STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXV**, Number 1, 2020
DOI: 10.24193/subbi.2020.1.04

# LOCATION PREDICTION IN MOBILE APPLICATIONS

ALEX CIPCIGAN-RAŢIU

Abstract. The vast developments of mobile technologies and applications in recent years produced a lot of issues to address, as mobile devices have surpassed the usage of classical computers. Predicting the future location of a user who utilizes a mobile application caught the eye of both academia and the industry. Most existing results either use excessive computing, most often relying on a server, or neglect battery usage. We propose a new method that takes these major points into consideration, which gives good results by only relying on the end user's mobile device, not draining the battery, respecting the privacy of the users and that achieves an accuracy of 80%.

## 1. Introduction

Given the growth and vast developments in mobile communication technology in recent years, mobile phones and tablets likewise have become the most used devices by people all around the globe, surpassing classical computers.

One of the most used services and possibly one of the main reasons why mobile took over the world is geolocation. Thus, location-based services (LBS) are nowadays used almost everywhere, in social networks, navigation, advertisement industry, recommendations and so on.

In the current environment, predicting the next location of a user is highly interesting and studied subject [17], which has caught the eyes of academia, as well as the ones of the industry. As research shows, human activities are likely to repeat themselves, having a strong regularity, showing that the potential predictability in user mobility is 93% [14].

There are multiple ways and patterns of predicting the location of a mobile user, but all have in common an important factor: prediction is done based on previously gathered locations that the user has visited. Gathering

Received by the editors: 10 March 2020.
2010 *Mathematics Subject Classification.* 68T05, 91E40.
1998 *CR Categories and Descriptors.* I.2.6 [**Artificial Intelligence**]: Learning – *Analogies*; I.5.2: [**Pattern Recognition**]: Design Methodology – *Pattern analysis*.
*Key words and phrases.* location prediction, mobile, individual next location prediction.

the geographical locations visited by an individual is done through Global Positioning System (GPS) satellites, Wi-Fi and cell towers, the former being the most accurate but also the most battery consuming [8].

Thus, predicting the next location is an interesting topic and new ways of achieving it can emerge, which constitutes a challenge in terms of efficiency and battery consumption. In the following sections, a unique approach that tackles this subject will be presented, which only uses the individual's mobile device, is efficient battery-wise, and gives good results in terms of accuracy.

Based on the stated information, in what follows, this paper will address the issue of predicting the next location of an individual using a mobile application, by illustrating various techniques and explain why opting for a simple solution gives remarkable results.

## 2. Related Work

Existing papers and solutions have been analyzed and compared in order to choose the most suitable one for our purpose. This section aims at taking a closer look at predicting the user location using Markov models [6, 4], k-means clustering mixed with Markov models [1], path matching [7], and decision trees [15] (see Table 1 for a quick comparison of the related works).

Most papers rely on Markov models for predicting the future location of an individual, being one of the best alternatives based on the results it delivers [6]. But, Markov models do require prior computation of collecting the locations (states). This is done most of the time by using a clustering algorithm, such as k-means clustering. Path matching is also an innovative solution, as described in [7]. A rather diverse alternative for tackling this problem is by using decision trees [15]. Using prior gathered points of interest and sequences of visited ones, the next most likely place to be visited can be predicted.

### 2.1. **Markov Models.**

One of the widely used ways of predicting the next geographical location of an individual is by using a Markov model or by extending it. A Markov model is a stochastic model used to represent a randomly changing system [10]. The base idea is that any future state depends only on the current state, not on events that occurred before. This can be extended in order to keep track of more than just the current state, taking into account the $n$ previously visited states and building a $n$-Markov chain ($n \geq 1$).

Certain variants of the Markov model were implemented to tackle the problem in need. Some interesting ones are based on a variant of mobility Markov chains [6] and on a continuous-time series Markov model [4].

Basic mobility Markov chains have no memory, meaning they do not keep track of the previously visited states, as in the idea behind a basic Markov model. As presented in [6], by keeping track of previous states and predicting

| Method Name | Input | Clustering Algorithm | Description |
|---|---|---|---|
| Mobility Markov chain | GPS coordinates | DBSCAN | Mobility Markov chain that keeps track of previous visited states |
| Continuous-time series Markov Model | GPS coordinates and time | DJ-Cluster | Time interval based location prediction using a continuous time series Markov model |
| K-means mixed with Markov model | GPS coordinates | K-means | Markov model |
| Path matching using cellular data | GSM data | Incremental clustering of cells and routes | Previously observed routes by matching their similarities with the current one |
| Decision trees | GPS coordinates | DBSCAN | A decision tree |

Table 1. A comparison of the highlighted related works

using this in mind, the efficiency of the prediction is between 70% and 95% as soon as two previous states are taken into account.

But, even a mobility Markov Chain that keeps in mind the previously visited locations is unable to predict the next location based on a time interval [4]. Thus, [4] proposes a solution in which not only the geographical paths that an individual takes are considered, but time as well. Using a recursive algorithm, it manages to give an average accuracy of 43% [4] when predicting the next location, which is quite high compared to other time-based prediction solutions.

In both solutions presented above, a clustering algorithm is used to gather the set of states, that represent the locations that the individual visited and will visit next. In the former, density-based spatial clustering of applications with noise (DBSCAN) algorithm is used as the clustering algorithm [5]. It groups geographical points that are close to each other, but unlike K-means, it does not need to set the number of clusters in advance. It requires only 2 parameters: the neighborhood radius and the minimum number of points to treat the point as a core point. The latter on the other hand uses a variant of the k-means algorithm to discover the points of interest, called Density-Joinable cluster (DJ-cluster) [18]. This was chosen to the detriment of k-means for the same reason, being that k-means needs to set the number of clusters beforehand. DJ-cluster is adaptive, the points of interest are discovered based on the mobility behavior of the individual.

2.2. **K-Means Clustering with Markov Models.** Another approach is by combining k-means clustering with Markov models. K-means clustering is an unsupervised algorithm that groups similar data in groups called clusters [9]. This is a well-known technique for discovering points of interest, such as locations, hence its popularity among academia and industry.

Identifying locations an individual visits can be done by using this algorithm and clustering places alongside a radius, using a variation of k-means [1]. All the geographical locations within the radius are treated as a common one, which is computed using the mean of all the geographical locations recorded within the radius. Setting a small radius will result in more identified places. But, setting one that is too small will end up identifying only one geographical location for each place. Hence, an optimal radius must be found and set. Having this set up, a Markov model [10] or one of its variants can be used to predict the next location of the user, each node in the model representing a place from the cluster.

2.3. **Path Matching Using Cellular Data.** Another proposed approach in order to identify points of interest is by using Global System for Mobile Communications (GSM) cell-based location data [7]. The data saved consists of a cell, which is identified by a string, and a timestamp associated with the cell. This data is then clustered location-based, each cluster representing a place of interest to the user. A cell cluster is a group of nearby cells. Further clustering the routes using incremental clustering, a route being a string representing cell identifiers, locations that the individual visits can be identified. Prediction is done by taking into account the recently visited cells and the current place visited by the user. The next place that the user will visit is decided by finding all the observed routes having as the starting point the current one. Then, the recently visited cells are compared against cells of those identified routes. The destination of the route that shares the most similarities with the current one will be the result of the prediction. Also, several routes may share the same number of similarities. So, the time of day, day of the week or route frequency are taken into account as well when choosing between two or more routes with the same number of commonalities.

2.4. **Decision Trees.** An interesting proposed solution in predicting the next user location is using decision trees. A decision tree is an algorithm that contains only conditional control statements [2]. It uses a tree-like model of decisions and their possible consequences. Thus, it can be used for predicting the most likely future location to be visited by an individual.

One such solution is presented in [15]. A preliminary step is gathering the points of interest, using the DBSCAN clustering algorithm [5]. Then, a decision tree is built, taking into account the temporal and sequential features. The former denotes the day of the week and the time of the day (split into 24 parts), while the latter refers to the sequence of places the individual visits. Thus, by taking as input the current location of the user and using the built decision tree, the most probable next location the user will visit can be predicted.

As seen in the same paper, the results of the experiments indicate a higher prediction accuracy when using such a decision tree in favor of the classical Markov model, especially in the case of locations with a low-frequency visit.

## 3. Contextual Next Location Prediction

This section details the proposed approach in solving the problem of predicting the next location of an individual in a mobile environment. Using a variant of a k-nearest neighbor algorithm and data collected for a particular individual, we can predict the next location he/she will visit.

### 3.1. **Collecting Data.**

Before being able to predict the future we need to look in the past. By using prior fetched GPS data for an individual we can gather some geographical locations which he/she has been to, giving us a sense of a routine (daily, weekly, etc.)[1]. These GPS coordinates are saved to a database, alongside their recorded time, giving us access to the day, week, month and year. Tying the geographical feature to the temporal one gives us a better model to work with, by associating the presence of the individual in that specific spot with a specific timestamp.

We chose to use GPS data because it is the most precise option in terms of geolocation. Even though it is the most intensive on the battery, we can minimize its effects on the devices' battery by collecting data when the app is in the background or closed, using a background service, scheduled once every couple of tenths of minutes (see Figure 1). Besides this, the background service effects on the battery are limited by the two major operating systems, Android and iOS. They both provide similar power-saving features, such that the effects of background services and unused apps on the battery are minimal [12], Doze, and App Standby for Android, the counterpart on iOS being Standby. Also,

[1]https://github.com/Lexcrd1337/Location-Prediction-in-Mobile-Applications/blob/master/all.gpx

the battery is not that affected because the collecting data process is done periodically and the phone screen is not always on.

With reference to the devices' memory space issues, the data collected can be saved in a cloud-hosted database. For this particular implementation, the Firebase Realtime Database [11] was used. By doing so, the devices do not need to keep the data on them, which would have led to a decrease in the available memory space.

Regarding privacy, only the collected data for each individual is taken into account when predicting one's next location and all the other users are unable to access this information. No other third party data is collected at all and collecting data about an individual cannot be done without the consent of the user. Also, most cloud-hosted database services offer the possibility of assigning security and access rules, as Firebase Realtime Database does.

### 3.2. **Algorithm.**

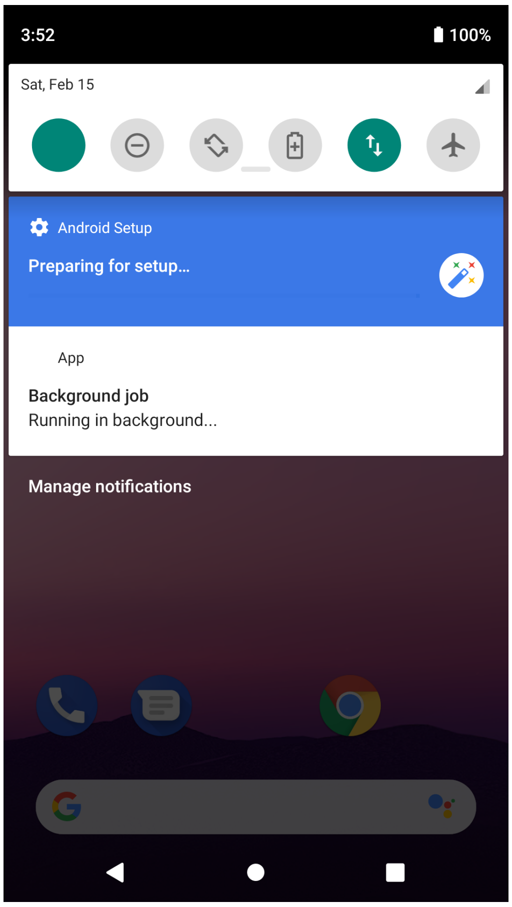
The algorithm proposed for predicting the location is a variant of the k-nearest neighbor algorithm. K-NN is a supervised classification algorithm that gives the $k$ closest data points for a new point [3]. The points in our model are the GPS coordinates collected for an individual.

The proposed variation of the k-NN (see Algorithm 1) takes as input the current timestamp from the mobile device and the GPS coordinates collected for the individual by using the application. It then extracts the day of the week from it and searches existing data collected for the same day of the week. Thus, there are 7 prediction problems, one for each day of the week. Next, it finds the $k$ nearest GPS based locations in which timestamps are the closest in terms of hours, minutes and seconds. For the identified locations, a mean is computed based on latitude and longitude, and the result represents the prediction output of the algorithm, consisting of a GPS coordinate.

Using this k-NN variation offers unique advantages as opposed to other solutions. Most existing work must complete a preliminary step before even starting to try and predict the next location. The step consists of gathering the data, the locations. Also, some of them need to train the data fetched, in order to identify common places or routes. But by using k-NN, there is no need for prior fetching the locations or training the data to recognize certain places of interest or routes. Additionally, it is an active learning algorithm. Each new location collected in the system is used in future predictions as well, thus enlarging its model and accuracy, respectively.

## 4. Experiments and Evaluations

This section presents the evaluation of the prediction algorithm based on collected data for various users. The dataset consists of GPS coordinates in the area of Cluj-Napoca, throughout December 2019 - January 2020 (see Figure


3:52 100%

Sat, Feb 15

Android Setup

Preparing for setup...

App

Background job

Running in background...

Manage notifications

FIGURE 1. Background service (i.e. Background job) for collecting GPS locations when the application is in background or closed

2). In what regards the time feature, the geographical locations were gathered in the time interval between 8:30 AM and 7 PM, with a variation of around 15 minutes between them [1]. More data has been collected during weekdays in comparison with weekends. There are around 80 collected data points for each weekday and 25 for each day of the weekend.

4.1. **Date and Evaluation Metrics.** We have collected GPS data and then started testing it against the algorithm. The locations gathered were saved to a database alongside their timestamp, as previously described.

**Algorithm 1** Next Location Prediction Algorithm
Where *gpsCoordinates* - array of the collected GPS coordinates alongside their timestamps,
*currentDateTime* - timestamp of the current date and time,
*NUMBER OF NEIGHBORS* - how many neighbors to find

```
function PREDICTNEXTLOCATION($gpsCoordinates, currentDateTime$)
    $closestLatitudes$ = [];
    $closestLongitudes$ = [];

    for $neighborNumber$ in $NUMBER\ OF\ NEIGHBORS$ do
        $closestNeighbor$ = findClosestNeighbor($gpsCoordinates$,
                                                $currentDateTime$);
        $closestLatitudes.push(closestNeighbor.latitude)$;
        $closestLongitudes.push(closestNeighbor.longitude)$;
        $gpsCoordinates.remove(closestNeighbor)$;
    end for

    $averageLatitude$ = sum($closestLatitude$);
    $averageLatitude$ /= $NUMBER\ OF\ NEIGHBORS$;
    $averageLongitude = sum(closestLongitudes)$;
    $averageLongitude$ /= $NUMBER\ OF\ NEIGHBORS$;

    return ($averageLatitude, averageLongitude$);
end function
```

In order to asses the results of the proposed algorithm, we computed the difference in meters between the predicted location and the actual location of the user. We compared the output of our algorithm that predicts the next geographical location with a precise value in meters.

As seen in [16], the simple relationship between distance and qualitative proximity between objects is sensitive to complex spatial structure effects, such as the relative location of objects and the geographic extent of the area being considered. Proximity perception is influenced by the size of the frame of reference in a perceiver's mental map. As a result, what is considered near at one scale may be perceived far at another.

Thus, we considered a prediction as being accurate if the computed difference did not surpass the threshold of 50 meters. We considered this value as being an appropriate one because it is an appreciable distance in a real-case scenario, considered close in walkable distance.

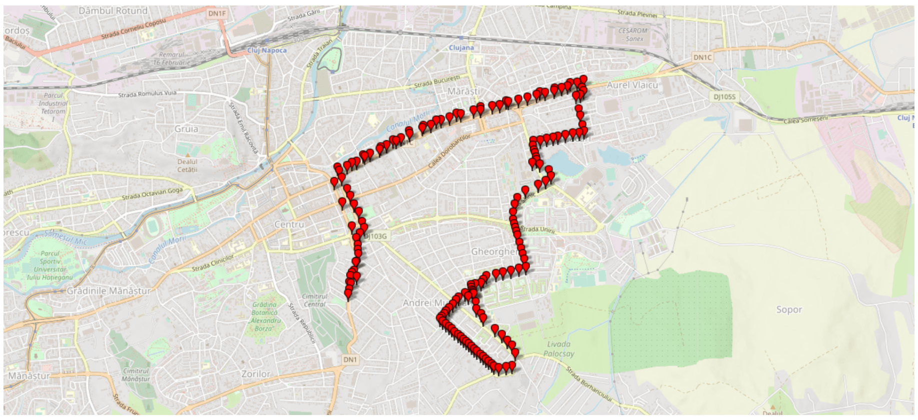


Figure 2. Collected data points in the course of one month for a particular user

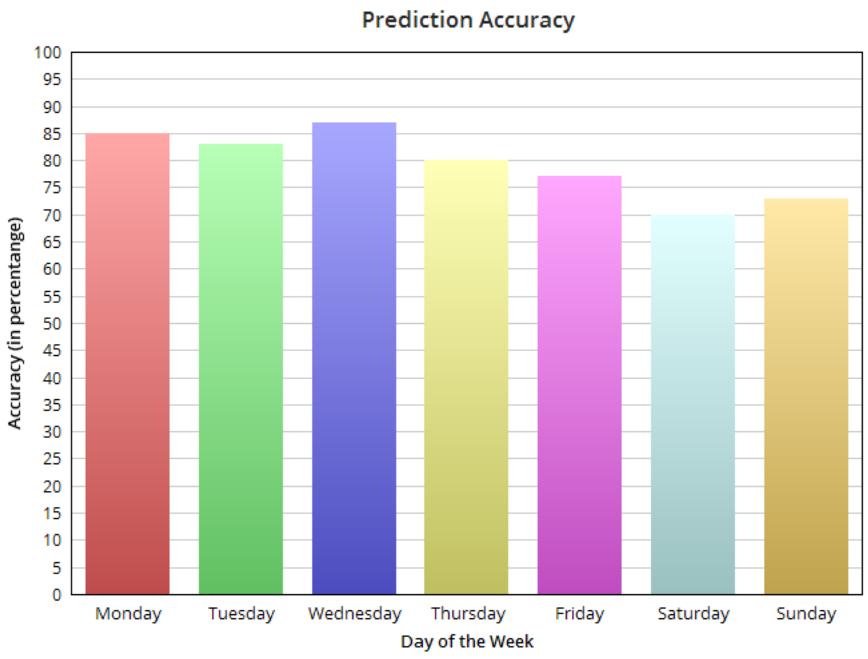


Figure 3. Accuracy of the algorithm tested against the collected data

**Algorithm 2** Finding the Closest Neighbor Algorithm
Finds the closest GPS coordinate based on the current day of the week and time,
*currentDateTime* - timestamp of the current date and time

```
function FINDCLOSESTNEIGHBOR($gpsCoordinates, currentDateTime$)
    $hourAndMinuteArray = []$;

    for $gpsCoordinate$ in $gpsCoordinates$ do
        $hourAndMinutes = gpsCoordinate.hour + gpsCoordinate.minutes$ / 100;
        $hourAndMinuteArray.push(hourAndMinute)$;
    end for

    $hourAndMinuteNow = currentDateTime.hours$;
    $hourAndMinuteNow$ += $currentDateTime.minutes$ / 100;
    $firstItem = hoursAndMinutesArray[0]$;
    $maxDifference = abs(firstItem - hourAndMinuteNow)$;
    $closestNeighborIndex = 0$;

    for $index, hourAndMinute$ in $hourAndMinuteArray$ do
        $difference = abs(hourAndMinute - hourAndMinuteNow)$;
        if $difference > maxDifference$ then
            $closestNeighborIndex = index$;
            $maxDifference = difference$;
        end if
    end for

    return $gpsCoordinates[closestNeighborIndex]$;
end function
```

4.2. **Numerical Results and Analysis.** Based on the chart (see Figure 3), we can see higher accuracy during weekdays, since there was more data collected during them, as opposed to weekends. Moreover, the collected data has a huge role in influencing the accuracy outcome. Weekdays tended to be more repetitive, mostly following the same course of locations. On the other hand, the data gathered on weekends differed from one weekend to another, only Sundays being much alike. Another factor would be the amount of data on which the test was conducted. The data collected consists of only one season, winter. The season greatly influences the places an individual visits [13]. Only the workplace and home do not vary significantly from one season to another. During spring and summer, people tend to go out more and practice more activities in the open, such as riding the bike, hiking, going to the beach, playing outdoor sports and so on. But even so, the test shows a high accuracy nonetheless.

In addition, the results of the experiment may fluctuate if the threshold of accepted predictions is changed. For example, if the threshold would be changed to a value between 30 and 70 meters, the accuracy would remain roughly the same throughout each day. But decreasing it to a value between 5 and 10 meters would result in a decrease of the algorithm's truthfulness with about 10%-15% for each day. Nonetheless, it would perform quite well, giving similar results with the other algorithms that have high accuracy, such as the one using a mobility Markov chain presented in [6].

Since the predictability in user mobility is 93% [14], the results of using k-NN are no surprise. It works best when the user is mostly at a commonly visited place for a given timestamp. It does not give so good results as the other presented solutions if the user is not at a commonplace, but given the fact that it is a fairly simple algorithm, not battery consuming and privacy-oriented, it is an acceptable trade-off to make. Furthermore, by collecting more and more data, its efficiency grows even in these cases, when previously it would perform more poorly than the rest.

4.3. **Threats of Validity.** No artificial intelligence-based algorithm works perfectly, with an accuracy of 100%. Our proposed solution may not behave so great when tested against more complex cases.

In our current scenario, only the day of the week and the current timestamp, including the date and the exact hour and minutes are taken into consideration. There are a lot of factors that can influence an individuals' behavior, hence changing the routes he/she takes. Thus, predicting the future location of an individual is a problematic process.

Moreover, depending only on the timestamp as the only input feature in our algorithm can decrease the accuracy of it in certain scenarios. For example, if we follow a common route, but delay the moment in which we start, the algorithm may not perform that well. It will need time to collect more data in order to accommodate the new timestamps for that specific geographical location. In order to avoid this, we may identify the common routes taken by an individual, and find the time and date intervals when they occur. Thus, by recognizing that the individual takes a common route, but at a slightly different hour or day, the algorithm would no longer drop its accuracy values in these scenarios.

Also, since we have only tested during the course of a month in a certain season, results may lose accuracy when the algorithm is tested for a full year or several years. In that case, we would have to take into account multiple variables. Considering the year and month, besides the current day of the week and timestamp, can improve the results of our algorithm when predicting the location for the course of multiple following years. Hence, extending our model by encapsulating features that can greatly affect how the algorithm behaves, such as the year, month, season or weather, will surely improve its efficiency overall, as well as in those complex scenarios.

## 5. Conclusions and Future Work

This paper presents a simple and accurate solution for predicting the next location of an individual in a mobile application.

Most prior works regarding this particular problem offer an intensive computing approach, not relying solely on the end-users' mobile device. But, the presented solution manages to do all the processing on the individuals' mobile device, in the background, which is not demanding at all in what regards memory and battery efficiency. Furthermore, users' privacy is also respected by using this approach. The experiments that were carried also show the high accuracy of the proposed solution.

Future work will be comprised of taking into consideration multiple features, such as the weather or the month and year as well, not only the day of the week, extending the method to be capable of predicting the next location using the additional features as well.

## References

[1] Ashbrook, D., and Starner, T. Using gps to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous computing 7*, 5 (2003), 275–286.

[2] Breiman, L. *Classification and regression trees.* Routledge, 2017.

[3] Cover, T., and Hart, P. Nearest neighbor pattern classification. *IEEE transactions on information theory 13*, 1 (1967), 21–27.

[4] Du, Y., Wang, C., Qiao, Y., Zhao, D., and Guo, W. A geographical location prediction method based on continuous time series markov model. *PloS one 13*, 11 (2018).

[5] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (1996), vol. 96, pp. 226–231.

[6] Gambs, S., Killijian, M.-O., and del Prado Cortez, M. N. Next place prediction using mobility markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility* (2012), pp. 1–6.

[7] Laasonen, K. Clustering and prediction of mobile user routes from cellular data. In *European Conference on Principles of Data Mining and Knowledge Discovery* (2005), Springer, pp. 569–576.

[8] Lin, K., Kansal, A., Lymberopoulos, D., and Zhao, F. Energy-accuracy trade-off for continuous mobile device location. In *Proceedings of the 8th international conference on Mobile systems, applications, and services* (2010), pp. 285–298.

[9] Lloyd, S. Least squares quantization in pcm. *IEEE transactions on information theory 28*, 2 (1982), 129–137.

[10] Markov, A. A. The theory of algorithms. *Trudy Matematicheskogo Instituta Imeni VA Steklova 42* (1954), 3–375.

[11] Moroney, L. The firebase realtime database. In *The Definitive Guide to Firebase.* Springer, 2017, pp. 51–71.

[12] Narzt, W. Context-based energy saving strategies for continuous determination of position on ios devices. In *2014 47th Hawaii International Conference on System Sciences* (2014), IEEE, pp. 945–954.

[13] Smith, K. The influence of weather and climate on recreation and tourism. *Weather 48*, 12 (1993), 398–404.

[14] Song, C., Qu, Z., Blumm, N., and Barabási, A.-L. Limits of predictability in human mobility. *Science 327*, 5968 (2010), 1018–1021.

[15] Xia, L., Huang, Q., and Wu, D. Decision tree-based contextual location prediction from mobile device logs. *Mobile Information Systems 2018* (2018).

[16] Yao, X., and Thill, J.-C. How far is too far?–a statistical approach to context-contingent proximity modeling. *Transactions in GIS 9*, 2 (2005), 157–178.

[17] Ye, J., Zhu, Z., and Cheng, H. What's your next move: User activity prediction in location-based social networks. In *Proceedings of the 2013 SIAM International Conference on Data Mining* (2013), SIAM, pp. 171–179.

[18] Zhou, C., Frankowski, D., Ludford, P., Shekhar, S., and Terveen, L. Discovering personal gazetteers: an interactive clustering approach. In *Proceedings of the 12th annual ACM international workshop on Geographic information systems* (2004), pp. 266–273.

Babeş-Bolyai University, Department of Computer Science, 1 M. Kogălniceanu Street, 400084 Cluj-Napoca, Romania

*Email address*: caie2278@scs.ubbcluj.ro