model. The list of values used was $[20, 50, 100]$.
max_depth: It specifies the maximum depth of each tree. The list of values used was $[5, 10, 20, 50]$

After the Grid Search, each RF is trained with the best parameters using the full training dataset (Fig. 1, item 4). Then, ERFM combines all RFs using a weighted average method, where the weight of each base predictor

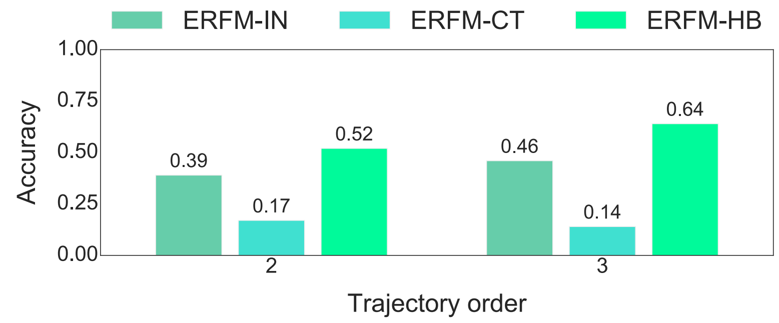

Fig. 2 Validation Accuracy based on different user behaviors: individual, collective and hybrid

is inversely proportional to the OOB error rate (Fig. 1, item 5). Therefore, RFs with a high rate of error receive a low weight value. In the end, we normalize the predictions using the output probabilities. Also, we rank the results from the highest possible location to the lowest one.

3.5 Model evaluation

We can distinguish models according to the type: classification or regression. In the first one, the output is a categorical class label. On the other hand, in the regression problem, the model learns a continuous function. It is common for classification models to predict a continuous value as the probability of a given example belonging to each output class. The probabilities can be interpreted as the likelihood or confidence of a given example belonging to each class. A predicted probability can be converted into a class value by selecting the class label that has the highest probability. In this article, we return a vector containing the highest predicted probabilities. Finally, we select the location with the highest probability.

In order to evaluate the classification performance, we compare different ML methods using two metrics based on the testing set (see Fig. 1, item 6): accuracy and f1-score (see Eqs. (6) and (7)). The former measures the number of correct predictions among the predictions made. F1-score is the harmonic mean of Precision and Recall, where the first is the ratio of correctly predicted positive observations to the total predicted positive observations while the second is the ratio of correctly predicted positive to the total number of actually positive observations.

$$accuracy = \frac{\#\ \text{correctly predicted}}{\#\ \text{predictions}} \tag{6}$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{7}$$

4 Results

ERFM can be built based on the personal or general aspect of user behavior. Therefore, we evaluated the ERFM model based on the three types of trajectories: individual, collective, and hybrid. Also, we used two maximum trajectory lengths: $k = 2$ and $k = 3$, as shown in Figs. 2 and 3 respectively. Moreover, instead of taking the entire dataset to evaluate the ERFMs, we randomly selected 50% of the users. In this sense, ERFM-CT had the worst performance with a low accuracy ratio and f1 score (below 0.17) for both maximum lengths (2 and 3). This worst performance occurs because the model is based on the trajectories of all users to predict the next location Therefore, for a specific trajectory and user, there will be a high number of possible locations that other users visited, which turns difficult to predict correctly. For instance, the higher the number of transitions of a user to a location from a specific trajectory,

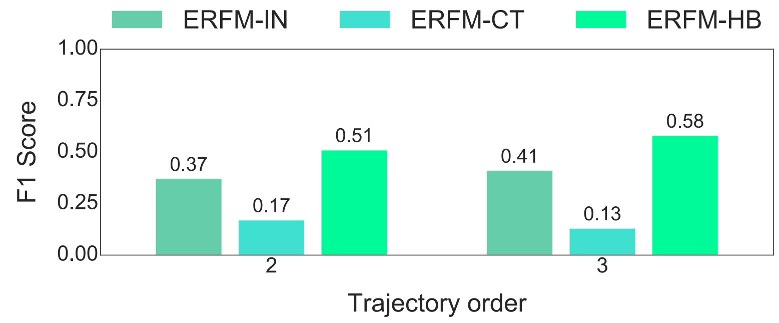

Fig. 3 Validation F1 Score based on different user behaviors: individual, collective and hybrid

the higher the probability of ERFM predict this location for other users if they move from the same trajectory.

On the other hand, ERFM-IN outperforms (absolute value) the collective-based model by 0.22, 0.32 (accuracy), 0.20, 0.28 (f1-score) for $k = 2$ and $k = 3$ respectively. It occurs because, for each user, is trained an ERFM-IN model using their mobility history, enabling the individual-based model to extract individual patterns. On the other hand, ERFM-IN is limited to predict locations that the user never visited, hence, accuracy still low. In contrast, ERFM-HB had the best accuracy (up to 0.64 for $k = 3$) and f1-score (up to 0.58 for $k = 3$) highlighting that the user similarity can be used to enhance ERFM performance.

In this context, since ERFM-HB had the best validation performance, we evaluated it using the full testing set for all users against our previous algorithm TEMMUS [18] and other ML predictors: Adaboost, SVC, Gradient Boosting, and Random Forest. Moreover, all the models use the hybrid trajectories sets as ERFM-HB. Also, they followed the same procedures used in ERFB to train and test. The main difference is on the trajectory length k. While all models (except ERFM-HB and TEMMUS) use a fixed trajectory length, ERFM-HB and TEMMUS use a variable-order approach. The former builds RFs for each order-i trajectories set (trajectories with i sequence of locations), where i ranges from 1 to k, and the last uses a Fallback-Markov Chain, in which decreases the Markov order if a given trajectory is not found on the Markov Chain.The following items detail each model and their associated parameters.

Adaptive Boosting (Adaboost): It is an ensemble predictor that focuses on incorrect predictions. It begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on these cases. We used Decision Stumps (one level Decision Tree) as base estimators (weak learners) with $n\_estimator = 120$

Support Vector Classifier (SVC): It is based on the Support Vector Machine. Given a set of features, it relies on finding the decision boundaries between every two classes.

Gradient Boosting: Similar to Adaboost but differs from it in certain aspects. It is based on the error residuals and a loss function. At each step, we add another weak learner to increase the performance and build a strong learner. This reduces the loss of the loss function. Hence, we iteratively add each model and compute the loss and the predictions are updated to minimize the residuals using this loss value.

RandomForest: It is an ensemble model based on Decision Trees and the bagging approach. While ERFM extracts different patterns by aggregating different RFs based on k value, it uses a fixed k value. In other words, it is equivalent to a simple model from the first layer of ERFM. The parameters used for RandomForest is $n\_estimator = 120$ and $max\_depth = 50$, which are the maximum values used for the tinning the ERFM hyperparameters.

TEMMUS: It is a Markov-based model that leverages the user similarity to predict the next location. Also, it considers trajectories of different sizes using a Fallback Markov-order approach.

Figure 4 illustrates the accuracy metric based on the maximum trajectory length. ERFM-HB had the best accuracy (0.71 and 0.83; $k = 2$ and $k = 3$), followed by Random Forest (0.64 and 0.66; $k = 2$ and $k = 3$), TEMMUS (0.58 and 0.63; $k = 2$ and $k = 3$), Gradient Boosting (0.57 and 0.48; $k = 2$ and $k = 3$), SVC (0.48 and 0.39; $k = 2$ and $k = 3$), and Adaboost (0.36 and 0.21; $k = 2$ and $k = 3$). In the same way, Fig. 5 illustrates the f1-score based on the maximum trajectory length. ERFM-HB had the best f1-score (0.72 and 0.76; $k = 2$ and $k = 3$), Random For-

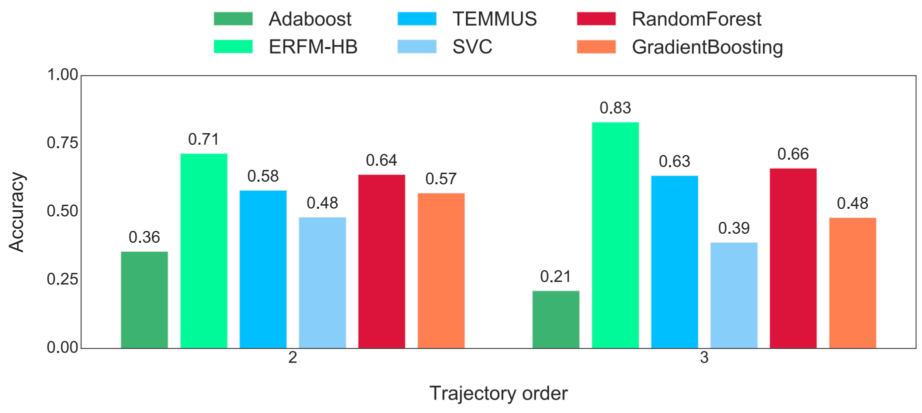

Fig. 4 Accuracy according to the trajectory order

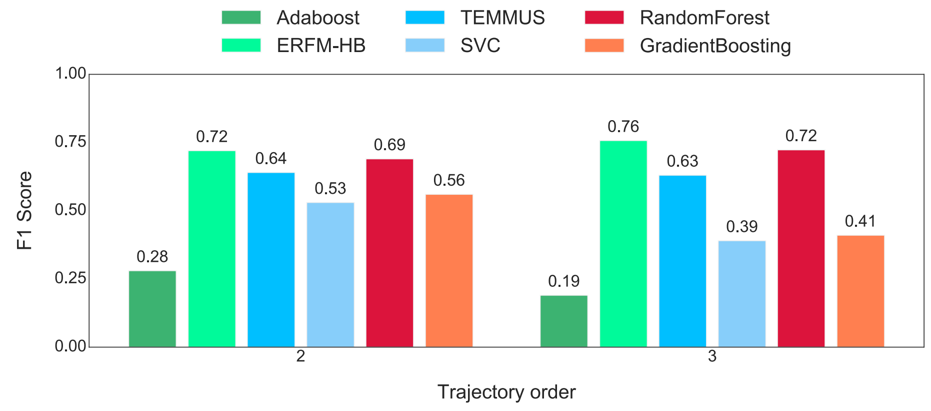

Fig. 5 F1 Score according to the trajectory order

est (0.69 and 0.72; $k = 2$ and $k = 3$), TEMMUS (0.64 and 0.63; $k = 2$ and $k = 3$), Gradient Boosting (0.56 and 0.41; $k = 2$ and $k = 3$), SVC (0.53 and 0.39; $k = 2$ and $k = 3$), and Adaboost (0.28 and 0.19; $k = 2$ and $k = 3$).

The performance of ERFM-HB occurs because it leverages trajectories of lengths. For instance, for the ERFM-HB algorithm, k is the maximum trajectory order since it creates other trajectories ranging from size 1 to $k - 1$, extracting a different pattern form each. In contrast, the trajectory length is fixed for the other models (except TEMMUS). Moreover, the ERFM-HB outperformance is due to its power to identify individual and collective patterns as well as TEMMUS. Also, since RFs are weighted based on the OOB error, it helps in minimizing incorrect predictions. For instance, RFs with high OOB error scores receive a low weight value while RFs with a low OOB error receive a high weight value.

5 Conclusion

In the article, we introduced a two-layer ensemble model based on the Random Forest algorithm and Markovian property, called ERFM. In the inner layer, we combine collections of Decision Trees to create Random Forest models. In the outer layer, the outputs from the previous layer are aggregated based on the classification performance. It predicts human mobility by exploiting user similarity into a ranking-classification approach, based on historical visiting information and by combining trajectories of different lengths using a weighted average aggregation method inversely proportional to the OOB error rate. Moreover, we used the bearing angle as well as Haversine distance between the locations to build a more sophisticated model and extract spatial patterns (direction and distance). We optimized the hyper-parameters using a Grid Search approach based on the parameters *n_estimator* and *max_depth*. ERFM-HB inherited the benefits of the methods, exhibiting high accuracy and f1-score in a challenging and realistic scenario (United States). Therefore, from the results, we can conclude ERFM is a promising solution for predicting human mobility in high-density scenarios.

On the other hand, ERFM has some limitations, such as the cold start problem and memory size. For instance, to extract different patterns, it requires a large dataset, with a high number of check-ins for each user. Hence, it can not be used in low-density scenarios. Moreover, since ERFM is a non-sequential ensemble model, it can build each base model in a parallel way. In contrast, it requires a large volume of memory to create trajectories of different sizes. In future work, we intend to explore new base models, features, and a better approach to overcome ERFM limitations.

Abbreviations

1-MC: First-order Markov chain; 2-MC: Second-order Markov chain; BRTS: Block-rolling time series; CDR: Call data records; D2D: Device-to-device; DT: Decision-tree; ERFM: Ensemble mobility predictor based on the random-forest algorithm and the Markovian property; ERFM-IN: ERFM individual; ERFM-CT: ERFM collective; ERFM-HB: ERFM hybrid; GMM: General Markov model; JS: Jensen-shannon; KL: Kullback-Lieber; LBS: Location based system; LBSN: Location-based social networks; MC: Markov chain; ML: Machine learning; NCV: Nested cross-validation; NLPMM: NextLocation predictor with Markov modeling; OOB: Out-Of-bag; PMM: Personal MarkovModel; RF: Random forest; POI: Points of interests; ROI: Region-of-interest; SLAW: Self-similar least action walk; SVM: Support vector machines

Acknowledgements

This study was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. The author Denis Rosario would like to thank the National Council for Scientific and Technological Development (CNPq) for the financial support through Grant 431474/2016-8.

Authors' contributions

All authors contributed equally. All author(s) read and approved the final manuscript.

Funding

Not applicable.

Availability of data and materials

Please contact the author for data requests.

Competing interests

The authors declare that they have no competing interests.

Author details
[1]Federal University of Pará, Rua Augusto Corrêa, 01, 66075-110 Belém, Pará, Brazil. [2]Federal University of Minas Gerais (UFMG), 13083-852, Belo Horizonte, Minas Gerais, Brazil. [3]Institute of Computing, University of Campinas, 13083-852, Campinas, São Paulo, Brazil.

Received: 1 April 2020 Accepted: 15 October 2020
Published online: 05 November 2020

References

1. Wang Y, Yuan NJ, Lian D, Xu L, Xie X, Chen E, Rui Y. Regularity and conformity: Location prediction using heterogeneous mobility data. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '15. New York: ACM; 2015. p. 1275–84. https://doi.org/10.1145/2783258.2783350.
2. Gao H, Liu H. Data Analysis on Location-Based Social Networks. In: Chin A, Zhang D, editors. Mobile Social Networking: An Innovative Approach. New York: Springer; 2014. p. 165–94. https://doi.org/10.1007/978-1-4614-8579-7_8.
3. Zheng Y. Tutorial on Location-Based Social Networks. In: Proceedings of the 21st International conference on World Wide Web. Lyon: WWW; 2012.
4. Machado K, Boukerche A, Cerqueira E, Loureiro AAF. Long-Term Spatiotemporal Analysis of Social Media for Device-to-Device Networks. In: Proceedings of the IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 4-8 Dec, 2016. New York: IEEE; 2016. p. 1–6. https://doi.org/10.1109/GLOCOM.2016.7841964.
5. Schipor O-A, Wu W, Tsai W-T, Vatavu R-D. Software architecture design for spatially-indexed media in smart environments. Adv Electr Comput Eng. 2017;17(2):17–23.
6. Gao H, Liu H. Mining Human Mobility in Location-Based Social Networks. Synth Lect Data Min Knowl Disc. 2015;7(2):1–115. https://doi.org/10.2200/S00630ED1V01Y201502DMK011.
7. Silva TH, Melo POSVD, Almeida JM, Loureiro AAF. Large-scale study of city dynamics and urban social behavior using participatory sensing. IEEE Wirel Commun. 2014;21(1):42–51. https://doi.org/10.1109/MWC.2014.6757896.
8. Silva TH, Viana A, Benevenuto F, Villas L, Salles J, Loureiro A, Quercia D. Urban computing leveraging location-based social network data: a survey. ACM Comput Surv. 2019;52(1):1–37. https://doi.org/10.1145/3301284.
9. Machado K, Boukerche A, Cerqueira E, Loureiro AA. A socially-aware in-network caching framework for the next generation of wireless networks. IEEE Commun Mag. 2017;55(12):38–43. https://doi.org/10.1109/MCOM.2017.1700244.
10. Abani N, Braun T, Gerla M. Proactive caching with mobility prediction under uncertainty in information-centric networks. In: Proceedings of the 4th ACM Conference on Information-Centric Networking ICN 2017, Berlin, Germany, September 26-28, 2017. New York: ACM; 2017. p. 88–97. https://doi.org/10.1145/3125719.3125728.
11. Yan X-Y, Wang W-X, Gao Z-Y, Lai Y-C. Universal model of individual and population mobility on diverse spatial scales. Nat Commun. 2017;8(1):1639. https://doi.org/10.1038/s41467-017-01892-8.
12. Silveira LM, de Almeida JM, Marques-Neto HT, Sarraute C, Ziviani A. Mobhet: Predicting human mobility using heterogeneous data sources. Comput Commun. 2016;95:54–68. https://doi.org/10.1016/j.comcom.2016.04.013Get.
13. Chen M, Liu Y, Yu X. NLPMM: A next location predictor with Markov modeling. In: Tseng VS, Ho TB, Zhou Z-H, Chen ALP, Kao H-Y, editors. Advances in Knowledge Discovery and Data Mining. Cham: Springer; 2014. p. 186–97. https://doi.org/10.1007/978-3-319-06605-9_16.
14. Wang C, Ma L, Li R, Durrani TS, Zhang H. Exploring trajectory prediction through machine learning methods. IEEE Access. 2019;7:101441–52. https://doi.org/10.1109/ACCESS.2019.2929430.
15. Zhang H, Hua Y, Wang C, Li R, Zhao Z. Deep Learning Based Traffic and Mobility Prediction. In: Machine Learning for Future Wireless Communications. West Sussex: Wiley-IEEE Press; 2019. p. 119–36. https://doi.org/10.1002/9781119562306.ch7.
16. Gebrie H, Farooq H, Imran A. What Machine Learning Predictor Performs Best for Mobility Prediction in Cellular Networks? In: Proceedings IEEE International Conference on Communications Workshops (ICC Workshops), 2019. New York: IEEE; 2019. p. 1–6. https://doi.org/10.1109/ICCW.2019.8756972.
17. Gron A. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd edn. Sebastopo: O'Reilly Media, Inc.; 2019.
18. Araújo F, Rosário D, Machado K, Cerqueira E, Villas L. TEMMUS: A Mobility Predictor based on Temporal Markov Model with User Similarity. In: XXXVII Brazilian Symposium on Computer Networks and Distributed Systems (SBRC), Gramado, Brazil, 6-10, May, 2019. Porto Alegre: SBC; 2019. p. 594–607. https://doi.org/10.5753/sbrc.2019.7389.
19. Wang H, Yang Z, Shi Y. Next location prediction based on an adaboost-markov model of mobile users. Sensors. 2019;19(6):1475. https://doi.org/10.3390/s19061475.
20. Nguyen HA, Giordano S. Context information prediction for social-based routing in opportunistic networks. Ad Hoc Netw. 2012;10(8):1557–69. https://doi.org/10.1016/j.adhoc.2011.05.007.
21. Jiang J, Pan C, Liu H, Yang G. Predicting human mobility based on location data modeled by markov chains. In: Proceedings of the Fourth International Conference on Ubiquitous Positioning, Indoor Navigation and Location Based Services (UPINLBS), Shanghai, China, 2-4 Nov. 2016. IEEE; 2016. p. 145–151. https://doi.org/10.1109/UPINLBS.2016.7809963.
22. Gao H, Tang J, Hu X, Liu H. Modeling temporal effects of human mobile behavior on location-based social networks. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM), San Francisco, USA, Oct., 2013. New York: ACM; 2013. p. 1673–8. https://doi.org/10.1145/2505515.2505616.
23. Cheng C, Yang H, King I, Lyu MR. Fused matrix factorization with geographical and social influence in location-based social networks. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI). Toronto: AAAI Press; 2012. p. 17–23.
24. Munjal A, Camp T, Navidi WC. SMOOTH: a simple way to model human mobility. In: Proceedings of the 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM) Miami, USA, Oct., 2011. New York: ACM; 2011. p. 351–60. https://doi.org/10.1145/2068897.2068957.
25. Dong W, Duffield N, Ge Z, Lee S, Pang J. Modeling cellular user mobility using a leap graph. In: Proceedings of the International Conference on Passive and Active Network Measurement (PAM). Berlin Heidelberg: Springer; 2013. p. 53–62. https://doi.org/10.1007/978-3-642-36516-4_6.
26. Yang D, Qu B, Yang J, Cudre-Mauroux P. Revisiting user mobility and social relationships in LBSNs: a hypergraph embedding approach. In: The World Wide Web Conference. New York: ACM Press; 2019. p. 2147–57.

Publisher's Note