Path Prediction through Data Mining

Theodoros Anagnostopoulos[1], Christos Anagnostopoulos[1], Stathes Hadjiefthymiades[1], Alexandros Kalousis[2], Miltos Kyriakakos[1]

[1]Pervasive Computing Research Group, Communication Networks Laboratory, Department of Informatics and Telecommunications, University of Athens, Panepistimiopolis, Ilissia, Athens 15784, Greece, tel: +302107275127, e-mail: {thanag, bleu, shadj, miltos}@di.uoa.gr

[2]Artificial Intelligence Laboratory, Department of Computer Science, University of Geneva, Uni-Dufour, Geneva 1211, Switzerland, tel: +41223797630, e-mail: Alexandros.Kalousis@cui.unige.ch

Abstract - **Context-awareness is viewed as one of the most important aspects in the emerging ubiquitous computing paradigm. Mobile applications are required to operate in pervasive computing environments of dynamic nature. Such applications predict the appropriate context in their environment in order to act efficiently. A context model, which deals with the location prediction of moving users, is proposed. Such model is used for trajectory classification through Machine Learning techniques. Hence, spatial and spatiotemporal context prediction is regarded as context classification based on supervised learning. Finally, two classification schemes are presented, evaluated and compared with other ML schemes in order to support location prediction and decision-making.**

Key words – machine learning, data mining, location prediction

I. INTRODUCTION

In order to render applications and services intelligent enough to support modern users everywhere / anytime and materialize the so-called ambient intelligence, information on the present context of the user has to be captured and processed accordingly. Such information may refer to the user's position, time, physical properties like temperature or other general parameters (e.g., the specific devices that the user carries, application context). The efficient management of contextual information requires detailed and thorough data modeling along with specific processing, reasoning and prediction capabilities.

A computing environment, which is based on such pervasive infrastructure, is called Pervasive Computing Environment (PCE). In a PCE, diverse contexts can appear (e.g., user is in his/her office, walking outdoors, driving a car). However, context - awareness allows an *entity* to adapt to its environment thus offering a number of advantages and possibilities for new applications. One of the more intuitive capabilities of such applications is their *proactivity*. Predicting user actions and contextual parameters enables a new class of applications to be developed. Spatiotemporal prediction can be used to improve resource reservation in wireless networks and facilitates the possibility of providing desired location based services by preparing and feeding them with the appropriate context in advance [22]. Predictive context-aware applications can perform context pre-evaluation aspects introducing innovative proactive services (e.g., alerts related to traffic conditions, certain information pre-fetching and triggering actuation rules in advance).

The concept of predicting spatial context with Machine Learning (ML) algorithms and techniques is quite novel. ML, in its broadest sense, can be considered as the study of algorithms that improve automatically through experience. It has a wide spectrum of applications including search engines and medical diagnosis. One of the most typical tasks in machine learning is the discovery of patterns from large datasets. The Data Mining (DM) sub-field of ML handles such data discovery. ML tasks can be roughly organized into: supervised learning (e.g., classification and regression) and unsupervised learning (e.g., clustering). In this paper, we exploit classification in order to predict contextual information. In classification, training data are most often represented as attribute-value vectors, though other complex structures can be also considered (e.g., sets and graphs). Each training data is assigned to a specific class from a fixed set of classes (e.g., symbolic locations). The goal of the classification task is the prediction of the class of a given *unseen* data.

We propose a context model that deals with location prediction of moving users. The proposed model predicts the *next* movement of a mobile user with a certain *moving profile* and history of movements. Furthermore, temporal context (e.g., morning, noon, afternoon and night) is also incorporated in the proposed model. Hence, the representation of a user can include both spatial and temporal information. Two classification schemes are introduced in order to support location prediction. These schemes are evaluated with three DM algorithms. Finally, the classification schemes are also compared with three non-DM schemes for location prediction, by means of prediction accuracy.

The paper is organized as follows: Section II reports certain supervised learning schemes used for context classification while, in Section III a context representation model taking into account the current and historical user context is proposed. Section IV discusses the application of the proposed model in a PCE using specific ML classification schemes. In Section V, we evaluate the discussed context model with both DM and non-DM classifiers while, Section VI reports prior work on that research area. Finally Section VII concludes the paper.

II. Machine Learning and Classification

Classification is the task of learning from examples, which are described by a set of attributes and a class attribute. The result of learning is a classification model that is capable of accurately predicting the value of the class attribute of unseen examples based only on the values of the attributes. For instance, a *user* is highly likely jogging if the following conjunction of certain attributes holds true at a specific time *t*, that is, "IF location(*user*, *t*) IS outdoors AND speed(*user*, *t*) IS high AND respiratory-level(*user*, *t*) IS high THEN context(*user*, *t*) IS highly likely jogging". Specifically, a training set is fed to a classification algorithm and the result is a *classification model*. This *classification model* can then be applied to new unseen and unlabeled instances in order to predict their class labels. A crucial element in the whole process is the quality of the predictions that the classification model produces. In order to estimate this quality usually a test phase is also adopted. One of the most popular measures of classification performance is accuracy, which is the percentage of correctly predicted instances from a test set.

Various algorithms exist for the task of classification. Some indicative algorithms are the following [1]: inferring rudimentary rules (e.g., 1R), statistical learning (e.g., naïve Bayes, support vector machines), decision tree induction (e.g., C4.5), classification rules (e.g., Ripper), instance-based learning (e.g., k-nearest neighbors) and nonlinear models (e.g., multi-layer perceptrons). Obviously, there is no single classification algorithm that works best independently of the application domain and problem. We have experimented with three different classification algorithms, which were initially selected for their good alignment with the domain assumptions (i.e., spatiotemporal context). Finally, the decision tree induction algorithm was selected for our experiments since it demonstrates satisfactory performance in the discussed domain.

A. Bayesian algorithms

Bayesian classification algorithms are statistical learning algorithms based on the Bayes theorem [8]. The Naïve Bayes algorithm, which is a simple Bayesian classifier, demonstrates a comparable performance with decision trees and neural network algorithms [2]. It assumes that, the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is known as the "class conditional independence".

B. Decision-Tree-Based algorithms

A decision tree consists of internal decision nodes and leaves. Each node corresponds to a test function on a given attribute of the learning examples, each of the different possible outcomes of the test function leads to a different branch of the decision tree. Given an example that arrives to a specific node the corresponding test is applied on the example and the example is sorted to the appropriate branch according to the result of the test. The process starts at the root of the tree and is repeated recursively until a leaf node is reached. At that point the class label predicted is the label associated with the specific tree. A leaf node defines a region in the input space instances falling within that region assume the same class label. The hierarchical placement of decisions allows a fast localization of the region in which a specific example belongs, [3]. Decision Tree learners (C4.5 is a well known representative classifier [4]) use heuristic hill climbing, employing heuristics such as Information Gain, Information Gain Ratio and Gini Index, to select the most appropriate test at each decision node The result of a decision-tree-based algorithms is a decision tree, which can be easily transformed into a set of classification rules, [4].

C. Rule induction for classification

Rule-induction behaves similar to tree-induction. Actually, rule-induction performs a depth-first search, in the data graph and generates one path (represented as a classification rule) [3]. Rules are constructed one at a time. Each rule is a conjunction of conditions on discrete or numeric attributes and such conditions optimize some criterion (e.g., minimize entropy). A rule is said to "cover" an example if that example fulfils all the conditions of the rule. Once a rule is constructed it is asserted to the rule base. A representative rule-induction algorithm, that we experiment with, is the RIPPER [6], which stands for Repeated Incremental Pruning to Produce Error Reduction.

III. Context Representation

The contextual information considered for location prediction is the spatial and temporal context of the user. Such contexts refer to the antecedent-part of the classification rules while the consequent-part of these rules is the spatial context. The spatial context refers to (i) the current user location, (ii) the history of user movements (represented by a vector of time-ordered transitions between locations) and (iii) the spatial representation into cells and clusters of cells. The temporal context refers to (i) the user residence time in a location, (ii) the time stamp of a transition occurrence and (iii) the taxonomy of user trajectories in specified time slots.

The discussed model defines the user moving space i.e., current location, as the basic context attribute for classification. The user roams through a cellular network thus, the network cell represents a location. The model also represents the spatial context as a cluster of neighboring cells resulting to a hierarchical spatial context representation. Various categories for clustering n cells can be used, $n > 0$. Each cell belonging to a cluster has its unique identifier, cell-id, while the unique identifier of the cluster, cluster-id, is that of its central cell.

Moreover, the model uses temporal context and examines the possibly enhancement in the prediction process. Such context refers to the occurrence time of a transition between locations (cells or clusters). The time duration of the day is split in four slots: morning, noon, afternoon and night. Each time slot relates to a specific time interval [*a*, *b*], where *a* and *b* represent the start- and the end-time stamps of the slot, respectively (e.g., the morning slot relates to the [9a.m., 12a.m.] time interval). The residence time of a user in each visited cell [5] is accumulated and categorized into specific time intervals.

A. Context Model for Classification

The proposed model uses the user historical context in order to predict the future location based on two classification schemes: the cell-based classification scheme (C_A) and the cluster-based classification scheme (C_B). Such schemes are further categorized to those that deal with spatial context and those that combine both spatial and temporal (spatiotemporal) contexts. Generally speaking, C_A and C_B use cell-ids and cluster-ids for the class-values in the classification process, respectively.

Let A_u be the set of cells that a user *u* has visited. Such set defines the user movement space. Let also *B* be the set of the defined clusters in the user movement space such that each cell maps to a cluster. For the *u* user there is a function, f_u, which associates each cell $a \in A_u$ with its cluster $b \in B$ i.e., $f_u : A_u \rightarrow B$ (e.g., $b = f_u(a)$ is the cluster of the visited cell *a*). A cell-transition from a cell $a \in A_u$ to a cell $c \in A_u$ is defined as the movement of the *u* user from the location (start) to the location (end), respectively and is denoted as $a \rightarrow c$. The cluster-transition is similarly defined i.e., $p \rightarrow q$, with $p, q \in B$ and for each $a \in A_u$, there exists $b \in A_u$: $p = f_u(a)$ and $q = f_u(b)$.

Definition: A *trajectory of movements* is defined as the ordered vector *Q* of *n* cell transitions observed at specific time stamps $ts(a_i \rightarrow a_{i+1})$, $i = 1, ..., n$, that is (represented in a matrix format):

$$Q(n) = [(a_1 \rightarrow a_2)\,(a_2 \rightarrow a_3)\,\dots\,(a_n \rightarrow a_{n+1})] \qquad (1)$$

The ordering of the transitions of a trajectory *Q* is obtained by the ordering of their corresponding time stamps of observation i.e., $ts(a_{i-1} \rightarrow a_i) < ts(a_i \rightarrow a_{i+1})$, $i = 1, ..., n$. The last column of the vector is the current transition. Updating *Q* for the upcoming transition at $ts(a_{n+1} \rightarrow a_{n+2})$ is repeatedly done in a straightforward manner:

$$Q(n+1) = [(a_2 \rightarrow a_3)\dots\,(a_n \rightarrow a_{n+1})\,(a_{n+1} \rightarrow a_{n+2} \qquad (2)$$

The oldest transition $(a_1 \rightarrow a_2)$ is discarded and archived in the *training set* for the classification process.

A *trajectory window* or *window* of a *Q* trajectory is the number of cell-transitions in *Q* i.e., *n*, and the duration of a window is $ts(a_1 \rightarrow a_2)$ - $ts(a_n \rightarrow a_{n+1})$. *Q* is mapped to a predefined time slot according to the value of the corresponding duration.

Let $Attr = \{r_1, r_2, \dots, r_m, r_t\}$ be a set of *m* spatial context attributes r_i, $i = 1, ..., m$, $m>0$, and r_t is the temporal context attribute. The domain of each attribute r_i, $i = 1, ..., m$, is the set of cells A_u of the user movement space and the domain of the r_t attribute is the set of time slots. It should be noted that, the incorporation of the r_t attribute in the set *Attr* refers to spatiotemporal classification thus in case of the spatial classification the set of attributes is $Attr \setminus \{r_t\}$. It is worth noting that, each pair of neighbouring attributes (r_k, r_{k+1}) with $r_k, r_{k+1} \in Attr \setminus \{r_t\}$, $k = 1, ..., m-1$, corresponds to the transition $(a_k \rightarrow a_{k+1})$. Let $Cttr = \{c\}$ be the singleton of the class-attribute, where the domain of the *c* attribute is the set of cells A_u or clusters *B* in the case C_A and C_B schemes, respectively. The value of the *c* attribute is the end location (the a_{m+1} cell or the $f_u(a_{m+1})$ cluster) of the last transition $(a_m \rightarrow a_{m+1})$ of a trajectory. Furthermore, a training tuple is represented by a trajectory of *m* window length for spatial and spatiotemporal schemes. In the latter case, the training tuple contains also a temporal value, r_t, that depicts the time slot in which the corresponding trajectory is observed.

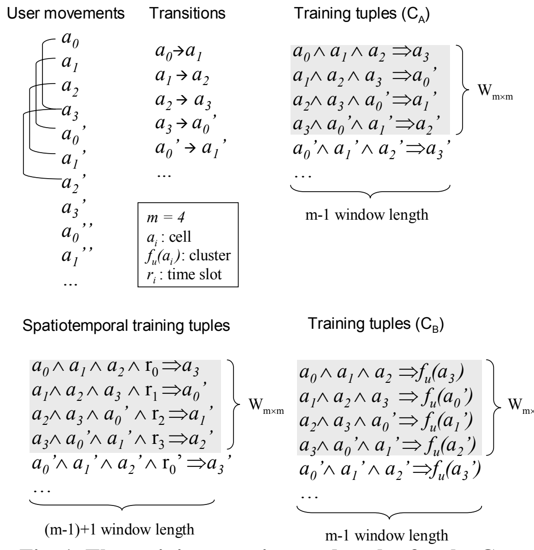

Fig. 1. The training-matrices and tuples for the C_A and C_B, with window length 3 (m = 4)

A training-matrix $M_{m\times m} = \{Q_i, i = 1, \dots, m\}$ is the set of *m* training tuples (historical trajectories) of *m*-1 window length (Fig. 1). In a $M_{m\times m}$, the kth transition of the ith trajectory is the (*k*-1)th transition of the (*i*+1)th trajectory, *i*, *k* = 1,...,*m*. The value of the start location for the last transition of the (*i*+1)th trajectory is either the value r_m of the class-attribute of the ith trajectory (in case of the C_A) or the $f_u^{-1}(r_m)$ cell (in case of the C_B). The $f_u^{-1}(b)$ corresponds to a cell that belongs to the cluster $b \in B$. Hence, the classification rule for the ith trajectory, $i = 1, ..., m$ is defined as follows:

$$(r_1 = a_{1i}) \wedge \dots \wedge (r_{m-1} = a_{(m-1)i}) \Rightarrow (c = r_m) \qquad (3)$$

a_{ji} is the jth visited cell in the ith trajectory. The trajectories in a matrix $M_{m\times m}$ are sequentially overlapped. This means that, the end-location of the last transition of the first trajectory becomes the start-location of the first transition of the last trajectory, forming a *shift* of transitions, as illustrated in Fig. 1.

Hence, the values of the class-attribute in the i^{th} trajectory determine the values of that attribute in the $(m+i)^{\text{th}}$ trajectory. The classifier is fed with N matrices of m (or $m+1$ in case of the temporal attribute) attributes thus with $N{\cdot}m$ trajectories.

IV. Context Model Application

The trajectories of the users are categorized into different groups according to the *degree of movement randomness*. We have distinguished five groups corresponding to five different values of such degree. The considered values of randomness are: 0.0, 0.25, 0.5, 0.75 and 1.0. A low degree of randomness indicates a deterministic movement (e.g., an ordinary moving pattern). A high value of randomness implies the opposite. Randomness is the main parameter for demonstrating the performance of the model under various conditions of uncertainty in the user mobility behavior. This means that, the performance of the classifier is influenced by the lack of regular moving patterns. Hence, the deterministic trajectories (i.e., those with a low value of randomness) are further specialized into regular trajectories. Regular trajectories represent user movements during a predefined period of time (e.g., a daily schedule from home to work). On the other hand, random trajectories represent arbitrary user movements (e.g., a quick coffee-stop before going to work).

A. Context Model Behavior

The C_A and C_B schemes predict the *next* cell and the *next* cluster of the user movement, respectively. The next location (cell or cluster) denotes the predicted user location. Actually, there can be a high possibility that, the next cell a_{pr} (determined by C_A) might belong to the next cluster b_{pr} (determined by C_B) i.e., $f_u(a_{\text{pr}}) = b_{\text{pr}}$. Let $E_n = \{a_{pr,n} \in A_u \mid f_u(a_{pr,n}) = b_{pr,n}, b_{pr,n} \subseteq B_{\text{pr}}\}$ be the set of the predicted cells $a_{\text{pr},n}$ that belong to the predicted clusters $b_{\text{pr},n}$ after n *runs*, $n > 0$. For each *run* the two classifiers use a set of M training trajectories of m-1 window length, M, m>0. Then, the *coherence* metric H_n in [8] between such classifiers indicates how much their predictions are *alike* for a specific user. *Alike* predictions means that, the two classifiers point to analogous results / locations. In other words, C_A predicts a cell that belongs to the same cluster as that determined by C_B. Once a_{pr} belongs to b_{pr} then, the location determination by the model is reasonable and not contradictory.

$$H_n = \frac{1}{n \cdot N}\sum_{N} |E_n|, \qquad (4)$$

for N training matrices $M_{m\times m}$

Fig. 2 depicts the behavior of H_n, $n = 1000$, against user randomness. H_n assumes high and stable values (approx. 96%) as the randomness increases indicating that the two schemes result in reasonable (i.e., not contradictory) prediction results. In particular, a high value of such metric depicts that, the user is predicted to be in a cluster determined by C_B and besides that in a specific cell within that cluster determined by C_A with a high degree of belief. A high value of the coherence metric enables the model to be more certain on decision making since the two schemes support equivalent pieces of evidence.

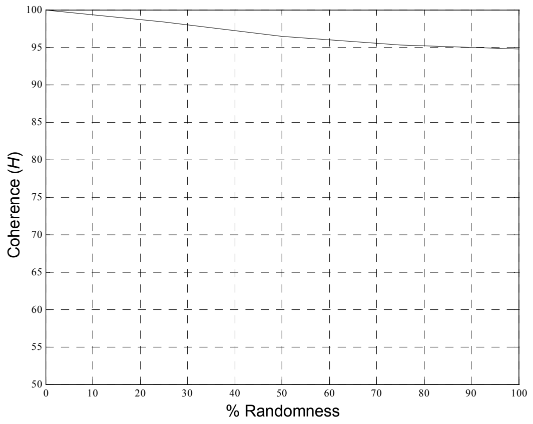

Fig. 2. Coherence metric for C_A and C_B

B. Decision Making

A classifier predicts the next cell $a_{\text{pr}} \in A_u$ (or cluster $b_{\text{pr}} \in B$) that maximizes the corresponding probability of occurrence given N training matrices, as follows:

$$\begin{aligned} a_{pr} &= \arg\max_{a \in A_u} P_{C_A}(a \mid \{M_{m\times m}\}_N) \\ b_{pr} &= \arg\max_{b \in B} P_{C_B}(b \mid \{M_{m\times m}\}_N) \end{aligned} \qquad (5)$$

where, P_{CA} and P_{CB} are the probability distributions of the class-values for the C_A and C_B schemes, respectively. However, the decision d_{pr} for the predicted user location can be modeled as the vector $d_{\text{pr}} = (a_{\text{pr}}, b_{\text{pr}})$ i.e., the *next* b_{pr} cluster and the *next* a_{pr} cell within that cluster. Such decision has to take into consideration the coherence metric after the application of C_A and C_B. A decision based only on the predicted cluster b_{pr} (i.e., applying only C_B) denotes that, one is certain (with a degree of $P_{CB}(b_{\text{pr}}) \in (0,1]$) that the user is predicted to be in a cell that belongs to the b_{pr} cluster but, has no knowledge about which cell within such cluster. In this case, $P_{C_A}(a) = \frac{1}{k}, \textit{for each } a : f_u(a) = b_{pr}$, where k is the number of cells in a cluster. The latter probability denotes the total ignorance of a user being in a cell belonging to the predicted b_{pr} cluster. Instead, the coherence between the two classification schemes can be exploited to infer the next cell within the predicted cluster. Consider the following two pieces of evidence H_1 and H_2 (Table I). H_1 is observed once one has applied the two schemes and the corresponding H metric has assumed values over a given threshold h. Then, the decision vector d_{pr} is defined in Table I. Upon a high value of h i.e., the two schemes deliver fully compatible prediction results, a_{pr} is selected since it belongs to the predicted cluster b_{pr}. In such decision, one can imply that $a_{\text{pr}} \in E_n$. The H_2 evidence is observed when the H metric assumes values below a given threshold h. This means that, C_A and C_B

predict a cell a_{pr} and a cluster b_{pr}, respectively with $f_u(a_{pr}) \neq b_{pr}$. In that case, two further heuristic decisions H_{21} and H_{22} are possible (Table I). H_{21} chooses as the next cell, the geometrically central cell g_{pr} of the predicted b_{pr} (i.e., $f_u(g_{pr}) = b_{pr}$) once the $f_u(a_{pr})$ is a neighbor cluster of the b_{pr} cluster. Such decision is preferred when a cluster consists of a small number of cells. The second decision results to a cell $c_{pr} : f_u(a_{pr}) = b_{pr}$ such that c_{pr} is the nearest cell (according to a measurable distance D) to the a_{pr}.

Table I. Decision Making

$if\left(\tilde{H} \geq h\right) then\ d_{pr} = (a_{pr}, b_{pr})\ where$ $a_{pr} = arg \max\limits_{a_i : f_u(a_i) = b_{pr}} P_{C_A}(a_i)\ and$ $b_{pr} = arg \max\limits_{b_i \in B} P_{C_B}(b_i)$	(H1)
$else$ $H_{21}\ case: if\left(f_u(a_{pr})\ is\ neighbor\ of\ b_{pr}\right) then$ $d_{pr} = (g_{pr}, b_{pr})\ where$ $g_{pr} = \mathrm{centralof}(b_{pr})\ and$ $b_{pr} = \arg\max\limits_{b_i \in B} P_{C_B}(b_i)$	(H21)
$H_{22}\ case: if\left(f_u(c_{pr}) = arg \max\limits_{b_i \in B} P_{C_B}(b_i)\right) then$ $d_{pr} = (c_{pr}, b_{pr})\ where$ $c_{pr} = arg \min\limits_{\substack{f_u(a_i) = b_{pr} \\ f_u(a_j) \neq b_{pr}}} D(a_i, a_j)$	(H22)

Notice: Evidently, the decisions based on C_B assume better prediction accuracy (the percentage of the correct predicted locations) of C_A. The predicted cluster is that resulting from the C_B and the predicted cell is chosen according to the discussed decision-making. In fact, C_B is used for supporting the perdition result determined by C_A.

V. Experimental Evaluation

A. Experiment Setup

The Weka machine-learning workbench [1] is used for our experiments. Weka is a collection of machine learning algorithms and data preprocessing tools for data mining. In order to evaluate the performance of the proposed model with the two schemes, the user trajectories are represented as a series of waypoints. Each waypoint is defined by the location in terms of a cell-id, time of day and speed. Specifically, a scenario in which users moved around a set of predefined locations was used. Such locations, derived from [9], are "home", "work", "restaurant", "coffee" and "movies". The considered regular pattern is "home $\rightarrow$ work $\rightarrow$ restaurant $\rightarrow$ coffee $\rightarrow$ movies $\rightarrow$ home". The number of cells is 100 and the number of clusters is 21.

We have enriched the user movement by a random waypoint algorithm [9], in which five discrete categories of randomness are used, i.e., from the regular pattern (%0 randomness with 500 training instances) to completely disordered trajectories (100% randomness with 1000 training instances). It should be noted that, the distribution assumed for cell *residence times*, as discussed in [7] is the Generalized Gamma Distribution (GGD) with probability density function G defined in (6). GGD is considered the best fit for cell residence times [5].

$$G(t;a,b,c) = \frac{c}{b^{ac}\Gamma(a)} t^{ac-1} e^{-\left(\frac{t}{b}\right)^{c}}, \quad t,a,b,c > 0 \qquad (6)$$

$\Gamma(\cdot)$ denotes the Gamma function. The (a, b, c) = (2.31, 1.22R, 1.72), where R is the cell radius and the average user speed is 50 km/h. The model exploits such distribution in order to map the user residence times (in cells) into the predefined time intervals. The time of day was divided in four time slots, e.g., morning – [09:00– 12:00], noon – [12:01-15:00], afternoon – [15:01–18:00] and night – [18:01 – 21:00]. Furthermore, to simplify the simulation, we assumed that the users moved with an average speed of 50 km/h. Hence, the collected way point' times were normalized to imitate a constant speed movement.

B. Prediction Accuracy for C_A and C_B schemes

We experiment with three classifiers with different characteristics, the Naïve Bayes learner, the J48 Decision Tree learner, which is an implementation of C4.5 algorithm, and the JRip Classification Rule learner, which is an implementation of Ripper in order to train (learn) and test our model. The J48 classifier demonstrates the best prediction accuracy (Fig. 5) thus we use it for our further experiments. Furthermore, if n is the number of instances (i.e., training tuples representing trajectories) and p the number of attributes (i.e., the trajectory window length) then, the time computational cost of the J48 algorithm is $O(n \cdot p \cdot \log n)$ [4]. The training phase was completed with two weeks of trajectories observation. The trajectory window length is set to 3, i.e., m = 4, and each cluster contains seven cells. Finally, the coherence threshold h for the decision-making is set to 0.96. The accuracy measure used for the application of each scheme on the J48 classifier is the 10-fold cross-validation. Cross-validation is a method for estimating the generalization error based on re-sampling [8,10]. Specifically, in p-fold cross-validation, the training data, i.e., training tuples, are divided into M, $M > 0$, subsets of approximately equal size. Hence, the classifier is trained p times, each time leaving out one of the subsets from the training set but, uses only the omitted subset to compute the errors of each scheme (C_A and C_B).

It can be observed from the above experiments (Table II and Table III) that, spatiotemporal context slightly improves the prediction accuracy of each scheme for 0%, 25% and 50% of degree of randomness, but slightly worse prediction accuracy for 75% and 100% degree of randomness, thus we should not consider it for further experimentation. Finally, the prediction accuracy of the two schemes for spatial simulated data is illustrated in Fig. 3. Specifically, the two classifiers are built for two weeks of training data and tested within five working days, (from Monday to Friday). The time duration of the entire set of trajectories is 12 hours.

Table II. Accuracy of C_A based on 10-fold cross-validation

Randomness	Spatial Context	Spatiotemporal Context
0%	95.32	95.64
25%	71.25	71.33
50%	45.43	45.87
75%	39.12	38.93
100%	33.29	32.06

Table III. Accuracy of C_B based on 10-fold cross-validation

Randomness	Spatial Context	Spatiotemporal Context
0%	98.63	98.97
25%	80.33	80.42
50%	61.56	61.91
75%	58.64	57.92
100%	55.17	54.02

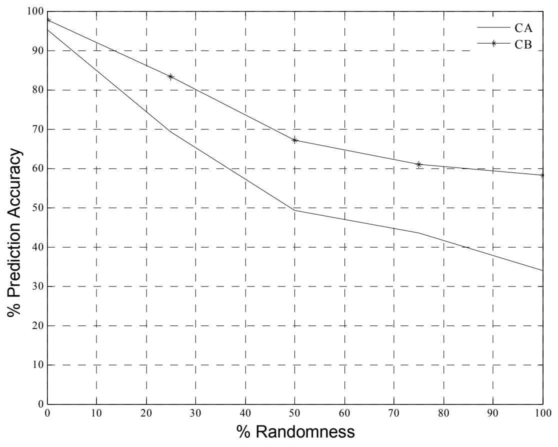

Fig. 3. Comparison of the prediction accuracy between C_A and C_B

C. Comparison with DM classifiers

Fig. 5 depicts the prediction accuracy between J48 and the Naïve Bayes and JRip classifiers. Obviously, J48 assumes better performance both in prediction accuracy and time complexity than the considered DM classifiers. For that reason, we select J48 algorithm in order to implement the C_A scheme.

D. Comparison with non-DM classifiers

We compare C_A scheme with LA [11], MMP [12] and HLP [13] schemas by means of prediction accuracy. Specifically, a Learning Automaton (LA) is based on a state transition matrix, which comprises the one-step state transition probabilities and follows a Linear Reward-Penalty (L_{R-P}) scheme. If the LA decision is correct a positive feedback is received from the environment and the probability of the respective state transition is increased ("rewarded"). The rest of the probabilities are evenly reduced ("penalized") in order to balance the increase. If the response is wrong the state transition is "penalized" and the rest of the transitions are "rewarded" accordingly. The path prediction algorithm in [11] uses LA and exploits the spatial and temporal contextual information. It could be derived from the comparison in Fig. 4. that, for the same training set (a two-week period) C_A demonstrates better prediction accuracy against LA.

The Mobile Motion Prediction (MMP) [12] algorithm consists of the "regularity-pattern detection" and the "model prediction" processes. The former process decomposes the complicated daily movement into the regular pattern part and the random movement part. The latter process detects itinerary-patterns of the user movement. Such patterns are used to predict the next user movement.

Moreover, the Hierarchical Location Prediction (HLP) [13] algorithm is based on random (pseudo-stochastic) movement model, which integrates deterministic behavior with randomness in an attempt to mimic actual human behavior. HLP comprises an approximate pattern-matching algorithm that extracts regular movement pattern to estimate the global inter-cell direction. It also uses the extended self-learning Kalman-filter that deals with "unclassifiable" random movements by tracking intra-cell trajectory and predicting the next-cell crossing. In Fig. 6 the prediction accuracy of the C_A with that of the non-DM classifiers is illustrated. Specifically, for a value of 50% randomness C_A assumes the worst prediction accuracy out of all classifiers, while for a value of 75% randomness C_A has similar prediction accuracy to that of the MMP classifier. Finally, for a completely random movement (100% randomness) C_A assumes better performance than MMP.

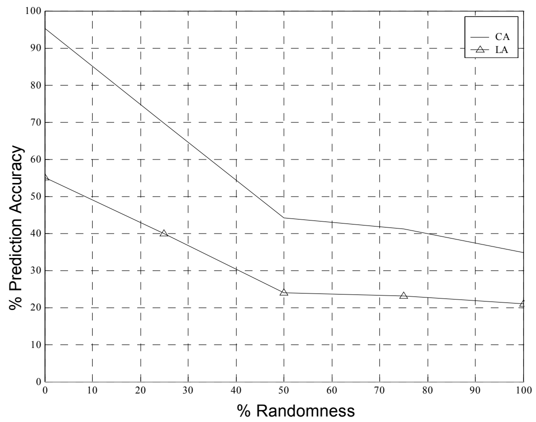

Fig. 4. Comparison of the prediction accuracy between C_A and LA

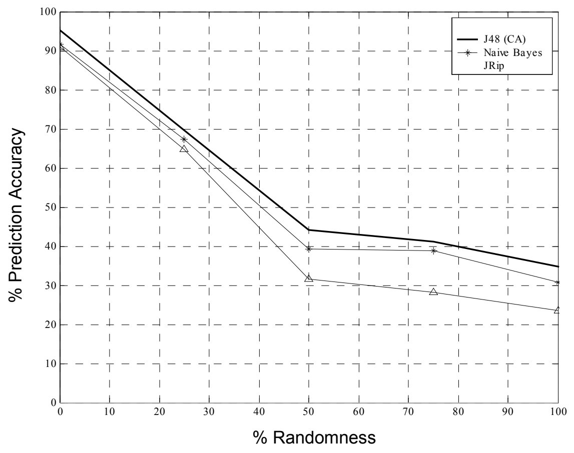

Fig. 5. Comparison of the prediction among DM classifiers

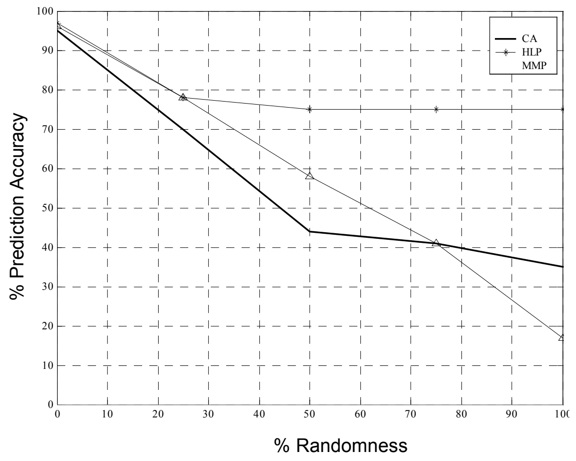

Fig. 6. Comparison of the prediction accuracy between C_A and non-DM classifiers

VI. Related Work

There are a lot of prediction models based on ML techniques. Specifically, the probabilistic model in [14] is based on the user movement history of handover behavior. Such model considers the history of all handovers that occurred in a given cell using Naïve Bayes classification. The authors in [15] report a probability-based and a learning-based model for trajectory prediction. The algorithm in [16] predicts the next inter-cell movement of a mobile user in a PCE. Actually, the user mobility patterns are mined from the history of the trajectories resulting in mobility rules extraction. The location prediction is based on such rules.

The authors in [17] implement an algorithm based on user mobility patterns discovery. Such patterns, which are derived from trajectory clustering, are used for location prediction and dynamic resource allocation. Moreover, an efficient online (incremental) algorithm that learns *routes* between *important* locations and predicts the future location is reported in [18]. Specifically, clusters of cell sequences are built to represent *physical routes*. Furthermore, the prediction is based on destination probabilities and temporal reasoning. A data-mining algorithm is proposed in [19], which efficiently discovers sequential mobile patterns. Such patterns exploit both spatial and application context (e.g., service requests). Moreover, the mobility tracking in a cellular network is based on information theory by using the compression Lempel-Ziv algorithm [20]. The algorithm [12] consists of the "regularity-pattern detection" and the "model prediction" processes used for predicting the user movements (Section V.*D*). The work presented in [13] discusses pattern matching techniques and extended, self-learning, Kalman filters in order to estimate the future location. In addition, a learning automaton that follows a linear reward-penalty scheme is used in [11] to facilitate user location prediction. Finally, the authors in [21] apply evidential reasoning based on the Dempster-Shafer's theory in mobility prediction when adequate knowledge about the history of user's travelling patterns is not available.

VII. Conclusions

We proposed a context model for spatial prediction based on spatial and spatiotemporal user context. Actually, we present how ML techniques are applied to the discussed model for predicting future locations in a PCE. Specifically, two approaches of classification schemes (C_A and C_B) are presented that exploit the user context in order to classify and predict future movements. The selected classifier for such model is the J48 Decision Tree classifier, which produces classification rules that represent movement trajectories. We also observe that spatial context achieves better prediction accuracy than spatiotemporal context.

The model is evaluated according to the 10-fold cross validation method and sets of simulated user movements assigned to different degrees of randomness. Evidently, the C_B scheme behaves better in prediction accuracy than the C_A scheme when both spatial and spatiotemporal historical context are considered. We also introduce the coherence metric indicating how the C_A and C_B schemes produce non-contradictory predictions. Hence, the final decision-making relies on the combination of the coherent predictions derived from C_A and C_B. Furthermore, we select the J48 algorithm for the implementation of the C_A and C_B schemes since it demonstrates better prediction accuracy than that of the certain well-known DM classifiers. C_A is also compared with non-DM classifiers where it converges faster than a LA and it assumes better prediction accuracy than MMP for 100% degree of randomness.

However, the model can be enhanced with more semantics and context data such as application context (e.g., service requests), proximity of people (e.g., social context) and destination and velocity of the user movement. Furthermore, the discussed model has to be validated with incremental, not-incremental classifiers, meta-learners and filters (i.e., processes that remove outliers). Hence, the proposed context representation should be able to adapt to such schemes. Finally, the use of approximate representation models (e.g., fuzzy-set theory) has to be considered in order to represent vague and imprecise spatial and temporal contextual information.

REFERENCES

[1] I. H. Witten and E.Frank, *Data Mining: Practical Machine Learning Tool and Techniques*, Morgan Kaufmann Series in Data Management Systems, 2005.

[2] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Series in Data Management Systems, 2001.

[3] E. Alpaydin, *Introduction to Machine Learning*, The MIT Press, 2004.

[4] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Series in Machine Learning, 1993.

[5] M. Mahmood, M. Zonoozi, and P. Dassanayake, "User Mobility Modeling and Characterization of Mobility Patterns", *IEEE in Communications*, vol. 15, no. 7, 1997.

[6] W. Cohen, A. Prieditis and S.J. Russel, "Fast Effective Rule Induction", *Proc. Int. Conf. on Machine Learning*, 1995.

[7] M. Kyriakakos, S.Hadjiefthymiades, N.Fragkiadakis, and L. Merakos, "Enhanced Path Prediction for Network Resource Management in Wireless LANs", *IEEE Wireless Communications Magazine*, Special issue on "The Evolution of Wireless LANs and PANs", vol. 10, no. 6, 2003.

[8] M. Weiss, and C.A. Kulikowski, *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Networks*, Machine Learning and Expert Systems, Morgan Kaufmann, 1991.

[9] 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, Elsevier, 2004.

[10] M. Plutowski, S. Sakata and H. White, "Cross-validation estimates integrated mean squared error", *Advances in Neural Information Processing Systems,* vol. 6, 1994.

[11] S. Hadjiefthymiades, and L. Merakos, "Proxies + Path Prediction: Improving Web Service Provision in Wireless-Mobile Communications", *ACM/Kluwer Mobile Networks and Applications*, Special Issue on Mobile and Wireless Data Management, vol.8, no. 4, 2003.

[12] L. George and G. Maguire, "A class of mobile motion prediction algorithms for wireless mobile computing and communications", *Mobile Networks and Applications 1*, J.C. Baltzer AG, Science Publishers, 1996.

[13] T. Liu, P. Bahl and I. Chlamtac, "Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks", *IEEE JSAC* vol. 16, no. 6, 1998.

[14] S. Choi and K. G. Shin, "Predictive and adaptive bandwidth reservation for hand-offs in QoS-sensitive cellular networks", *Proc. ACM SIGCOMM '98*, 1998.

[15] L. Xiong and H. A. Karimi, "Location Awareness through Trajectory Prediction", *Computers, Environment and Urban Systems*, Elsevier, 2006.

[16] G. Yavas, D. Katsaros, O. Ulusoy, and Y.Manolopoulos, "A data mining approach for Location Prediction in Mobile Environments", *Data & Knowledge Engineering* vol. 54, 2005.

[17] D. Katsaros, A. Nanopoulos, "Clustering Mobile Trajectories for Resource Allocation in Mobile Environments", *Intelligent Data analysis*, 2003.

[18] K. Laasonen, "Clustering and Prediction of Mobile User Routes from Cellular Data", *Proc. PKDD,* 2005.

[19] V. S. Tseng, and K. W. Lin, "Efficient mining and prediction of user behavior patterns in mobile web systems", *Information and Software Technology* vol. 48, Elsevier, 2006.

[20] A. Bhattacharya and S.K. Das, "LeZi update: An Information Theoretic Approach to Track Mobile Users in PCS Networks", *Proc. ACM/IEEE Mobicom '99*, 1999.

[21] A. Karmouch, N. Samaan, "A Mobility Prediction Architecture Based on Contextual Knowledge and Spatial Conceptual Maps", *IEEE Transactions on Mobile Computing*, vol. 4 no. 6, 2005.

[22] C. Anagnostopoulos, P. Bougiouris, and S. Hadjiefthymiades, "Prediction intelligence in context aware applications", *Proc. 6th International Conference on Mobile Data Management*, 2005.