IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 27, 2018, accepted December 28, 2018, date of publication February 13, 2019, date of current version February 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2894809

Improving Grid-Based Location Prediction Algorithms by Speed and Direction Based Boosting

ITAY HAZAN AND ASAF SHABTAI
Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev, Beersheba 8410501, Israel
Corresponding author: Asaf Shabtai (shabtaia@bgu.ac.il)

This work was supported by Telekom Innovation Labs.

ABSTRACT Grid-based location prediction algorithms are widely researched and evaluated. These algorithms usually integrate the speed and the direction during the learning process as regular contextual features, for example, like the time of the day or the day of the week. Unfortunately, the way speed and direction are currently used does not fulfill their potential. In this paper, we propose an alternative approach for integrating the user's current speed and direction in a post-processing mechanism that highly improves the algorithms' accuracy. We dynamically update the probabilities of the predictions provided by the existing (base) algorithms by dividing the surface into four areas while boosting the probabilities in some areas and reducing the probabilities in others. We evaluated our method on three well-known grid-based location prediction algorithms and two different datasets and were able to show that our method improves the predictions made solely by the algorithms. Our improvement was stable during the entire experiment for long-term predictions and for greater prediction distances, particularly in the cold start phase which is considered more difficult to improve.

INDEX TERMS Location prediction, grid-based, mobile, speed, direction.

I. INTRODUCTION

A key component of many of the personalized, context-based, mobile device services and applications (e.g., Yelp, Uber, Google Maps, and Google Now) is the ability to predict the future location of users based on location sensors embedded in these devices. Such capability enables service providers to present relevant and timely offers to their users or manage better traffic congestion control, and more, thus increasing customer satisfaction and engagement.

There are two main types of geographical area spatial representation used by location prediction algorithms: *grid-based* [1]–[4] and *point of interest (POI)-based* [5]–[7].

In the POI approach significant locations are either indicated (i.e., specific points of interest such as a shopping center) or automatically deduced, and the algorithm attempts to predict the next POI the user is expected to visit. POIs can automatically be deduced through clustering algorithms (e.g., the user's location cluster at night is marked as 'home') [8] or by using predefined rules and constraints such as the distance from the closest antenna in GSM networks [9], [10]. However, inferring types of location using the POI approach is a challenging task [6], which usually requires human assistance to continuously label POIs.

In grid-based prediction the map of a certain geographical area is split into cells according to a predefined grid, and the prediction for the location the user is expected to visit (the next cell on the map) is made using a probability function that, given the user's current location and context, assigns a probability to each cell of the grid. The vector of cells and their probabilities produced by the algorithm is referred as the *probability vector*.

In this paper we focus on the grid-based approach and propose a method for improving location prediction by dynamically boosting the probability of predictions, leveraging the recent movement direction of the user to do so.

Location prediction algorithms usually consider the current location of the user, previous routes, and other contextual features such as the time of the day, day of the week [5], [11], movement pattern (e.g., walking, driving), recent phone calls [1] and more information that requires a large training set [12], [13]. While the use of these features can eventually provide more accurate predictions, it requires a large number of observations and consequently a longer learning period.

Movement direction and speed are important contextual features which have been used by supervised learning and rule-based location prediction algorithms in prior research [11], [14]. Other publications proposed spatial regression functions that predict the user's location according to his or her current direction and speed [9]. Thus, previous studies show that the major benefit of the speed and direction features is that they are useful from the first prediction and can also be learned and improved overtime.

These methods, however, require a sufficient amount of observations for each context which results in a longer learning period, or alternatively, they are limited to small areas and are only effective for short-term predictions; i.e., predicting the nearest cells within the next few minutes. Another limitation is that these approaches cannot produce predictions in stationary cases.

In this paper we propose a method that integrates the observed speed and direction of the user differently, in order to improve grid-based location prediction algorithms. The proposed method does not consider the speed and direction of the user as features that have to be learned over time, but rather applies a heuristic approach that utilizes the speed and direction. Specifically, we propose a ''wrapper'' algorithm that can be applied on the predictions (i.e., output) of *any* grid-based location prediction algorithm without requiring learning over time. Thus, our method does not produce predictions but only changes their probabilities and adapts the probabilities of the base location prediction algorithm according to the current context of the user defined by the user's speed and direction.

We applied the proposed method on three grid-based algorithms (frequent cells, Markov chain, and matrix factorization) using two different datasets collected during 2015 in different cities (Jerusalem and Beer-Sheva, Israel) and were able to improve the prediction accuracy on all of the datasets, even in the cold start training period.

Therefore, the main contributions of the paper are as follows:

- we propose a method that can dynamically improve the predications made by any grid-based location prediction algorithm by utilizing the user's current context, represented by the current speed and direction;
- the proposed method does not require that any changes be made to the location prediction algorithm;
- the proposed method can improve both long-term and short-term predictions during the cold start phase of the location prediction algorithms (where there isn't enough training data) and during normal operation of the models over time.

The remainder of this paper is organized as follows: Section II presents related work. Section III provides a description of the proposed method. Section IV presents the evaluations and results, and finally, Section V discusses the conclusions and future work.

II. RELATED WORKS

A. LOCATION PREDICTION

Predicting people's whereabouts at a specific time in the future is a challenging and extensively researched problem. There are two main types of geographical area spatial representation in location prediction algorithms. The first type deals with defining *points of interest* (POI) according to their type (e.g., ''home,'' ''work,'' or ''shopping center'') and its prediction algorithm attempts to predict the user's next POI [5]–[7], [15]. In the second approach, referred to as grid-based location prediction, a specific geographical area is represented as grid of cells, and then, within a given context (e.g., current location, day, hour, previous locations) [16], the location prediction algorithm attempts to predict the cell of a user at a specific time in the future using the user's previously observed transitions between cells [1]–[4]. This is usually performed by assigning a probability to each cell which indicates the likelihood of the user being in the cell at a future time. In our research, we opt to use the grid-based approach because it is usually simpler and easier to implement and does not require inference by location semantics (i.e., POI).

Location prediction algorithms can also be categorized based on the algorithm used to model previous transitions of the user in order to predict the user's future location. Prior studies suggested various classes of algorithms. The simplest class of algorithm is the frequent cells model that simply predicts the most frequent cell(s) for each context (e.g., day and part of day). This basic model was used as a baseline in previous studies to evaluate the performance of proposed location prediction algorithms [1], [17]. This model can also provide an indication of users' predictability. The second class of algorithms are based on different variations of neural networks such as back propagation and radial basis function networks [18] and recurrent neural networks [19]. Another prominent class of algorithms is based on the Markov chain model which was used for predicting the next POI [8], [20] or the next cell in the grid-based approach [1]. This class of predictors represents the mobility of an individual using a Markov chain model, and predicts the next location based on the previously visited location(s) that are part of a trajectory that ends with the current user location [20]. Another state-of-the-art class of algorithms is matrix factorization [21] which was adapted from the recommender systems domain and are able to represent contextual information [22] as well as utilizing data from other users [23].

All the presented categories of algorithm, when applied on grid-based representation, can be used with our proposed speed and direction methods presented in this paper. In order to evaluate our methods, we showed its results on top of three different grid-based location algorithms: frequent cells, Markov model, and matrix factorization.

Previous studies can also be grouped according to whether the prediction is based on the user's own data (personal) or on cumulative data gathered from multiple users (collaborative) [24]–[29]. In this study we combine personal and collaborative data in order to produce better predictions and to enhance the learning phase.

B. SPEED AND DIRECTION

Previous publications considered different contextual parameters, such as day, time of day, movement patterns, speed, direction, and previous locations, as part of the learning process. For example, Fukano *et al.* [30] trained a model for predicting the destination of the user by using features extracted from the smartphone's GPS readings including the day, time, current location, and speed. Ying et al. [14] presented a location prediction method based on collaborative filtering, which utilizes the speed and direction and is aimed at identifying users' stay locations. Nizetic *et al.* [4] used the speed to dynamically set the size of cells in grid-based partitioning when using Markov chains. Once the user exceeds a speed threshold the model is rebuilt on a different grid space.

These methods, however, require a sufficient amount of observations for each context, which results in a longer learning period. In our proposed method, the speed and direction are integrated differently - not as regular features, but as a "wrapper" that can be applied (as a black-box) on the predictions of any grid-based algorithm without requiring learning over time. We suggest a method that does not produce predictions but rather updates their probabilities according to the current user behavior. Therefore, it can improve predictions during the cold start phase of the location prediction algorithms (as we show in the evaluation section), as well as when using the location prediction models over time.

Fülöp *et al.* [31] introduced extensions to widely used mobility models in order to improve their accuracy; in particular, they focused on Markov chain models. In their proposed method, the probabilities of moving to the nearest cells are updated according to the user's current direction. In our proposed method, we also update and boost the probabilities provided by the location prediction algorithm, however in addition to focusing on the nearest locations, we update all possible locations on the grid by also considering the speed. In addition, our method is not specific to one prediction model (e.g., Markov chain).

The speed and direction parameters were also used for deriving location prediction spatial regression models in order to improve SMS delivery [32] and the performance of wireless applications [9]. Such an approach tends to be limited to small areas and, as the authors concluded, is only effective for short-term predictions; i.e., predicting the nearest cells within the next few minutes. Anagnostopoulos *et al.* [33] also proposed a spatial, context-based classifier to predict users' future locations with the limitations of short-term predictions and low computational cost. In this paper we show that the combined speed and direction information can also be used to improve long-term location prediction algorithms.

Several other papers used direction for sequence models. These models predict the next location of the user by using the sequence of previous locations and the user current direction; only adjacent locations in the sequence are considered as possible predictions for the next location. Anisetti *et al.* [34] developed a technique for mobility prediction in GSM networks. After extracting the most likely places for the users to attend, they use Kalman filtering to smooth out the multiple position estimates to form a coherent path. Krumm and Horvitz [2] use the user's speed to estimate the driving duration in order to predict the user's departure at the predicted location. Liu *et al.* [35] develop a hierarchical location prediction algorithm based on two hierarchies, where the lower one applies a dynamic linear model based on the user location, speed, and direction, and the higher one is based on pattern matching techniques for user paths. Anagnostopoulos *et al.* [36] use trajectory distance classification to match the user's current route with previous routes, while proposing a new trajectory distance metric that is delay tolerant. Haitao and Xiangwu [37] use the speed as a contextual nominal feature (high, normal, low) among other features such as weather and time of the day. They apply pattern matching of user's routes and predict the next location for long-term prediction. Jeung *et al.* [38] transformed the location prediction problem into finding paths in graphs. They use the speed and direction in order to select the next edge in the greedy path algorithm. Anagnostopoulos *et al.* [39] built an online updating location prediction model using spatial and velocity context. The authors developed a novel distance metric combining both the sequence of velocities and locations; however, they did not utilize the direction of the user to select the best route.

Our method cannot be applied on, or improve location prediction methods that are based upon route/path matching. These methods consider the user's recent sequence of locations and match it with previously seen sequences of locations in order to find the next location. These methods cannot be improved by boosting areas with direction or speed because in each case only the most matching route is counted and only one predicted location is offered. Our method, however, takes into account an aggregated direction of the user, while using the gradient of several locations within a certain time interval we assume that there are several offered cells for the user next location and not just one.

III. PROPOSED SPEED AND DIRECTION METHOD

In this section we present the proposed method for dynamically updating the probabilities of the cells based on the user's movement direction. Assume current time is t, and we attempt to predict the location of user at time $t + 1$.

The input to the proposed method includes the following:

- *probVector*: The vector of cells and probabilities $\{<cell_1, p_1>, <cell_2, p_2>, \ldots, <cell_m, p_m>\}$ provided by the grid-based location prediction algorithm for a specific prediction request, where the probability $p_i \geq 0$ for each $cell_i$.
- *timeWindow*: A predefined parameter of the proposed method which determines the time frame of the sequence of previous locations to be considered by the algorithm.

- *prevSeq* : The sequence of previous locations (i.e., cells) $l_{t-n}, \ldots, l_{t-1}, l_t$ which the user visited within a predefined *timeWindow*. Note that l_t, the last in the sequence, denotes the current location (cell) of the user (i.e., at the time of prediction).
- *minimalDistance*: Determines the minimal distance that the user is required to travel within the predefined *timeWindow* in order to apply the proposed method on *probVector*.

The output of the algorithm is as follows:

- *newProbVector*: The vector is similar to the input in its cell and structure, but the probabilities are changed, and thus the order of the cells and even the highest probability cell are also changed. The output of the proposed method is the updated probability vector: $\left\{<cell_1, p'_1>, <cell_2, p'_2>, \ldots, <cell_m, p'_m>\right\}$.

The proposed algorithm includes the following main steps which are presented in pseudocode below (Figure 1).

Input:
- $prevSeq - \{l_{t-n}, \ldots, l_{t-1}, l_t\}$, last locations of the user in w time.
- $probVector \quad - \quad \{< cell_1, p_1 >, < cell_2, p_2 > \cdots < cell_m, p_m >\}$, vector of next possible locations.
- β – incline of the divisions

Output:
- $newProbVector - \{< cell_1, p'_1 >, < cell_2, p'_2 > \cdots < cell_m, p'_m >\}$

```
1. $probNewSum \leftarrow 0$
2. $newProbVector \leftarrow \emptyset$
3. $currentLocation \leftarrow prevSeq.getCurrentLocation()$
4. $\nabla prevSeq \leftarrow prevSeq.calculateGradient()$
5. $maxProb \leftarrow \max(\{p| < cell, p > \in probVector\})$
6. $minProb \leftarrow \min(\{p| < cell, p > \in probVector\})$
7. $medianProb \leftarrow \mathrm{median}(\{p| < cell, p > \in probVector\})$
8. $\textbf{for} < cell, prob > \textbf{in}\ ProbVector$:
    8.1. $areaNumber$
        $\leftarrow calculateArea(cell, \beta, \nabla prevSeq, currentLocation)$
    8.2 $\textbf{if}\ areaNumber\ \textbf{is}\ 1$:
        8.2.1. $BoostingCoefficient_1 \leftarrow \sqrt{\frac{maxProb}{minProb}}$
        8.2.2. $newProb \leftarrow prob \cdot BoostingCoefficient_1$
    8.3 $\textbf{else if}\ areaNumber\ is\ 2$:
        8.3.1. $BoostingCoefficient_2 \leftarrow \sqrt{\frac{maxProb}{medianProb}}$
        8.3.2. $newProb \leftarrow prob \cdot BoostingCoefficient_2$
    8.4 $\textbf{else if}\ areaNumber\ is\ 3$:
        8.4.1. $newProb \leftarrow prob$
    8.5 $newProbVector \leftarrow newProbVector \cup \{< cell, newProb >\}$
    8.6 $probNewSum \leftarrow probNewSum + newProb$
9. $\textbf{for} < cell, prob > \textbf{in}\ newProbVector$
    9.1 $newProbVector[cell] \leftarrow \frac{prob}{probNewSum}$
10. $\textbf{return}\ newProbVector$
```

FIGURE 1. Pseudocode presenting the boosting probabilities method.

Step I (Calculating the Movement Trend Based on Previous Location Sequence): In the first step we apply linear regression on the previous cells in the *prevSeq* vector in order to derive the trend line that minimizes the mean square error measure. We denote the gradient of the trend line by $\nabla prevSeq$. This phase is illustrated in Figure 1. Given the sequence of previous locations (indicated by the labeled black dots - $l_1, l_2, \ldots, l_n$), the trend line of the previous location sequence indicated by *PLSLine* is derived. The gradient of the trend line indicates the expected movement direction of the user.

Step II (Partitioning of the Grid Map): Based on the current location (l_t), the trend line *PLSLine* and a dynamically determined angle β, the map is divided into three area types ($Area_1$, $Area_2$ and $Area_3$) as illustrated in Figure 2. Notice that there are two places on the map that represents together $Area_2$. The four areas are defined by four vectors denoted by $innerVector_1$, $innerVector_2$, $outerVector_1$, and $outerVector_2$.

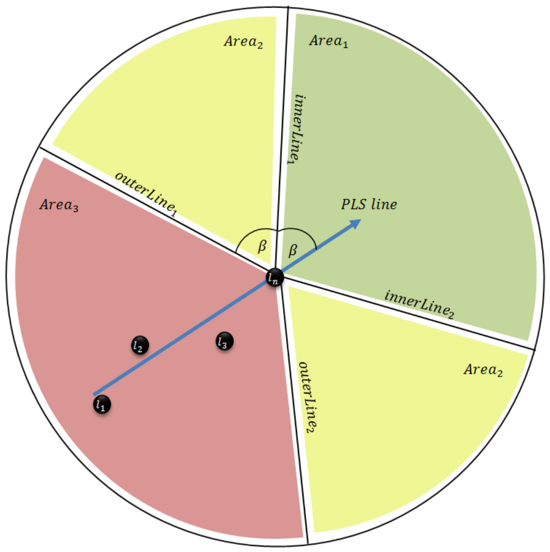

FIGURE 2. Illustrating step I and step II of the proposed method (i.e., dividing the surface into four areas).

The β angle is a main component of the method as it determines the surface division and consequently how the cells' probabilities will be updated. As shown in our evaluations, β can be predetermined for a specific dataset to obtain improvement in prediction accuracy (for the tested datasets the best results obtained for $\beta = 60°$). In addition, we propose adjusting β dynamically by maintaining a regression tree [40] model that is updated over time and aim to predict the optimal β. The features for predicting β were extracted from the *prevSeq* vector and the *probVector* vector as presented in Table 1. The prediction model is continuously updated with the observed optimal β value of previous locations.

Step III (Calculating the Probability Update Coefficients): We define two probability update coefficients (i.e., boosting coefficients). Based on the assumption that there is a higher chance that the user will continue his or her movement according to the calculated *trendPrevSeq*, we update the probabilities of the cells in $Area_1$ and $Area_2$ accordingly (i.e., cells in $Area_1$ are updated with the highest coefficient).

In addition to the fact that we aimed to boost the probabilities of the cells in the direction of the user, we wanted it to fit the current scenario and existing probability distribution. The guiding thought was that if a prediction is in $Area_1$ and higher

TABLE 1. Extracted features for predicting beta.

$prevSeq_t$	$\nabla prevSeq$, mean square error (MSE), St.Dev $\lvert prevSeq \rvert$ (number of samples), user's speed, average latitude and longitude of $\{l_{t-n}, \ldots, l_{t-1}, l_t\}$, coordinates of l_t, passed distance, delta time from l_{t-n}
$probVec_t$	MaxProbability($probVector$), MeanProbability($probVector$) Sum of Probabilities in Area1, both Area2 and Area3 Boosting Coefficient1 Boosting Coefficient2

than the average probability of the prediction in *probVector*, it will be able to bypass the max probability (i.e., *maxProb*) in the *probVector*. We defined the average as the geometrical average of the highest and the lowest ($=\sqrt{maxProb} \cdot \sqrt{minProb}$). In $Area_2$, though, the situation is a bit different, because we want it to have priority over $Area_3$ but remain under $Area_1$. Thus, the cells in $Area_2$ are able to bypass the *maxProb* only if it is above the average of the highest priority and the median ($=\sqrt{maxProb} \cdot \sqrt{medianProb}$).

If a certain probability cell X is equals to the geometrical average of the *probVector* it will be able to be equal to the maxProb. Thus:

$$X = \sqrt{maxProb} \cdot \sqrt{minProb}; X \cdot BoostingCoefficient_1 = maxProb$$
$$\Downarrow$$
$$BoostingCoefficient_1 \cdot \sqrt{maxProb} \cdot \sqrt{minProb} = maxProb$$
$$\Downarrow$$
$$BoostingCoefficient_1 = \sqrt{\frac{maxProb}{minProb}}$$

the same idea work for $Area_2$ except that X should be at least $\sqrt{maxProb} \cdot \sqrt{medianProb}$ which is the geometrical mean of the upper half of the probability vector therefore $BoostingCoefficient_2$ will be:

$$BoostingCoefficient_2 = \sqrt{\frac{maxProb}{medianProb}}$$

Step VI (Updating the Probability Vector): In the last step, the probabilities of the cells in the probability vector *probVector* are updated according to the area ($Area_1$, $Area_2$, or $Area_3$) and the calculated boosting coefficients. After applying the boosting coefficients, we normalize the probabilities for them to sum up to 1 again.

IV. EVALUATION OF SPEED AND DIRECTION

A. EXPERIMENTAL SETUP

In the experiments conducted we set the algorithm parameters as follows:

1) The time window was set to 15 minutes which was assumed to be sufficient for our datasets.
2) The *cellSize* was chosen to be 100×100 meters which is considered in many use-cases to be an accurate prediction but still takes into account the inherent GPS sensor error.
3) In order to reduce the impact of inaccurate location reading by the sensors [41], we consider movements to be transitions with a distance of two cells or more (i.e., $minDistance = \sqrt{2} \cdot cellSize$; the cell's diagonal).
4) Do *et al.* [3] discussed the difficulty of predicting user location for a specific time period in the future, i.e., predicting not only where a user will be next, but also predicting exactly when the user will be there. In our evaluation we attempt to predict the location of a user one hour ahead.
5) Based on previous work we set the trajectory size of the N-Markov model to 2 ($N = 2$). In addition, in order to predict one hour ahead we considered user transitions every 30 minutes. By multiplying the Markov transition matrix, we were able to obtain one hour trip predictions as discussed in [3].

Because users tend to spend most of the time in the same locations (e.g., home, work), predicting cases in which the user did not move from the current location can provide us with an overall accuracy rate approaching 80% [42]. Therefore, our evaluation focuses on improving the cases where the user moved to other locations (cells) which are more difficult to predict. We assume that when predicting the *stay* cases the radius is fixed (and equal to one) around the current cell of the user and therefore the accuracy rate is not affected by the radius prediction methods.

The evaluation was conducted chronologically, such that at the end of each week the location prediction model was updated with the observations from the last week, and the updated model was used for predicting the location during the next week. for evaluation purposes we took 30 weeks from each dataset (BGU and Sherlock). The dynamic boosting model (β and the boosting coefficient) was updated with each prediction.

Grid-based methods, in particular, tend to split places into several different cells in cases in which the point of interest encompasses two or more cells. Previous methods of grid-based algorithms try to overcome this problem by giving a predefined fixed radius surrounding the predicted cell that will cover cases in which a POI is split. Krumm *et al.* [2] showed that radiuses of different sizes produce different levels of prediction accuracy, and therefore the size of the radius needs to be determined based on the accuracy and error required; thus, the evaluation has to conclude several possible radiuses. We refer to this approach as the *fixed-radius*method, which means that if the current fixed radius is 200 meters and the actual location of the user is anywhere within 200 meters from the center of the predicted location cell it will be counted as a correct prediction. For example, if the radius is set to 0, the actual location has to be the exact predicted location cell in order to be counted as a correct prediction. The fixed radiuses we evaluated are *fixedRadius* $\in$ 0, 100, 200, 300, 400, 500.

In our evaluation, we divided the day into six parts (i.e. POD) four hours each, and used them as contextual features for each algorithm. This was done by creating different prediction sub-models of the same type (i.e., algorithm) for each combination of day and part of day (POD). Smaller PODs will result in a larger amount of sub-models and take a longer time to converge, but on the other hand, smaller PODs result in a more accurate prediction model.

In order to understand the effectiveness of the proposed method we implemented three different grid-based location prediction algorithms. The first is the prediction of frequent cells, presented by Song *et al.* [17], the second is a Markov chain predictor widely used in location prediction research [6], and the third is matrix factorization [22]. We applied the proposed method presented in Section 4 on the predictions made by these location prediction algorithms.

B. DATASETS

We tested the proposed method using two different datasets collected in our research labs at the department of software and information systems engineering, Ben-Gurion University of the Negev. Different attributes of the two datasets are presented in Table 2.

TABLE 2. Description of the datasets.

Dataset	City	Users	Sampling interval	Transmission hrs/day avg, (stdev)	Number of cells	Coordinates
BGU	Beer-Sheva	20	1 Minute	22.07 (4.72)	6,800 (80x85)	Min Latitude: 31.212072 Max Latitude: 31.288824 Min Longitude: 34.751552 Max Longitude: 34.835936
Sherlock	Jerusalem	32	5 Minutes	18.28 (6.9)	25,000 (100x250)	Min Latitude: 31.737000 Max Latitude: 31.827000 Min Longitude: 35.137000 Max Longitude: 35.401000

1) BGU

The first dataset contains data collected from 20 Android mobile device owners (students at Ben-Gurion University) over several months in 2015 in the city of Beer-Sheva, Israel. The size of the city is approximately 8 $\times$ 8.5 kilometers, and the area is divided into 80 $\times$ 85 cells, each measuring 100 $\times$ 100 meters. During the experiment the participants were asked to turn on their GPS, and using a dedicated application, we tracked their device every five minutes, resulting in an average number of 2,575 location samples per user per month.

2) SHERLOCK

The second dataset is a public dataset collected by Mirsky *et al.* [43] and contains data from 32 users in the city of Jerusalem, Israel during 2015. We divided the map to 250 $\times$ 100 cells of 100 $\times$ 100 meters each. The participants were given a Galaxy S5 and were asked to use it as their main phone and leave the GPS on as much as possible. The location samples were taken every minute, resulting in an average number of 10,568 location samples per user and month.

Table 2 summarizes the characteristics of the two datasets. The table contains the average number and standard deviation (in parenthesis) of hours with transmitted data per day, along with the number of cells, sampling interval, number of users, and map coordinates.

C. RESULTS

We present the results for the following research questions.

Research Question I: Does comparing the distribution of 'predicted cells' over $Area_1$, $Area_2$, and $Area_3$ with the distribution of 'actual cells' provide the motivation for using dynamic updating of the probabilities?

In order to demonstrate the motivation for the proposed approach, in this initial research question we compare the distribution of predicted cells (by the grid-based location prediction algorithms) over $Area_1$, $Area_2$ and $Area_3$ with the distribution of actual cells (i.e., actual location of the user after one hour).

The mean probability of predicted cells, calculated over all of the predictions in the test set, in $Area_1$, $Area_{2,}$ and $Area_3$ (denoted as 'Predicted') and the distribution actual location/cells over $Area_1$, $Area,_2$ and $Area_3$ (denoted by 'Actual') for each of the datasets (BGU and Sherlock) and for the evaluated grid-based algorithms are presented in Table 3.

TABLE 3. Distribution of users' predicted vs. actual location in Area1, Area2, and Area3 ($\beta = 60$).

Dataset	Algorithm	Area 1 Predicted/Actual	Area 2 Predicted/Actual	Area 3 Predicted/Actual
BGU	Frequent Cell	0.30 / 0.51	0.20 / 0.26	0.50 / 0.23
BGU	Markov chain	0.38 / 0.52	0.26 / 0.29	0.36 / 0.19
BGU	Matrix Factorization	0.40 / 0.46	0.26 / 0.28	0.34 / 0.26
Sherlock	Frequent Cell	0.30 / 0.50	0.24 / 0.29	0.46 / 0.21
Sherlock	Markov chain	0.48 / 0.52	0.24 / 0.27	0.28 / 0.21
Sherlock	Matrix Factorization	0.65 / 0.58	0.29 / 0.27	0.06 / 0.15

Based on the observed mean probability distribution of the algorithms, the following observations can be made. Predictions produced by the frequent cells model usually high in $Area_3$ that is opposite to the user's trajectory. The mean probability of predictions by the Markov model in BGU is highest probability cell is in $Area_1$; however, it can be seen that the mean probability of predictions in $Area_3$ is the second highest and very close to $Area_1$. In addition, it can also be observed that in 51% of the cases the actual location of the user is in $Area_1$; i.e., within the user's general direction. The same observation can also be seen in the BGU matrix factorization, only 40% of the cases the predicted cell is in $Area_1$ while in actual it supposed to be 46% thus giving us the motivation to enhance the $Area_1$ probabilities. In the Sherlock dataset with the Matrix Factorization it is the only case in which the Predicted is higher than the actual in $Area_1$ and the other way around on $Area_3$. In that case we can assume that our method would not work well as $Area_1$ is already boosted.

Therefore, these results indicate that the direction of the user at the time of prediction is not correlated to the probability distribution, thus we can improve the predictions and provide the motivation for dynamically updating the predictions based on the user's direction.

***Research* Question II: How effective is the proposed dynamic boosting approach at improving location prediction?**

In Table 4 we present the mean accuracy of predicting the correct cell any boosting, with constant boosting and with the dynamic boosting approach for each combination of the datasets (BGU and Sherlock) and grid-based algorithms (frequent cells, Markov chain and matrix factorization). The results are averaged over all the fixed radiuses. We define a prediction as accurate (successful) if the actual location falls within the predicted area (the cell with the highest probability after the boosting process is applied).

TABLE 4. Mean accuracy of predicting the correct cell, without any boosting, with constant and with the dynamic boosting approach (*indicates significant difference according to paired t-test).

Dataset	Algorithm	Without boosting	Constant boosting	Dynamic boosting	Number of predictions
BGU	Markov chain	0.612	**0.630 (2.9%)***	**0.652 (6.5%)***	1,146
BGU	Frequent Cells	0.486	**0.525 (8%)***	**0.568 (16.9%)***	5,240
BGU	Matrix Factorization	0.600	**0.623 (3.8%)***	**0.629 (4.8%)***	2,031
Sherlock	Markov chain	0.540	**0.546 (1.1%)**	**0.584 (8.1%)***	16,151
Sherlock	Frequent Cells	0.428	**0.459 (7.2%)***	**0.475 (11%)***	47,144
Sherlock	Matrix Factorization	0.552	**0.567 (2.7%)***	**0.539 (-2.4%)***	23,516

We conducted a paired t-test comparing the mean accuracy of each user and week of location prediction with and without the dynamic boosting method (p-value < 0.01). The results show that the improvement in the frequent cells method with both datasets, and the Markov model with dataset BGU, are significant. For the Markov model with dataset Sherlock, the change in the mean accuracy was insignificant.

We set the constant boosting to be $BoostingCoefficient_1 = 3$ and $BoostingCoefficient_2 = 2$, based on the idea to give suitable boosters to the overall average cases and see what is the performance.

We can also observe that the improvement in the frequent cells method is much higher (16.9% and 11%) than in the Markov model (6.5% and 8.1%) and the last from Matrix factorization (4.8% and -2.4%). This observation can be explained by the fact that the Markov model encapsulates information on the transitions of users and therefore is able to provide better predictions itself, and has less place for improvement and the negative results in the Sherlock with the Matrix factorization we see as we expected.

Moreover, the Dynamic boosting is almost constantly better than the Constant (7.5% average improve and 3.9% average improve respectively).

However, on the other hand, since by providing prediction, the Markov model requires previous information on the user's transitions on the specific current cell of the user, the number of predictions that can be made was relatively low compared to the number of predictions made by the frequent cells model. The matrix factorization method was able to complete some of the data thus have more predictions than the Markov chain model but still less than frequent cells (a total of 52,384 in frequent cells and 25,547 for matrix factorization vs. 15,417 in the Markov model). Note that the differences in the number of predictions is because each algorithm requires different data for making predictions.

Thus, we conclude that by combining the two grid-based location prediction models with the dynamic boosting approach, we can achieve more accurate predictions, and at the same time, obtain a large number of predictions. This can be accomplished by using the Markov model for predicting, and in cases in which a prediction cannot be made, applying the matrix factorization model and lastly use frequent cells model.

Research Question III: Does the dynamic boosting approach help overcome the cold start problem associated with prediction algorithms?

As the prediction algorithms evaluated (and other learning algorithms) rely on sufficient data to provide accurate predictions, we wanted to understand if the proposed dynamic boosting method also improves the predictions during the initial phase of applying the location prediction algorithm when the training data is limited (i.e., the cold start). Therefore, we analyzed the performance of the prediction algorithms on the two datasets during the first ten weeks of the experiment as presented in Table 5.

TABLE 5. Mean accuracy of predicting the correct cell, without the dynamic boosting approach and with the dynamic boosting approach, calculated for only the first ten weeks (*indicates significant difference according to paired t-test).

Dataset	Algorithm	Without dynamic boosting	With dynamic boosting	Number of predictions
BGU	Markov chain	0.662	**0.690 (4.1%)**	334
BGU	Frequent Cells	0.488	**0.578 (18.3%)***	2,334
BGU	Matrix factorization	0.651	**0.681 (4.6%)***	896
Sherlock	Markov chain	0.614	**0.622 (1.4%)**	2,289
Sherlock	Frequent Cells	0.378	**0.438 (15.7%)***	13,047
Sherlock	Matrix factorization	0.640	**0.641 (0.1%)**	3,288

As can be seen in Table 5, the results are consistent with the previous results, indicating that the proposed dynamic boosting method can improve location prediction algorithms even during the initial phase in which the training set is limited.

V. CONCLUSION AND FUTURE WORK

In this research we attempt to leverage the current direction of the user in order to improve location prediction algorithms. The method measures the movement direction of the user, calculates the boosting coefficients according to the probability distribution in the vector, uses a regression tree to predict the β angle that splits the map (dividing the map into three types of areas), and updates the prediction's probability accordingly. We show that we can improve the predictions of the three location prediction algorithms evaluated by using the proposed method on non-stationary cases; we observed an accuracy improvement of 7.5% on average.

In the future we intend to apply this method on additional grid-based location prediction algorithms (e.g., neural network), POI-based location prediction algorithms, and use it to improve the next location radius such as done in [44] in addition to the next location cell.

REFERENCES

[1] T. M. T. Do and D. Gatica-Perez, "Contextual conditional models for smartphone-based human mobility prediction," in *Proc. ACM Int. Conf. Ubiquitous Comput.*, 2012, pp. 1–10.

[2] J. Krumm, and E. Horvitz, "Predestination: Inferring destinations from partial trajectories," in *Proc. Int. Conf. Ubiquitous Comput.* Newport Beach, CA, USA: Springer, 2006, pp. 243–260.

[3] T. M. T. Do, O. Dousse, M. Miettinen, and D. Gatica-Perez, "A probabilistic kernel method for human mobility prediction with smartphones," *Pervasive Mobile Comput.*, vol. 20, pp. 13–28, Jul. 2015.

[4] I. Nizetic, K. Fertalj, and D. Kalpic, "A prototype for the short-term prediction of moving object's movement using Markov chains," in *Proc. 31st Int. Conf. Inf. Technol. Interfaces (ITI)*, Jun. 2009, pp. 559–564.

[5] T. M. T. Do and D. Gatica-Perez, "Where and what: Using smartphones to predict next locations and applications in daily life," *Pervasive Mobile Comput.*, vol. 12, pp. 79–91, Jun. 2014.

[6] A. Sae-Tang, M. Catasta, L. K. McDowell, and K. Aberer, "Semantic place prediction using mobile data," in *Proc. Mobile Data Challenge Workshop (MDC)*, 2012, pp. 1–6.

[7] R. Ahas, S. Silm, O. Järv, E. Saluveer, and M. Tiru, "Using mobile positioning data to model locations meaningful to users of mobile phones," *J. Urban Technol.*, vol. 17, no. 1, pp. 3–27, 2010.

[8] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, "Next place prediction using mobility Markov chains," in *Proc. 1st Workshop Meas., Privacy, Mobility*. New York, NY, USA: ACM, 2012, pp. 1–6.

[9] Y.-L. Tang, D.-J. Deng, Y. Yuan, C.-C. Lin, and Y.-M. Huang, "Dividing sensitive ranges based mobility prediction algorithm in wireless networks," in *Proc. 6th Int. Wireless Commun. Mobile Comput. Conf.* New York, NY, USA: ACM, 2010, pp. 1223–1227.

[10] T. Anagnostopoulos, C. Anagnostopoulos, and S. Hadjiefthymiades, "Mobility prediction based on machine learning," in *Proc. IEEE 12th Int. Conf. Mobile Data Manage.*, vol. 2, Jun. 2011, pp. 27–30.

[11] M. De Domenico, A. Lima, and M. Musolesi, "Interdependence and predictability of human mobility and social interactions," *Pervasive Mobile Comput.*, vol. 9, no. 6, pp. 798–807, 2013.

[12] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw GPS data for geographic applications on the web," in *Proc. 17th Int. Conf. World Wide Web*. New York, NY, USA: ACM, 2008, pp. 247–256.

[13] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on GPS data," in *Proc. 10th Int. Conf. Ubiquitous Comput.* New York, NY, USA: ACM, 2008, pp. 312–321.

[14] J. J.-C. Ying, W.-C. Lee, and V. S. Tseng, "Mining geographic-temporal-semantic patterns in trajectories for location prediction," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 1, pp. 1–34, 2013.

[15] D. Ashbrook, and T. Starner, "Learning significant locations and predicting user movement with GPS," in *Proc. 6th Int. Symp. Wearable Comput. (ISWC)*, Oct. 2002, pp. 101–108.

[16] B. N. Schilit *et al.*, "Challenge: Ubiquitous location-aware computing and the place lab initiative," in *Proc. 1st ACM Int. Workshop Wireless Mobile Appl. Services WLAN Hotspots*, 2003, pp. 29–35.

[17] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[18] S. Bhattacharya and S. S. Singh, "Location prediction using efficient radial basis neural network," in *Proc. Int. Conf. Inf. Netw. Technol. (IPCSIT)*, vol. 4, 2011, pp. 68–72.

[19] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proc. 30th Conf. Artif. Intell. (AAAI)*, 2016, pp. 194–200.

[20] Y. Chon, H. Shin, E. Talipov, and H. Cha, "Evaluating mobility models for temporal prediction with high-granularity mobility data," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Mar. 2012, pp. 206–212.

[21] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[22] H. Gao, J. Tang, and H. Liu, "Exploring social-historical ties on location-based social networks," in *Proc. 6th Int. AAAI Conf. Weblogs Social Media (ICWSM)*, 2012, pp. 1–8.

[23] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "GeoMF: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proc. 20th ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD)*, 2014, pp. 831–840.

[24] J. Lafferty and C. Zhai, "Document language models, query models, and risk minimization for information retrieval," in *Proc. 24th Annu. Int. ACM Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2001, pp. 111–119.

[25] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory," *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, Jun. 2010.

[26] A. Asahara, K. Maruyama, A. Sato, and K. Seto, "Pedestrian-movement prediction based on mixed Markov-chain model," in *Proc. 19th ACM Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*, 2011, pp. 25–33.

[27] A. Fridman, "Mixed Markov models," *Proc. Nat. Acad. Sci.*, vol. 100, no. 14, pp. 8092–8096, 2003.

[28] J. J.-C. Ying, W.-C. Lee, T.-C. Weng, and V. S. Tseng, "Semantic trajectory mining for location prediction," in *Proc. 19th ACM Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*, 2011, pp. 34–43.

[29] Y. Liu, W. Wei, A. Sun, and C. Miao, "Exploiting geographical neighborhood characteristics for location recommendation," in *Proc. 23rd ACM Int. Conf. Conf. Inf. Knowl. Manage.*, 2014, pp. 739–748.

[30] J. Fukano, T. Mashita, T. Hara, K. Kiyokawa, H. Takemura, and S. Nishio, "A next location prediction method for smartphones using blockmodels," in *Proc. IEEE Virtual Reality (VR)*, Mar. 2013, pp. 1–4.

[31] P. Fülöp, S. Imre, S. Szabó, and T. Szálka, "Accurate mobility modeling and location prediction based on pattern analysis of handover series in mobile networks," *Mobile Inf. Syst.*, vol. 5, no. 3, pp. 255–289, 2009.

[32] Y.-C. Lai, J.-W. Lin, Y.-H. Yeh, C.-N. Lai, and H.-C. Weng, "A tracking system using location prediction and dynamic threshold for Minimizing SMS delivery," *J. Commun. Netw.*, vol. 15, no. 1, pp. 54–60, Feb. 2013.

[33] T. Anagnostopoulos, C. Anagnostopoulos, and S. Hadjiefthymiades, "Efficient location prediction in mobile cellular networks," *Int. J. Wireless Inf. Netw.*, vol. 19, no. 2, pp. 97–111, 2012.

[34] M. Anisetti, C. A. Ardagna, V. Bellandi, E. Damiani, and S. Reale, "Map-based location and tracking in multipath outdoor mobile networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 3, pp. 814–824, Mar. 2011.

[35] T. Liu, P. Bahl, and I. Chlamtac, "Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 6, pp. 922–936, Aug. 1998.

[36] C. Anagnostopoulos and S. Hadjiefthymiades, "Intelligent trajectory classification for improved movement prediction," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 44, no. 10, pp. 1301–1314, Oct. 2014.

[37] X. Haitao and M. Xiangwu, "User mobility prediction method based on spatial cognition and context-awareness," *J. Converg. Inf. Technol.*, vol. 6, no. 10, 2011.

[38] H. Jeung, M. L. Yiu, X. Zhou, and C. S. Jensen, ''Path prediction and predictive range querying in road network databases,'' *VLDB J.*, vol. 19, no. 4, pp. 585–602, 2010.

[39] T. Anagnostopoulos, C. Anagnostopoulos, S. Hadjiefthymiades, and A. Zaslavsky, ''On-line location prediction exploiting spatial and velocity context,'' *Int. J. Wireless Inf. Netw.*, vol. 22, no. 1, pp. 29–40, 2015.

[40] L. Breiman, *Classification and Regression Trees*. Evanston, IL, USA: Routledge, 2017.

[41] *Android Location API Reference*. Accessed: Jan. 23, 2019. [Online]. Available: http://developer.android.com/reference/android/location/Location.html

[42] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, ''WhereNext: A location predictor on trajectory pattern mining,'' in *Proc. 15th ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD)*, 2009, pp. 637–646.

[43] Y. Mirsky, A. Shabtai, L. Rokach, B. Shapira, and Y. Elovici, ''Sherlock vs moriarty: A smartphone dataset for cybersecurity research,'' in *Proc. ACM Workshop Artif. Intell. Secur.*, 2016, pp. 1–12.

[44] I. Hazan and A. Shabtai, ''Dynamic radius and confidence prediction in grid-based location prediction algorithms,'' *Pervasive Mobile Comput.*, vol. 42, pp. 265–284, Dec. 2017.

Authors' photographs and biographies not available at the time of publication.