Int J Wireless Inf Networks (2012) 19:97–111
DOI 10.1007/s10776-011-0166-9

Efficient Location Prediction in Mobile Cellular Networks

Theodore Anagnostopoulos · Christos Anagnostopoulos · Stathes Hadjiefthymiades

Received: 11 June 2011 / Accepted: 18 October 2011 / Published online: 17 November 2011

Abstract Mobile context-aware applications are capable of predicting the context of the user in order to operate proactively and provide advanced services. We propose an efficient spatial context classifier and a short-term predictor for the future location of a mobile user in cellular networks. We introduce different variants of the considered location predictor dealing with location (cell) identifiers and directions. Symbolic location classification is treated as a supervised learning problem. We evaluate the prediction efficiency and accuracy of the proposed predictors through synthetic and real-world traces and compare our solution with existing algorithms for location prediction. Our findings are very promising for the location prediction problem and the adoption of proactive context-aware applications and services.

Keywords Symbolic location classification · Location prediction · Efficient prediction · Context-aware mobile application

T. Anagnostopoulos · S. Hadjiefthymiades
Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 15784 Athens, Greece
e-mail: thanag@di.uoa.gr

S. Hadjiefthymiades
e-mail: shadj@di.uoa.gr

C. Anagnostopoulos (✉)
Department of Informatics, Ionian University, 49100 Corfu, Greece
e-mail: christos@ionio.gr

1 Introduction

A mobile context-aware application provides information everywhere and anytime through identification, classification, reasoning and efficient processing of the user's context. Context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the integration between a user and an application, including the user and the application themselves [1]. Specifically, context refers to the current values of specific ingredients that represent the activity of an entity (e.g., user, application) and environmental state, (e.g., position, direction and speed of a mobile user), and environmental parameters (e.g., temperature, sound, illumination, humidity).

A very innovative feature of a mobile context-aware application is the proactive behavior and adaptation to the user context. A new class of mobile applications focuses on the efficient prediction on the future mobility behavior of a user. In such applications, the key contextual ingredient is the location of the user. The identification, classification, and prediction of the future location of a mobile user enable the introduction of advanced Location-based Services (LBS) [2]. For instance, location prediction can be adopted to improve resource reservation and assist in critical operations like handoff management in mobile networks [3, 4] and, thus, improve the quality of service experienced by users.

The process of location / context prediction is based on pattern classification in the sense that the user's mobility patterns are collected, processed, classified and used for predicting future mobility behavior. Typically, Machine Learning (ML) is primarily concerned with the design and development of algorithms for pattern classification and prediction. Such algorithms train a system (classifier) to

classify observations in order to predict (predictor) unknown situations based on a history of patterns. Hence, if a mobile application is capable of tracing, learning and classifying the mobility patterns of a user, (i.e., the observed trajectories) then, it can predict his/her future mobility behavior.

In this paper, we adopt pattern classification in order to predict the future location of a mobile user based on the spatial context i.e.,

- the current position and direction of the user,
- the history of the trajectories followed by the user, and
- the information on the user's surroundings (neighbouring network cells).

Each observed trajectory (pattern) is assigned to a specific class (a symbolic location) from a fixed set of classes. Our objective is to predict the symbolic location of a trajectory based on a collected set of popular trajectories. We propose a Location Predictor (LP), which is based on a Trajectory Classifier (TC), in order to address the location prediction problem. We design, implement and evaluate different versions of the proposed LP. Each variant exploits differently the derived knowledge on the mobility behavior of the user. The proposed, classifier-based, LP relies on short-term historical data of the user movement. Prediction is accomplished through the matching of established (historical) movement patterns with the short-term history of the user movement (spatiotemporal knowledge) seen up to the moment of prediction. In our case, a user mobility pattern is a series of symbolic locations. The size of the mobility pattern is directly linked to prediction accuracy. Limiting the size of the user's short-term history leads to loss of information, which results in false predictions and, on the other hand, redundant or excess historical knowledge results to noisy information and biased predictions. The challenge is to introduce an approach based on (a) short term spatio-temporal knowledge on user mobility behavior, and, (b) low computational cost on the decision predictor based on short-term historical movement data. The outcome of the proposed LP is provided through efficient and low complexity ML algorithms. Our findings show a clear advantage on the adoption of ML algorithms for location prediction as we obtain increased prediction accuracy (probability of correctness) at a reasonable computing cost. In addition, we perform a comparative analysis of the proposed LP with popular prediction algorithms reported in the literature ([5, 6] and [7]).

This paper is structured as follows. In Sect. 2, we report prior work on location prediction in mobile networks while in Sect. 3 we define the trajectory classification system. Section 4 discusses the proposed location predictor along with the specific variants and in Sect. 5 we report the parameters and metrics for the performance evaluation. In addition, in Sect. 5 we perform a comparative assessment with other prediction algorithms and schemes found in the literature. Section 6 presents a discussion on certain issues of the proposed predictors and Sect. 7 concludes the paper.

2 Prior Work

Previous work in the area of location prediction includes several models and approaches. The model in [8] applies Naive Bayes classification to the movement history. This model does not deal with fully random or semi-random mobility patterns and assumes normally distributed training data. Such constraints are relaxed in our LPs (due to the adopted non-parametric method—see Sect. 3.1) since mobility patterns range from deterministic to completely random. The authors in [9] apply evidential reasoning for location prediction when knowledge on the mobility patterns is not available. This model suffers from increased computational complexity attributed to the Dempster-Schafer algorithm. It also requires detailed user information (e.g., daily movement profile, favorite meeting places). The method for predicting locations in [10] deals with user movement in highways. However, this model has been limited in scope since it considers only rectilinear movement patterns.

The authors in [11] adopt a rule extraction method for predicting the user location. The time complexity of the prediction process is comparable to our LPs but the prediction accuracy is lower (in the order of 80% for zero randomness mobility—see Sect. 5.1.1). In [12] the authors adopt a clustering method for location prediction. The time complexity is better to our LPs but performance is still lower (in the order of 66% for zero randomness). Similar observations hold for the learning automaton technique used in [13] (for low randomness performance goes up to 70%). The authors in [14] apply specific movement rules for location prediction given the past movement patterns of the user. However, the performance is still low (i.e., in the order of 65% for deterministic movement). In [15] the authors introduce a prediction model that uses grey theory (i.e., a theory used to study uncertainty). This approach also achieves performance lower than our LPs (i.e., in the order of 82% for deterministic movement).

The authors in [16] apply a hybrid model for location prediction. The key component of the model in [16] is a database of association rules. Such model uses association rule mining for specific user patterns. If no rules can be exploited (i.e., an unknown user pattern) the query engine uses a Recursive Motion Function (RMF) to predict the next user location. A major drawback of the hybrid model is that it uses more detailed mobility information (i.e., time and velocity) than our LPs. The performance of the

proposed scheme is not assessed w.r.t. movement randomness (see Sect. 5.1.1). In the prediction model proposed in [17] a single, quantized timestamp is attached to each training trajectory. This timestamp refers to the last cell transition seen in the trajectory. The prediction performance is comparable to some of our LPs. Also, in [18], we compare our LPs with other state-of-art predictors for varying moving behavior. The multi-expert combination method in [18] achieves better prediction accuracy than the other LPs. However, that comes at the expense of higher time complexity for reaching a prediction decision. The method proposed in [19] uses Bayesian learning to train a Neural Network and Markov Chain Monte Carlo methods (MCMC) to obtain a prediction. In addition, the authors in [20] exhibit a Prediction-by-Partial-Match model. The key component of the proposed model is a variable order Markov Model which uses various lengths of previous patterns to build a predictive model. Both prediction models in [19] and [20] assume high computational complexity due to the adopted Markov model. Finally, the authors in [21] present a Maximum Likelihood and greedy algorithm for predicting the location of an object. However, a drawback of this model is that it uses more information (e.g., speed) than our prediction models reported in the paper.

In [22] the authors propose a short-memory adaptive location predictor that works without extensive historical mobility information. The discussed predictor is based on a adaptive local linear regression model. Such model is tuned through a fuzzy controller which tries to minimize the location prediction error and provide fast adaptation to sudden movement changes.

3 Trajectory Classification

3.1 Classification Methods

The process of classification consists of two phases: the learning phase and the testing phase. During the learning phase, the classifier is built i.e., it fits historical data to parameters and data structures. In the testing phase, the classifier is used for predicting the class of an unseen pattern. There are two basic methods for the development of a classifier: the *parametric* and the *non-parametric* method [23, 24]. The parametric method builds a classifier from normally-distributed training examples. However, in our case there is no assumption on the training data distribution. In the non-parametric method, the classification process deals with arbitrary distributions. A notable example of this method is the *k*-nearest-neighbors (*k*NN) classifier. We can distinguish between the *metric* and *non-metric* methods for building a classifier depending on the nature of the classified data. The metric method applies to real-valued data, while the non-metric method applies to nominal data. Finally, the *multi-expert combination* method involves base classifiers that are executed in parallel (each of them with different data assumptions). The final decision, (i.e., prediction) is formed by the fusion of the decisions of each classifier through a voting scheme.

3.2 The Trajectory Classifier

We consider a mobile user who traverses a number of cells in a cellular mobile network. Each cell has a unique identifier x_i (Cell Identifier, CID). A cellular network consists of a set $X = \{x_i : i = 1, \ldots, |X|\}$ of cells ($|X|$ is the cardinality of the set X). We assume that at any given time the user is within a cell $x_i \in X$. We formulate the learning problem of developing a trajectory classifier (TC) and a location predictor (LP), relaying on the TC, as follows:

Given the mobility behavior of a mobile user, construct a TC that classifies a user trajectory of short duration and a LP that predicts the next cell to which that user is expected to move in at the next time slot.

In our approach, TC and LP cover a single mobile user. Each training pair $(\mathbf{x}_i, g_i)$ consists of a trajectory $\mathbf{x}_i = [x_{i1}, \ldots, x_{im}]^\top$ where x_{ij} is a CID and g_i is the CID of the predicted cell, which is called the *class label* for that pair. We adopt the notation $t(x_i)$ to denote the time of user entrance to cell x_i. For each cell in trajectory $\mathbf{x}_i$, it holds strictly that $t(x_{i1}) < \cdots < t(x_{id})$ and the cell g_i is the antecedent of the cell x_{id}, i.e., $t(x_{id}) < t(g_i)$—the next user movement. Then $\mathbf{T} = \{(\mathbf{x}_1, g_1), \ldots, (\mathbf{x}_N, g_N)\}$ is the training set with N training pairs. The classification of an unknown m-dimension trajectory $\mathbf{x}$ is the process of assigning $\mathbf{x}$ to a known class label-cell g. We adopt the classification methods *k*NN (non-parametric method) and Decision Trees (non-metric method) in order to construct the TC.

3.2.1 The Non-Parametric Trajectory Classifier

The *k*NN classifier [25] requires no data structure to be fit. Given a query trajectory $\mathbf{x}_0$, we find the k trajectories $\mathbf{x}_{(r)}, r = 1, \ldots, k$ closest in distance to $\mathbf{x}_0$. Then, the TC classifies using majority voting among the k neighbors, that is,

$$g_0 = \arg\max_{(r)} P(g_{(r)}|\mathbf{x}_0) \quad (1)$$

$P(g_{(r)}|\mathbf{x}_0)$ is the probability for classifying $\mathbf{x}_0$ with class $g_{(r)}$. Hence, *k*NN classifies $\mathbf{x}_0$ by assigning to $\mathbf{x}_0$ the most frequent label among the k nearest trajectories, i.e., the predicted cell g_0. In addition, a trajectory assumes nominal values (CIDs). Consider the $\gamma(\cdot, \cdot)$ indicator such as $\gamma_j(x_{ij},$

$x_{0j}) = 0$ if the two CIDs x_{ij}, x_{0j} are the same and 1 otherwise, where x_{ij} is the jth cell of the ith trajectory, i.e.,

定义轨迹距离计算

$$\gamma_j(x_{ij}, x_{0j}) = \begin{cases} 0 & \text{if } x_{ij} = x_{0j} \\ 1 & \text{otherwise.} \end{cases}$$

The distance between two trajectories $\mathbf{x}_{(i)}, \mathbf{x}_0$ is then

$$\Gamma_{(i)} = \| \mathbf{x}_{(i)} - \mathbf{x}_0 \| = \sum_{j=1}^{m} \gamma_j(x_{(i)j}, x_{0j})$$

选用１ＮＮ，即为测试轨迹匹配最相似的那条训练轨迹

For the kNN, the error on the training data is an increasing function of k and is always zero for $k = 1$. In fact, the 1-nearest-neighbor (1NN) classifier is typically adopted for low-dimensional problems [25], i.e., for low values of m. In our case the dimension of $\mathbf{x}_0$ is $m = 4$ or $m = 3$ depending on the spatial context representation as discussed in Sect. 5.2. If m is small (i.e., 1 or 2) there is no need to perform a prediction once we only obtain a single cell transition. The 1NN partitions the movement space into convex hulls consisting of all trajectories closer to a $\mathbf{x}_i$ than to any other training trajectory. All trajectories of a convex hull are labelled by the corresponding g_i. Hence, the movement space is abstracted as a Voronoi tessellation. In addition, the bias estimate for the 1NN is low as long as it uses only the training trajectory closest to the query trajectory.

所有靠近xi的轨迹组成的凸壳

3.2.2 The Non-Metric Trajectory Classifier

采用C４５决策树分类

Decision trees are a common approach for classifying nominal data, where no similarity and ordering can be established. The classification process proceeds from top to bottom; each node denotes a particular cell of the trajectory. A path from the root to a leaf corresponds to a specific trajectory that leads to a possible g_0, which appears in the tree as a leaf node. Each decision outcome at a node r is called a *split* and corresponds to splitting a subset T_r of the $\mathbf{T}$ training set. Such decision is affected by specific entropy measures of the sub-trees of the current node. However, rather than splitting each node into just two subsets of $\mathbf{T}$ at each stage, we may consider multi-way splits into more than two subsets. The problem is that multi-way splits fragment the $\mathbf{T}$ set quickly, leaving insufficient data at the next level down [26], i.e., we omit the time sequence of cells of the user mobility pattern. As long as, the multi-way split can be achieved by a series of (recursive) binary splits, the latter way is preferred (see discussed below). Hence, the TC classifies a trajectory pattern in node r to the majority class $g_{(r)}$, that is,

从根节点到叶子节点，对应着一个确切的轨迹

$$g_{(r)} = \arg\max_g P_{(r)g} \tag{2}$$

The class $g_{(r)}$ maximizes the quantity $P_{(r)g} = N_{(r)}^{-1} \sum_i (g_i = g)$, where $N_{(r)}$ is the number of observations in T_r, with $\mathbf{x}_i \in T_r$, $(\mathbf{x}_i, g_i) \in \mathbf{T}$; I is unity if it holds that $g_i = g$; otherwise I is zero.

We adopt the C4.5 algorithm [26], which is a popular decision-tree based classifier. In C4.5 each leaf node corresponds to a conjunctive association rule of the decisions leading from root to that leaf—that is the timed sequence of a recorded trajectory. C4.5 also classifies trajectories with missing values, i.e., unrecorded cells and, even, trajectories with spatial gaps. Finally, the adopted C4.5 with binary splits has the capability of pruning / deleting redundant antecedents (i.e., cell transitions) in the formed tree.

3.2.3 The Multi-expert Trajectory Classifier

不懂

A common way to combine the non-metric TC and the non-parametric TC is the voting approach, which is a linear combination of them [27]. Specifically, let $w_j \in [0, 1]$ be the weight for the jth classifier TC(j) (1NN and C4.5 in our case) and d_{ji} be the class posterior probabilities $d_{ji} = P(g_i|\mathbf{x}_0, TC(j)), i = 1, \ldots, N$. The adopted voting scheme assumes that the two classifiers are equally important, which means that $w_j = .5$. Hence, the class g_i that maximizes $P(g_i|\mathbf{x}_0)$ is chosen, that is,

计算两个分类器对同一预测位置的概率之和P，概率和最大的即为所得的预测位置

$$P(g_i|\mathbf{x}_0) = \sum_{j=1,2} P(g_i|\mathbf{x}_0, TC(j)) \cdot P(TC(j)) \tag{3}$$

and $g_0 = \arg\max_{i=1,\ldots,N} P(g_i|\mathbf{x}_0)$. In [18] we present our findings about the selection of the appropriate classifier. We have experimented with 1NN, C4.5 and voting (which includes both 1NN and C4.5).

4 Location Prediction

In our case, the spatial context consists of:

- the current cell identifier x_i of the mobile user,
- the direction of the user movement, which is implied by the pair (x_i, x_j) with $t(x_j) > t(x_i)$ denoting the transition from cell x_i to adjacent x_j cell,
- the set of the trajectories of the user, in which each trajectory comprises m cell transitions, and
- the surrounding cellular information of x_i, i.e., the neighboring cells of x_i.

The design model for the LP and the corresponding TC is illustrated in Fig. 1. Specifically, the system assesses the short-term movement history of the user (m visited cells) and the TC attempts to classify such trajectory into a known symbolic class label. The predictor, based on the classification results of the TC (trained with a $\mathbf{T}$ set), predicts the future symbolic location (i.e., future cell).

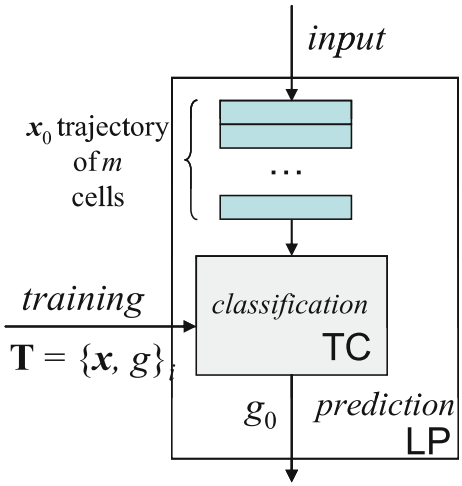


Fig. 1 The TC and the LP design model

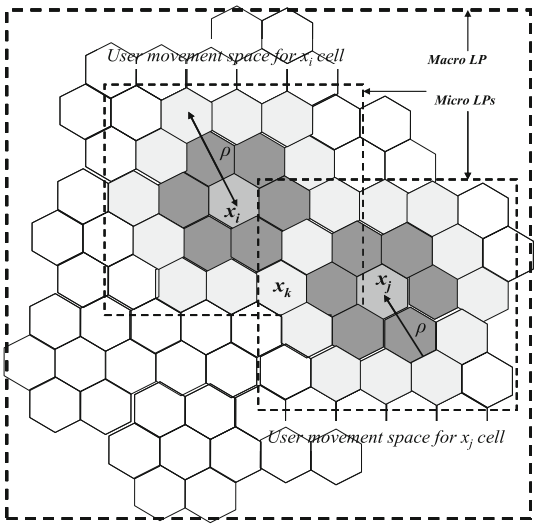


Fig. 2 The Macro LP for a set of X cells and the Micro LPs for the x_i and x_j cells; the x_k cell belongs to $V(\rho, x_i)$ and to $V(\rho, x_j)$

We propose two design alternatives for the LP. We introduce the concept of the Macro and Micro LP. The main difference is the way the corresponding TC builds the knowledge base. In the Macro LP the TC classifier maintains knowledge for all visited cells in the movement space. In the Micro LP, each TC (one for each cell) maintains knowledge only for neighboring cells (see Fig. 2). Furthermore, the LPs are scalable, which means that, if a movement pattern changes then one has to exploit only the cell neighborhood of the newly visited cells and not to rebuild the TC of the Macro LP or all the TCs of the corresponding MicroLPs.

4.1 The Macro Location Predictor

The Macro LP is based on a TC which is fed with the trajectories of the user from the entire cellular network. We introduce the Macro Cell-based LP (MaC) and the Macro Direction-based LP (MaD). The macro feature of the MaC and MaD LPs denotes that the considered spatial contextual information refers to the entire user movement space, i.e., all the CIDs of the network. In the MaC case, there are $|X|$ different cells while, in the MaD case, there are $|U|$ different directions (see below).

In MaC, the trajectory is the time ordered sequence of the last m cells visited by the user. For instance, consider a network of cells $X = \{x_1, x_2, x_3, x_4, x_5\}$ with $m = 3$. The training pair $(\mathbf{x}, g) = ([x_1, x_2, x_3], x_1)$ in MaC consists of the trajectory $\mathbf{x} = [x_1, x_2, x_3]$, which is classified as $g = x_1$, that is the user transits from x_1 to x_2, from x_2 to x_3 and returns to x_1. Consider now the next user movement: the training pair now represents the transitions from x_2 to x_3, from x_3 to x_1 and the next visited cell, say, x_4. Hence, the time shifted training pair is now $([x_2, x_3, x_1], x_4)$. The MaC trajectory is represented through the vector

$$\mathbf{x} = [x_1, \ldots, x_m] \tag{4}$$

where x_m is the cell, in which the user is currently located. The respective training set is

$$\mathbf{T} = \{(\mathbf{x}_i, g_i), i = 1, \ldots, N\}, g_i \in X \tag{5}$$

In MaD, the trajectory is the time-ordered sequence of $m - 1$ direction literals (e.g., NW, W) that the user has followed when moving through m cells. In the hexagonal cell modeling, a direction u from the cell x_i to cell x_j, takes six possible direction literals from the set U = {E (east), W (west), SE (southeast), SW (southwest), NE (northeast), NW (northwest)} assuming an infinite, perfectly segmented and 'oriented' terrain with hexagonal cells. Such space modeling is very popular in the mobile computing literature [28]. In a trajectory $\mathbf{x}$ of m cells, a user has performed $m - 1$ cell transitions. During the jth transition, the user moves from cell x_j to cell x_i, with $x_j \neq x_i$. The symbolic direction of the transition (x_j, x_i) is given by $u_j \in U$. For instance, consider the network of cells $X = \{x_1, x_2, x_3, x_4, x_5\}$ with $m = 3$ from the previous example. A possible training pair $(\mathbf{u}, h)$ consists of a $(m - 1)$ dimensional vector of directions $\mathbf{u} = [NW, E]$ and $h = SW$, that is the user transits northwest from x_1 to x_2, east from x_2 to x_3 and returns southwest from x_3 to x_1. The MaD trajectory is given by the sequence of symbolic directions

$$\mathbf{u} = [u_1, \ldots, u_{m-1}] \tag{6}$$

in which the $(m - 1)$th transition indicates the last direction of the user and the training set is

$$\mathbf{T} = \{(\mathbf{u}_i, h_i), i = 1, \ldots, N\}, h_i \in U \tag{7}$$

4.1.1 Characteristics of the MaC and MaD LPs

None of the MaC and MaD LPs considers how much time a user spent within a certain cell, i.e., the cell sojourn time, or

when, in the course of a day, a cell transition took place. In other words, the MaC and MaD predictors are based only on spatial context (cell identification and direction of movement). In [17] the authors have addressed the issue of time by extending ML algorithms through timestamps. The relevant findings do not justify the overhead of including time information in the underlying knowledge base. The obtained benefit is insignificant. A discussion on the inclusion of time/timestamps in the location prediction process is provided in Sect. 6.

When adopting MaD, we do not predict the next cell $g \in X$ but the future direction $u \in U$, to which the user is heading. Therefore, predicting the u direction and knowing, beforehand, the last x_m cell provides sufficient information to determine the next cell, i.e., the class g_0 for a query trajectory $\mathbf{x}_0$ or query sequence of directions $\mathbf{u}_0$. However, for both MaD and MaC, several characteristics related to physical constraints of the user movement have to be taken into account. Specifically, in MaC, it is assumed that the user moves from cell x_i to cell x_j irrespective of the physical distance between x_i and x_j and the in-between obstacles. However, in reality, the user moves only between neighboring cells. Moreover, moving between any neighboring cells might not be possible due to the presence of physical obstacles, e.g., there is no possible physical route connecting two neighboring cells due to the presence of a hill. In MaD, it is assumed that, the sequences of directions follow similar patterns within the space of cells. For instance, if we observe, at two different cellular 'areas' of the network, the sequence of directions $\mathbf{u}$ = [NW, NW, NW, NW] then the predicted direction is always the same. Still, MaD does not take into account the various physical constraints in the user movement space. Therefore, in the previous example if we are in a cell and cannot move in the NW direction the near future then we will have to follow a different direction than NW. This leads the TC of MaD to classify sequence $\mathbf{u}$ twice each time with different class value, thus leading to potential miss-classification of test sequences. This is the reason of considering MaD less accurate than MaC. On the other hand, if we take into consideration locality information as we will see in Sect. 4.2, we obtain more powerful models like Micro Direction-based LP (MiD). In MiD for the previous example the sequence $\mathbf{u}$ might appear in a number of different cell-transitions, but each one would be treated separately in the areas covered by the corresponding Micro TCs.

The MaC predictor is capable of capturing spatial constraints through the last location of the user x_m. Specifically, the C4.5 TC adopts, as the first attribute in the decision tree the x_m value of the $\mathbf{x}$ trajectory. On the other hand, the kNN TC is not capable of modeling the spatial constraints since it assigns similar semantics to all attributes. The MaD predictor is also incapable of handling constraints, since it has no knowledge of where in the network topology the specific sequence of directions is observed. Nevertheless, we can introduce such information by incorporating the x_m attribute in $\mathbf{u}$, that is

$$\mathbf{u}' = [u_1, \ldots, u_{m-1}, x_m] \quad (8)$$

The $\mathbf{u}'$ represents a trajectory of $m-1$ symbolic directions and the cell identifier x_m. The advantage of this context representation is that, it exploits the direction information and not the actual location. For instance, for certain locations, a direction sequence $\mathbf{u}$ = [NW, NW, NW, NW] could mean that the next direction will also be NW, i.e., the user will continue moving along the same direction (e.g., moving in a highway). Moreover, this representation allows for the discovery of similarities between movement patterns even if these occur in different areas within the user movement space. In order for the C4.5 TC to discover such patterns, we adopt binary splitting (see Sect. 3.2.2).

4.2 The Micro Location Predictor

The Micro LP explicitly accounts for the local spatial specificities of the cell that the user is currently located in, i.e., the physical obstacles in a certain area. The idea in the micro predictor is to build *one* TC(x) for each cell $x \in X$ for a specific mobile user, as depicted in Fig. 2. This means that, instead of discovering and learning the spatial specificities of the entire user movement space, (i.e., for all network cells) a micro classifier TC(x) for a specific cell x is explicitly trained to discover the spatial specificities of the neighborhood (neighboring cells) of x. The TC(x) locally learns the mobility behavior of the user in the area of cell x and, consequently, predicts the next cell to be visited by the user currently found in x.

By assuming a network with $|X|$ cells, we obtain $|X|$ micro classifiers / predictors. This means that, there are $|X|$ corresponding training sets $\mathbf{T}(x)$ which are required for those predictors. Specifically, the $\mathbf{T}(x)$ describes the user movement patterns in the area of cell x. Based on the spatial context representation we obtain the Micro Cell-based LP, MiC(x), and the Micro Direction-based LP, MiD(x), $x \in X$ (see Fig. 2). The cell neighborhood $V(\rho, x)$ of MiC(x), is defined as the set of cells that are in distance of ρ cells from x, $|X| \geq \rho \geq 1$. Hence, the training pair $(\mathbf{x}_i, g_i)$ for MiC(x) is the same as that of MiC by omitting the x_m cell since this is always the same for the micro classifier. This means that, a micro predictor MiC(x) is used for classification and prediction once the last location of the user trace is the cell $x_m = x$. That is the trajectory for the MiC(x) is

$$\mathbf{x} = [x_1, \ldots, x_{m-1}] \text{ with } x_i \in V(\rho, x), i = 1, \ldots, m-1 \quad (9)$$

and the training set for the MiC of a specific x cell is

$$\mathbf{T}(x) = \{(x_i, g_i) : g_i \in V(1,x) \setminus \{x\}, i = 1, \ldots, N(x)\} \quad (10)$$

$N(x) \leq N$ is the number of trajectories corresponding to the x cell. Consequently, the length of x is now $m - 1$ instead of m. The training set $\mathbf{T}(x)$ for the MiD(x) is similarly defined with $h_i \in U$.

不同单元建立相似的移动模式是不可能的

The weakness of building locally TC(x) classifiers and predictors for each network cell x is that it is not possible to establish movement patterns that are common between different cells. Consequently, the knowledge derived from the movement patterns corresponding to a TC(x_i) cannot be exploited in a different cell x_j. It should be noted that, x_j might also be a neighboring cell of x_i. For instance, the movement pattern of directions **u** = [NW, NW, NW, NW] might appear in a number of different cell-transitions, but each one would be treated separately in the areas covered by the corresponding micro TCs.

5 Performance Assessment

In this section we evaluate the four proposed location predictors, namely, the Macro LPs, MaC and MaD, and the Micro LPs, MiC and MiD. Each TC of each LP combines the base 1NN and C4.5 classifier through the voting method in Sect. 3.2.3. The prediction results for the Micro LPs are the average performance of each micro LP(x) for all network cells $x \in X$. At first we define the specific parameter degree of movement randomness for the simulated mobility data and then introduce certain metrics for the performance evaluation of the proposed LPs. Specifically, a LP is evaluated w.r.t. 关于

1. the probability of correct predictions (accuracy),
2. the statistical significance between the proposed LPs,
3. the computational effort required for a LP to reach a prediction and
4. the prediction efficiency metric which indicates the capability of a LP to predict a future user location w.r.t. a given prediction accuracy and mobility behavior.

Finally, we provide a qualitative and quantitative comparative assessment of the proposed LPs with other prediction algorithms and schemes in the literature.

5.1 Parameters and Metrics for Location Prediction

5.1.1 Degree of Movement Randomness

We introduce the parameter *degree of movement randomness*, $\delta \in [0, 1]$ in order to express the way a mobile user transits between cells and changes directions. The adoption of the δ parameter provides an objective criterion for assessing the proposed LPs under various levels of uncertainty and unpredictability. The δ degree indicates the transitional behavior between cells. A trajectory can be either characterized as a deterministic movement as indicated by a low δ value or as a random movement as indicated by a high δ value. The use of δ can guarantee a correct interpretation of the performance evaluation of a predictor since a high prediction accuracy might not necessarily indicate an efficient predictor if the patterns were quite deterministic. The deterministic trajectories represent regular movements, for instance, the route from home to work. On the other hand, random trajectories represent purely random movements between predefined locations (e.g., a quick detour for a coffee after leaving home and before getting to work). Therefore, a value of δ close to unity does not mean an explicitly non-deterministic mobility behavior. Instead, it is constrained by obstacles in the considered space.

对用户方向变化性、在单元间移动随机性的度量

We used synthetic trajectories from the Realistic Mobility Pattern Generator (RMPG) [29]. The RMPG produces transitions and traces of a mobile user in a cellular network. The synthetic trajectories involved δ values in the set {0.0, 0.25, 0.5, 0.75, 1.0}. The five discrete values of δ range from the regular pattern ($\delta = 0.0$ with 500 example trajectories) to completely random pattern ($\delta = 1.0$ with 1000 example trajectories). It is worth noting that the value of δ influences also the size of the training trajectories **T** since the more random a movement is, the more transitions are generally required for a certain itinerary (i.e., moving from specific origin to specific destination).

5.1.2 Probability of Correct Prediction

The most common statistical metric for assessing the capability of a LP to predict the future user location is the probability of correct predictions. The TC is trained with a learning set **T**. The produced classifier TC(**T**) is then applied on a test set **Y**, such that $\mathbf{Y} \cap \mathbf{T} = \emptyset$, and the corresponding predictor LP is evaluated. Specifically, the prediction accuracy for a predictor LP is quantified through the *probability of correctness*, $a(LP) \in [0, 1]$. The $a(LP)$ value is quantified through the correct predictions achieved by the LP over **Y** after the learning of the TC(**T**). That is, let $g_{\mathbf{P}}$ and $g_{\mathbf{Y}}$ be the predicted and actual cell identifier of a mobile user at his mth cell transition assuming the trajectories $\mathbf{x_P} \in \mathbf{Y}$ and $\mathbf{x_Y} \in \mathbf{Y}$ of the visited $m - 1$ cells, respectively. Then $a(LP)$ is the ratio

用T训练好的轨迹分类器TC

$$a(LP) = \frac{|\mathbf{P}|}{|\mathbf{Y}|} \quad (11)$$

of the correctly predicted trajectories $P = \{\mathbf{x_P} | g_{\mathbf{P}} = g_{\mathbf{Y}}, \mathbf{x_Y} \in \mathbf{Y}\}$ from the **Y** set. A popular method to estimate

the $a(LP)$ value is the re-sampling method cross-validation [30, 31]. In n-fold cross-validation, the training set $\mathbf{T}$ is divided into n subsets $\mathbf{T}_i, i = 1, \ldots, n$, of equal size. The classifier TC($\mathbf{T}$) is trained n times, each time setting aside one $\mathbf{Y} = \mathbf{T}_i$, which will be used as the test set for the determination of the a value. The reported value is the average value for the n repetitions of the learning and testing phases.

每次留出一个作为测试集

5.1.3 Statistical Significance

统计显著性

A LP depends heavily on the training set $\mathbf{T}$ of its corresponding TC, thus, there is a need for statistical testing in order to (i) assess the expected prediction error rate of the LP, and, (ii) compare the expected prediction error rates of two LPs, thus, arguing about their relative efficiency. It is important to determine whether the difference $|a(p) - a(q)|$ for p and q LPs is *statistically significant*, i.e., whether the difference in the a values for p and q is not attributed to statistical errors.

The statistical significance $e(p, q) \in \{0, 1\}$ is determined through the McNemar test [32]. Based on such test, we are interested either in the proportion of the correctly predicted trajectories by p and in the proportion of the incorrectly predicted trajectories by q for the same training set $\mathbf{Y}$ or the opposite. Both p and q predictors are trained and tested with the same $\mathbf{T}$ and $\mathbf{Y}$ sets, respectively. Hence, the McNemar test indicates whether a correct prediction by p and a false prediction by q is more or less likely than the reverse. In the case that p and q are statistically insignificant, i.e., $e(p, q) = 0$, the difference of their probabilities of correctness is meaningless and, thus, both predictors can be, safely, considered as equivalent; otherwise $e(p, q) = 1$. The threshold level for the e value is typically set to 0.05 [23]. This value derives from the chi-square distribution with one degree of freedom.

5.1.4 Computational Complexity and Prediction Efficiency

The probability of correctness and statistical significance are not the only metrics for evaluating the prediction efficiency of a LP. To reach safe conclusions, we have also to consider the time taken by a LP to reach a decision (prediction). Such time is of prime importance in certain mobile applications.

Let N be the number of the classified training trajectories of m cells. The complexity of the adopted 1NN is $O(m)$ for the distance calculation and $O(mN)$ for classification. However, a Voronoi tessellation implementation of 1NN exhibits complexity $O(1)$ in time and $O(N)$ in space by adopting the partial distance, search tree and editing algorithms [33]. Moreover, the adopted C4.5 (with binary splitting) results in $O(N) + (N - 1)O(m)$ calculation time for splitting a training example. The total average time complexity for tree construction is $O(mN\,(\log N)^2)$ as long as the splitting of the root exhibits time complexity $O(mN \log N)$ time complexity, the splitting of the two nodes in the first level assumes $O(mN\log(\frac{N}{2}))$ time complexity, in the second level $O(mN\log(\frac{N}{4}))$, and so on. The classification time is proportional to the depth of the tree, i.e., $O(\log N)$ and the space complexity is proportional to the number of nodes, that is, $O(N)$. Hence, the final classification complexity for a macro LP ranges from $O(1) + O(\log N)$ to $O(dN) + O(\log N)$. For a micro LP the classification and prediction complexity is reduced. This is attributed to the fact that, the number of trajectories required for the training of a micro LP is significantly smaller than the number of all trajectories in the user movement space. For a LP, we introduce the *prediction efficiency* metric, $\beta(LP)$, of the prediction accuracy $a(LP)$ against the required prediction time $\tau(\delta)$ given a certain movement history $\mathbf{T}$ and movement behavior δ, that is

$$\beta(LP) = \frac{a(LP)}{\tau(\delta)} \tag{12}$$

For two LPs, whose TCs are trained with the same $\mathbf{T}$, with similar $\tau(\delta)$ the efficient predictor is the one that achieves higher a values. Similarly, for two LPs, which assume identical a values, the efficient predictor is the one that demonstrates the lowest prediction time $\tau(\delta)$. In both cases, the efficient LP has to assume a high β value. Table 1 and Table 2 summarize the parameters and the introduced metrics for the performance evaluation of the location predictors.

5.2 Location Prediction Evaluation

The MaC, MaD, MiC and MiD predictors were implemented in the Weka ML workbench [34] and fed with traces produced by the RMPG. The RMPG traces were generated for a single moving user with appropriately defined movement profile. Such profile was tuned to be compatible with the values of the δ degree. We used a 40-day training set $\mathbf{T}$, which corresponds to N training

Table 1 Parameters for a location predictor

Parameter	Range	Notation
m	>2	Trajectory length
N	>m	Number of training pairs
$\|X\|$	>6	Number of network cells
δ	[0, 1]	Degree of mobility behavior
n	>1	n-fold cross validation (repetitions of the learning and testing phase)

Table 2 Metrics for evaluating a location predictor

Metric	Range	Notation
$a(LP)$	[0, 1]	Probability of correctness for LP
$e(p, q)$	{0,1}	Statistical significance between p and q LP
$\beta(LP)$	$\mathcal{R}$	Prediction efficiency for LP

pairs. Specifically, we obtained $N = (2200, 2495, 3173, 3790, 4582)$ for $\delta = (0.0, 0.25, 0.5, 0.75, 1)$, respectively. The user movement space was a cellular network of $|X| = 100$ cells.

The basic parameter for the proposed LPs is the length m (visited cell) of the trajectory derived from the user movement. Specifically, if the mobility behavior is highly random (e.g., the user is moving within the city center) then the longer the history we gather in the trajectory the greater the probability gets of inducing noise into the regression model. It is worth noting that a large value of m implies that the predictor takes into consideration an extensive movement trace which possibly includes a large count of sudden movement changes. Specifically, we experimented with the *voting* scheme (see Sect. 3.2.3) in order to examine which value of m of the training pair is appropriate in terms of a high probability of correct prediction (combining the classifiers 1NN and C4.5). Moreover, we have to take into account not only the a value but also the mobility behavior δ of the mobile user for determining the value of m. A different mobility behavior results to different prediction accuracy. Figure 3 depicts the a value that is obtained for certain m values ($3 \geq m \geq 20$) w.r.t. the δ. For regular patterns ($\delta = 0$) any value of $m \geq 4$ results to similar prediction accuracy, thus, we can use the minimum value (i.e., $m = 4$) for tracing a movement history. For mobility behaviors which range from lightly regular to fully stochastic we can observe that for $m = 4$ we obtain the highest a values compared to other m values. Hence, the trajectory length of each training pair and tracing trajectory for prediction is $m = 4$, through which we also obtain a more efficient 1NN (see Sect. 3.2.1). Finally, we performed a 10-fold cross validation (i.e., $n = 10$) for measuring the probability of correct predictions for each LP.

Table 3 shows the probability of correct predictions for all proposed LPs w.r.t. the degree of movement randomness. The $a(MiC)$ and $a(MiD)$ derive values from the average $a(LP(x))$ value for all micro predictors for any cell $x \in X$. One can observe that for a high value of δ ($\delta > .75$) all predictors demonstrate a relatively satisfying prediction accuracy. In addition, for a moderately stochastic mobility behavior the probability of correct prediction for all predictors is close to 0.7. We can see that both macro and micro predictors assume very similar values in the a metric. This was quite expected since we adopted the same TCs for all the proposed LPs.

In order to choose the most efficient LP w.r.t. the a metric, we also performed the statistical significant test

Table 3 The probability of correct predictions $a(LP)$ vs. the degree of movement randomness δ for all LPs trained with a 40-day set

δ	MaC	MaD	MiC	MiD
0.00	0.9558	0.9635	0.9626	0.9584
0.25	0.7452	0.7460	0.7543	0.7551
0.50	0.7156	0.7186	0.7228	0.7144
0.75	0.6648	0.6505	0.6693	0.6790
1.00	0.6690	0.6555	0.6775	0.6783

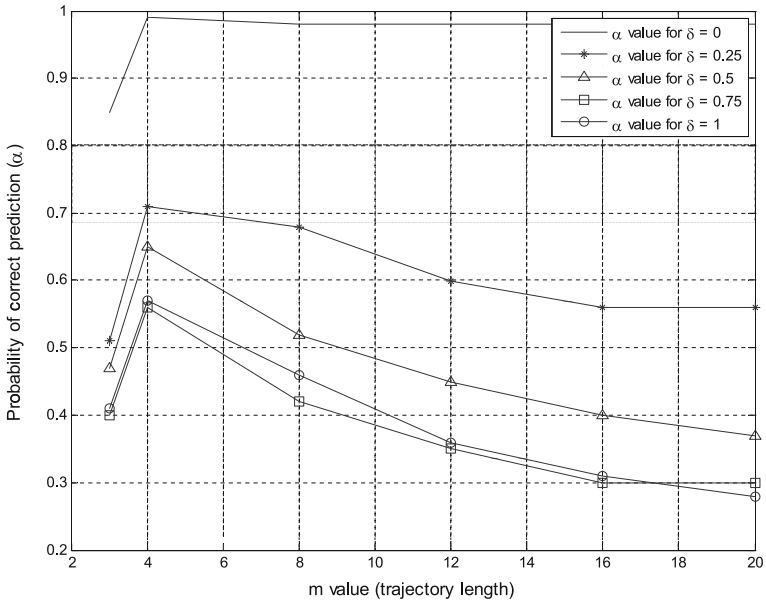


Fig. 3 The probability of correct predictions vs. m for different δ values

(see Sect. 5.1.3) for each pair of the proposed LPs for different training set $\mathbf{T}$ with $\delta = 0.25$ (a rather regular mobility behavior). In Table 4 we observe that, for training sets consisting of y-day ($10 \leq y \leq 40$) user movements, we obtained satisfactory prediction results (with converging accuracy) even with limited training data ($y = 10$ days). Based on this experiment the statistical relations among LPs are represented by the statistical significance test values (see Sect. 5.1.3) as shown in Table 5.

By adopting a micro LP (MiC or MiD), we capture the specific knowledge for the movement patterns related to a cell and its neighboring cells. Therefore, we are more confident on a micro LP than on a macro LP. In a macro LP (MaC or MaD), instead, we cannot acquire knowledge related to a given area (i.e., the cell neighborhood) as long as the LP learns the user movements for the entire user movement space (cellular network). Hence, based only on the statistical significance test on the a metric in Table 5 and the values of the a metric from Table 4, the most efficient predictors are MiC and MiD. However, we have to identify the most efficient LP based not only on the a metric and the corresponding e metric but also on the classification and prediction time and space required for reaching a prediction decision given an a value.

We examine the behavior of the current efficient predictors MiD and MiC w.r.t. the prediction efficiency metric β for exactly the same training set $\mathbf{T}(\delta)$ used for producing the a value. Figure 4 depicts the β value vs. the δ mobility behavior indicator. It is worth noting that the size of $\mathbf{T}(\delta)$ increases linearly with the value of δ. In the same figure we have plotted the β value for all the proposed predictors for comparison reasons. As reported in Sect. 5.1.3, for the two micro LPs whose a values are statistically insignificant, the efficient one is the predictor with the lowest prediction time $\tau(\delta)$. One can observe in Fig. 4 that the MiD predictor exhibits the highest β indicating that it achieves the highest probability of correctness with minimum classification and prediction time. The counterpart MiC predictor ranks second but with significant distance from MiD. This is very important because, although there is not statistical significance between MiD and MiC (see Table 5), the MiD achieves a far better β value; we obtain a 58.57% increase in β for $\delta = 0.5$). The MaD assumes a low β value (70% decrease in β on average for all δ w.r.t. the corresponding MiD) and MaD performing better than MaC (56.66% increase on average for all δ). To sum up, the most efficient

Table 4 The probability of correct predictions $a(LP)$ vs. the size of the training set (in days)

Days (y)	MaC	MaD	MiC	MiD
10	0.6785	0.6590	0.7077	0.7012
20	0.7260	0.7092	0.7290	0.7245
30	0.7416	0.7234	0.7516	0.7305
40	0.7452	0.7460	0.7543	0.7551

Table 5 Statistical significance metric for all proposed LPs for the a value

$p(LP)$	$e(p, q)$	q (LP) MaC	q (LP) MaD	q (LP) MiC	q (LP) MiD
$p(LP)$	MaC	–	0	1	1
$p(LP)$	MaD	0	–	1	1
$p(LP)$	MiC	0	0	–	0
$p(LP)$	MiD	0	0	0	–

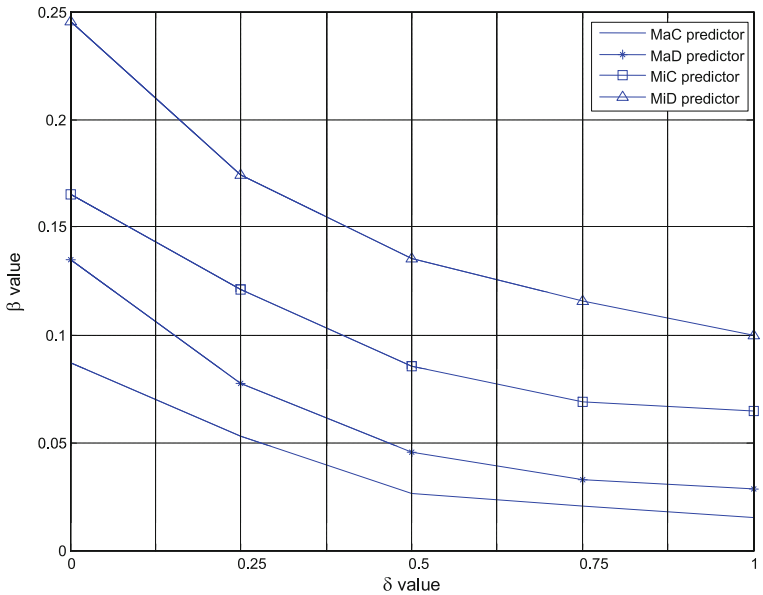


Fig. 4 The β vs. δ value for the proposed location predictors

LP of the proposed variants is the MiD predictor. In the comparative assessment we compare the MiD predictor with other prediction algorithms and schemes in the literature w.r.t. the a and β metrics and provide the corresponding computational complexity analysis.

Finally, it is worth noting that we also evaluate the proposed LPs predictor not only with synthetic RMPG traces but also with and real-world GSM cell data from the MIT Media Lab Reality Mining project [35] and [36]. The real-world data refer to the movements of a MIT student during a period of 2.5 years (from 2004 until the middle of 2006). Specifically, the student daily commutes from home to work and travels from home to downtown. Moreover the movement patterns range from very simple (daily commute, weekend and holiday trips) to moderately complex. We have used spectrum analysis in order to assess the equivalency of the RMPG synthetic trace with the MIT data. We processed 22 40-day sets of visited cell identifiers for the MIT student. Half the traces (eleven) were found equivalent to RMPG traces with randomness $\delta = 0.5$. For $\delta = 0.5$ the probability of correctness of the RMPG trace for the MaC predictor is $a(MaC) = 71.56\%$ and the probability of correctness for the MIT student trace is $a_{mit}(MaC) = 70.97\%$. Since both a and a_{mit} are comparable and statistical insignificant we can safely conclude that $\delta = \delta_{mit}$. Quite similar observations were made for the MiC, MaD and MiD models.

5.3 Comparative Assessment

In this section we provide a comparative assessment of the proposed LPs with location prediction algorithms extensively cited in the relevant literature as we consider that their features, complexity and assumptions render them viable solutions to the location prediction problem. Specifically, we are mostly concerned with the efficiency and the complexity exhibited by these algorithms during location prediction.

5.3.1 Computational Complexity in Location Prediction

The approximate pattern-matching algorithm, Hierarchical Location Prediction (HLP), in [37] uses pre-recorded regular trajectories to estimate the inter-cell direction. Such algorithm uses an extended self-learning KF for unclassifiable random trajectories. HLP, which is a semi-Markov process, calculates k state transition probabilities. Each state refers to a discrete level of speed acceleration, thus, adding $O(k^2)$ space and time complexity. HLP uses the edit distance for matching an observed trajectory of length cm (c is constant) to a pre-defined trajectory (of length m), thus, adding $O(cm^2)$ time and $O(m)$ space complexity. In our proposed LPs, the distance calculation between observed and pre-defined trajectories is $O(m)$ time and $O(1)$ space complexity. Additionally, HLP highly depends on very specific spatial information that is, exact user position, speed, direction, cell geometry, cell residual time and overlapping areas among cells. On the other hand, our LPs require only the identifiers of the present and past cells visited by the user. HLP has to continuously track the user location, thus, leading to a large amount of historical data. In addition, due to such detailed information, HLP makes assumptions not well harmonized with mobile environments. HLP strictly depends on the underlying data as it assumes smooth movements. In our models, the adopted TCs are trained for any distribution of the underlying mobility information. Finally, HLP exhibits a satisfactory probability of correctness once the KF stabilizes, that is $a(KF)$ reaches 0.75 for any $\delta > 0.3$. This can be achieved by the MiD predictor without the need for extensive historical information and data distribution assumptions.

We also compare the efficiency of our LPs with a HMM approach for location prediction. The adoption of a HMM as a possible TC in our LPs, i.e., instead of the 1NN and C4.5 TC, presents the following drawbacks: (1) the time and space complexity [38], (2) the inability of HMM to classify totally unknown trajectories (the same problem as in [5]) and, (3) the limited scalability of HMM when adding new, unvisited cells (all emission probabilities and probabilities of hidden states have to be recalculated). Specifically, by assuming the first, u_1, and last, u_m, directions of a vector $\mathbf{u}$ then the HMM calculates all possible hidden cell-transitions between u_1 and u_m for the corresponding x_1 and x_m cells. This means that, HMM calculates the probability $P(\mathbf{x})$ that the user produces a trajectory $\mathbf{x}$ of m hidden cell-transitions. In the MaC (MaD) with $|X|$ hidden states and a sequence of m observations we obtain $|X|^m$ possible terms in calculating $P(\mathbf{x})$, thus, leading to $O(m|X|^m)$ complexity which is prohibitive. However, for the MiD(x) (MiC(x)) an improvement in the calculation of $P(\mathbf{x})$ can be achieved with $O(m|V(\rho, x)|^m)$. The quantity $|V(\rho, x)|$ increases proportionally to the square of ρ, that is, $|V(\rho, x)| = c\rho^2$ in a hexagonal cellular network, in which $c = 6$. Note also that the HMM forward algorithm reduces the computational complexity to $O(m|X|^2)$ -far more efficient than the time complexity associated with exhaustive enumeration of paths—but stills leaves the space complexity unjustifiably high. The HMM adopted in [38] achieves performance comparable to our LPs at the expense of increased (location prediction) time complexity.

5.3.2 Efficiency in Location Prediction

In [7], the authors introduce the Temporally Annotated Sequence (TAS) mining. We have implemented TAS in order to compare its prediction performance with that of our LPs. Specifically, based on the model in [7], in our case a

temporally annotated sequence is defined as a couple $(\mathbf{x}, A)$, where the trajectory $x = [x_1, \ldots, x_m]$ is of length m and $A = [(t(x_2) - t(x_1)), \ldots, (t(x_m) - t(x_{m-1}))]$ is the handover time between cells (A is referred to as the temporal annotation of a TAS in [7]). The prediction performance of the TAS model is in the order of $a(TAS) = 0.69$ with $\delta = 0.25$, compared to our LPs which achieve a prediction performance in order of $a(LP) = 0.75$ with respective δ.

In addition, we deal with the characteristics of two known algorithms [5, 6]. Such algorithms were also simulated and compared to our LPs. The predictor in [5] models user mobility patterns as a sequence of some elementary movement patterns. A pattern matching algorithm, Mobile Motion Prediction (MMP), is proposed in order to estimate the future location of the user. The main drawback of this algorithm is its high sensitivity to movement randomness. Any trajectory that cannot be classified through the predefined elementary mobility patterns is treated as random. As reported in [5], the prediction accuracy of MMP decreases linearly with the increase of the δ factor.

The predictor in [6] adopts concepts from lossless compression (i.e., the Lempel-Ziv algorithm) for location prediction. This predictor (LeZi-Update) creates dynamically a search tree (trie), which is a dictionary of path updates. Such dictionary supports an adaptive on-line method that learns the user mobility behavior. After processing the input trajectory, a blending strategy is used to predict the next location of the user.

In Table 6 and Fig. 5, we provide a quantitative comparison of the algorithms discussed in [5] (MMP), [6] (LeZi-update) and [7] (TAS) w.r.t. the a and β metric, respectively. Such algorithms were implemented by the authors as standalone Java applications to facilitate the comparative assessment process. The proposed MiD and the MMP, LeZi and TAS predictors were fed with the same training set (a 40-day set $\mathbf{T}$). In Table 6, one can observe that the TAS, LeZi-Update and MMP predictors achieve lower $a(LP)$ values w.r.t. the MiD predictor as δ value increases. In addition, the TAS and LeZi-Update is more stable for great values of δ than MMP. Nevertheless, we have to examine the efficiency of the predictors. Figure 5 depicts the β value for all considered predictors. We can observe the efficiency of the MiD predictor, which is 522% higher than that of the TAS predictor on average for all values of δ. This is very interesting since the MiD and TAS predictors assume quite similar a values; we obtain a 11% difference in the a value between MiD and TAS, on average, for all δ. On the other hand, the MMP and LeZi predictors assume quite similar and low values of β. Note that the LeZi predictor has 59.75% better prediction accuracy than MMP (average a value for all δ values). Based on such comparison results we can conclude on the fact that the β metric is much more crucial than the a metric for LP schemes that are exploited by certain mobile applications.

Table 6 The probability of correct predictions vs. the degree of movement randomness for the MiD, MMP, LeZi and TAS predictors

$a(LP)$				
MiD	MMP	LeZi	TAS	δ
0.9584	0.5785	0.8332	0.9525	0.00
0.7551	0.3543	0.4567	0.6908	0.25
0.7144	0.2822	0.3841	0.6444	0.50
0.6790	0.1513	0.3123	0.5908	0.75
0.6783	0.1118	0.2719	0.5642	1.00

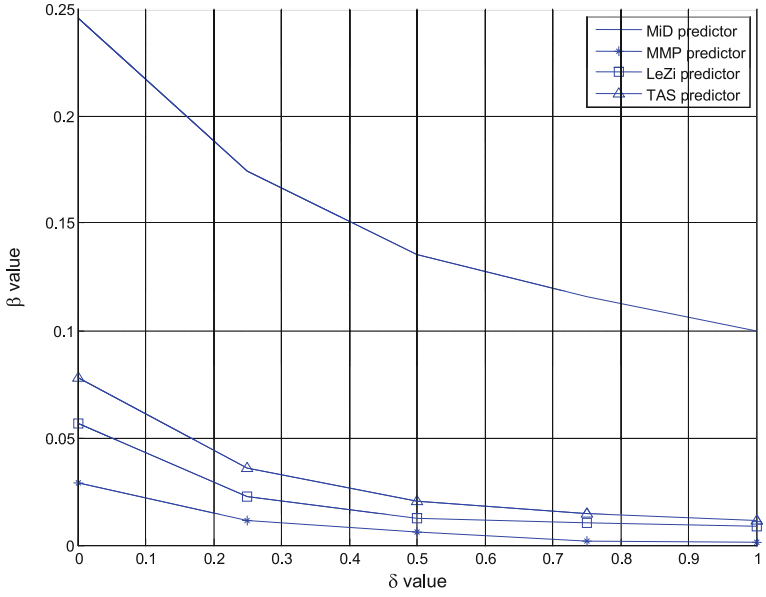


Fig. 5 The β vs. δ value for the MiD and the compared location predictors

6 Discussion on Time-dirven Location Prediction

In the authors' view, many different approaches for the inclusion of time in location prediction can be through of. We consider two cases as discussed below.

Case 1: The 'simplest' predictor is completely independent from time, i.e., spatial context is time-agnostic. This predictor implies that quite similar movement patterns are observed in different time instances (e.g., different periods throughout the day), therefore, it is meaningless to include time information in the predictor. Inclusion of time information implies the assertion of extra information in the underlying knowledge base and more complex reasoning schemes. Such overhead should therefore, be incurred only if it leads to considerable performance improvement. The absence of time information in the underlying ML model is adopted in this paper and in [18].

Case 2: The underlying spatial context model is enriched with time information. Time information should be quantized (i.e., adopt the granularity of quarter of an hour or half hour) to avoid a state space explosion in the underlying knowledge base. Apart from that, it is reasonable to assume small deviations in the mobility behavior exhibited by nomadic users. Patterns with similar spatial characteristics should be treated similarly irrespective of small time misalignment. Time information can be inserted in the knowledge base in two different ways, depending on the history window that the prediction scheme adopts. Specifically,

- If the history window, involves more than one transition, time information can be attached to each of these transitions (this is the case studied in [7]).
- An alternative approach involves the attachment of time information only to the last transition (this is the case considered in [17]). Evidently, this option is the only feasible scenario when the history window spans a single cell transition.

In Case 2, time information is used to distinguish between different spatiotemporal patterns and, further, specialize the decisions taken by the ML scheme. A LP could be enhanced with a residual time estimation scheme. This scheme, calculates, based on information collected from different sources (e.g., the network, historical data), the approximate handover time. Such estimate can be exploited by intelligent resource management schemes and advanced services [39, 40]. The handover time estimation supplements the decisions taken by the previously discussed algorithms (i.e., location prediction) to deliver an integrated mobile predictive model.

7 Conclusions

In this paper we proposed efficient LP schemes based on ML algorithms for trajectory classification. Specifically, the proposed spatial context classifier and a short-term predictor for predicting the location of a mobile user in cellular networks exploits (i) the current position and direction of the user, (ii) history of the trajectories of the user, and, (iii) surrounding location information. We design, implement and evaluate different variants of the proposed LP. Each variant exploits differently the derived knowledge on the mobility behavior of the user (namely the macro and the micro LP). We define the parameters of the short-term LP and introduce certain metrics for evaluating the ability of correct predictions and the efficiency in the prediction process. Moreover, we compare our LP (all the corresponding variants) with popular predictors discussed in the relevant literature. Such predictors are also based on ML algorithms. Simulations with synthetic and real-world mobility data shown that the proposed short-term MiD predictor achieves high prediction efficiency and accuracy, thus, delivering LPs suitable for advanced context-aware applications.

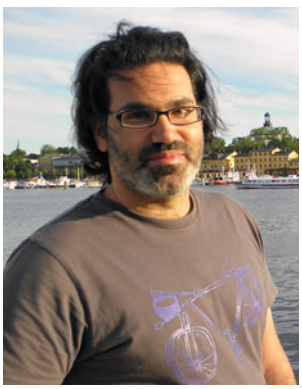
References

1. A. Dey, Understanding and using context, *Personal and Ubiquitous Computing*, Vol. 5, No. 1, pp. 4–7, 2001.
2. J. Hightower and G. Borriello, Location systems for ubiquitous computing, *IEEE Computer*, Vol. 34, No. 8, 2001.
3. I. Priggouris, E. Zervas, and S. Hadjiefthymiades, Location Based Network Resource Management, chapter in "Handbook of Research on Mobile Multimedia" (editor: Ismail Khalil Ibrahim), Idea Group Inc., May 2006.
4. S. Hadjiefthymiades, S. Papayiannis, and L. Merakos, Using path prediction to improve TCP performance in wireless/mobile communications, *IEEE Communications Magazine*, Vol. 40, No. 8, 2002.
5. G. Liu and G. Maguire Jr., A class of mobile motion prediction algorithms for wireless mobile computing and communications. *MONET* Vol. 1, pp. 113–121, 1996.
6. A. Bhattacharya and S. Das, LeZi update: an information theoretic approach to track mobile users in PCS networks. *Proceedings of ACM/IEEE Mobicom*, 1999.
7. F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli, *Trajectory Pattern Mining, KDD Intl. Conf.*, 2007.
8. S. Choi and K. G. Shin, Predictive and adaptive bandwidth reservation for hand-offs in QoS-sensitive cellular networks, *ACM SIGCOMM*, 1998.
9. N. Samaan and A. Karmouch, A mobility prediction architecture based on contextual knowledge and spatial conceptual maps, *IEEE Transactions on Mobile Computing*, Vol. 4, No. 6, 2005.
10. R. Viayan and J. Holtman, A model for analyzing handoff algorithms, *IEEE Transactions on Vehicular Technology*, Vol. 42, No. 3, 1993.
11. G. Yavas, D. Katsaros, O.Ulusoy, and Y. Manolopoulos, A data mining approach for location prediction in mobile environments, *Data and Knowledge Engineering*, Vol. 54, No. 2, 2005.

12. D. Katsaros, A. Nanopoulos, M. Karakaya, G. Yavas, O. Ulusoy, and Y. Manolopoulos, Clustering mobile trajectories for resource allocation in mobile environments, proceedings IDA, pp. 319–329, 2003.
13. M. Kyriakakos, S. Hadjiefthymiades, N. Fragkiadakis, and L. Merakos, Enhanced path prediction for network resource management in wireless LANs, *IEEE Wireless Communications*, Vol. 10, No. 6, 2003.
14. V. T. H. Nhan and K. H. Ryu, *Future Location Prediction of Moving Objects Based on Movement Rules*, Springer ICIC 2006, LNCIS 344, pp. 875–881.
15. Y. Xiao, H. Zhang, and H. Wang, Location prediction for tracking moving objects based on grey theory, *IEEE FSKD*, 2007.
16. H. Jeung, Q. Liu, H. Tao Shen, and X. Zhou, A hybrid prediction model for moving objects, *IEEE 24th International Conference on Data Engineering*, pp. 70–79, 2008.
17. T. Anagnostopoulos, C. Anagnostopoulos, S. Hadjiefthymiades, A. Kalousis, and M. Kyriakakos, *Path Prediction through Data Mining, Proceedings of International Conference on Pervasive Services (ICPS)*, Istanbul, Turkey, 2007.
18. T. Anagnostopoulos, C. Anagnostopoulos, S. Hadjiefthymiades, M. Kyriakakos, and A. Kalousis, Predicting the location of mobile users: a machine learning approach, *Proceedings of International Conference on Pervasive Services (ICPS)*, London, UK, 2009.
19. S. Akoush and A. Sameh, Mobile user movement prediction using Bayesian learning for neural networks, *ACM IWCMC*, August 2007.
20. I. Burbey and T. L. Martin, Predicting future locations using prediction-by-partial-match, *ACM MELT*, September 2008.
21. H. Jeung, M. L. Yiu, X. Zhou, and C. S. Jensen, Path prediction and predictive range querying in road network databases, *VLDB*, Vol. 19, pp. 585–602, 2010.
22. T. Anagnostopoulos, C. Anagnostopoulos, and S. Hadjiefthymiades, An adaptive location prediction model based on fuzzy control, *Computer Communications*, Sept. 2010. doi:10.1016/j.comcom.2010.09.001.
23. E. Alpaydin, *Introduction to Machine Learning*, The MIT Press, 2004.
24. R. Duda , P. Hart , and D. Stork, *Pattern Classification*, Wiley-Interscience, 2001.
25. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning; Data Mining, Inference and Prediction*. Springer, New York, 2001.
26. Quinlan, C4.5: Programs for Machine Learning, MK Series in Machine Learning, 1993.
27. J. Killer, M. Hatef, R. Duin, and J. Matas, On combining classifiers, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 3, 1998.
28. T. S. Rappaport, *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.
29. M. Kyriakakos, N. Frangiadakis, S. Hadjiefthymiades, and L. Merakos, RMPG: a realistic mobility pattern generator for the performance assessment of mobility functions. *Simulation Modeling Practice and Theory*, Vol. 12, No. 1, 2004.
30. M. Weiss and C. Kulikowski, Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Networks, Machine Learning and Expert Systems, MK in Machine Learning, 1991.
31. M. Plutowski, S. Sakata, and H. White, Cross-validation estimates integrated mean squared error, *Advances in Neural Information Processing Systems*, 6, 1994.
32. T. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, *Neural Computation*, Vol. 10, pp. 1895–1923, 1998.
33. D. Lopresti and A. Tomkins, Block edit models for approximate string matching, *Theoretical Computer Science*, Vol. 181, No. 1, 1997.
34. I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tool sand Techniques*, Morgan Kaufmann Series in Data Management Systems, 2005.
35. Web Site: http://reality.media.mit.edu/, visited on 30 May, 2009.
36. P. Hui and J. Crowcroft, Human mobility models and opportunistic communications system design, *Proceedings of the Royal Society A* Vol. 366, No. 1872, pp. 2005–2016, 2008.
37. T. Liu, P. Bahl, and I. Chlamtac, Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks. *IEEE Journal on Selected Areas in Communications*, Vol. 16, No. 6, 1998.
38. J.-M. Francois, G. Leduc, and S. Martin, Learning movement patterns in mobile networks: a generic method, *Proceedings of European Wireless*, pp.128–134, 2004.
39. A. Aljadhai and T. F.Znati, Predictive mobility support for QoS provisioning in mobile wireless environments, *IEEE JSAC*, Vol. 19, No. 10, 2001.
40. M. Poulakis, V. Vassaki, and S. Hadjiefthymiades, Proactive radio resource management using optimal stopping theory, *Proceedings of WoWMoM 2009, 10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Kos, Greece, June 2009.

Author Biographies

Theodoros Anagnostopoulos received his B.Sc. in Informatics from the Department of Informatics at the Technical Educational Institution of Athens, Athens, Greece in 1997. He also graduated the Department of Informatics at the Athens University of Economics and Business (AUEB), Athens, Greece in 2001. He was awarded a M.Sc. in Information Systems from the same Department in 2002. Currently, he is a Ph.D. candidate in the National and Kapodistrian University of Athens (NKUA), Department of Informatics and Telecommunications. His research interests are in the areas of Pervasive Computing and Machine Learning. He is a member of the Pervasive Computing Research Group of NKUA.



Christos Anagnostopoulos his B.Sc., M.Sc. and Ph.D. in Informatics and Telecommunications from the Department of Informatics & Telecommunications (DIT) of the University of Athens (UoA), Athens, Greece. Since the beginning of 2011, he belongs to the faculty of the Ionian University, Department of Informatics, Corfu, Greece, where he presently is an assistant professor. His research interest is focused on mobile and distributed computing systems and context-aware computing. He had also participated in projects realized in the context of EU programs.

Stathes Hadjiefthymiades received his B.Sc., M.Sc. and Ph.D. in Informatics and Telecommunications from the Department of Informatics & Telecommunications (DIT) of the University of Athens (UoA), Athens, Greece. He also received a joint engineering-economics M.Sc. degree from the National Technical University of Athens. In 1992 he joined the Greek consulting firm Advanced Services Group, Ltd., as an analyst/developer of telematic applications and systems. In 1995 he became a member of the Communication Networks Laboratory of UoA. During the period 2001–2002, he served as a visiting assistant professor at the University of Aegean, Department of Information and Communication Systems Engineering. In 2002 he joined the faculty of the Hellenic Open University (Department of Informatics), Patras, Greece, as an assistant professor. Since the beginning of 2004, he belongs to the faculty of UoA, DIT where he presently is an assistant professor. He has participated in numerous projects realized in the context of EU programmes and national initiatives. His research interests are in the areas of mobile, pervasive computing, web systems engineering, and networked multimedia applications. He is the author of over 150 publications in these areas.