University of Kasdi Merbah Ouargla

Faculty of New Information and Communication Technologies

Department of Computer Science and Information Technology

Dissertation Submitted to the Department of Computer Science and Information Technology in Candidacy for the Degree of Master LMD in Industry

Presented by:

BenKrima Imane & Razoug OumElhana

The consideration of time factor in mobility prediction and location recommendation

Publicly defended:

On:

Before the jury:

Dr Amirat Hanane	Supervisor	UKM Ouargla
Dr Ben Kherourou Chafika	Examiner	UKM Ouargla
Dr Benmir Abdelkader	Examiner	UKM Ouargla

Academic Year: 2019 – 2020

Acknowledgments

We would like to express our gratitude to all those who have made this thesis possible. We wish to thank first and foremost our ALLAH, the Almighty, the greatest of all. We are deeply indebted to our families for their support and assistance through the provision of the comfort conditions

(Mum and Dad Thank you for everything!).

Our special thanks go also to our sisters and brothers for their encouragement and support.

Our sincere thanks also go to dear Dr Amirat Hanane for her valuable comments and for her superb guidance.

Abstract

Route prediction is the missing piece in several proposed ideas for intelligent vehicles and smart cities. This field has taken great importance recently. Because of the different problems that occur on roads, the specificity of transportation and the nature of the mobility data, various models have been proposed for effective route prediction. For instance, Markov model, sequential patterns-based models, etc. As roads problems evolve, the application of these classical methods is no longer sufficient.

In this thesis, we propose two novel models for route prediction, namely, PreNext and PreGraph. The first model PreNext depends on CPT (compact prediction tree) model, thus it offers all its advantages including its lossless property that allow conserving all the data in to perform prediction, its lower storage space requirement, predicting rare cases with high accuracy, etc. Our second model PreGraph is a dependency graph-based model. PreGraph represents roads as a graph which is then used to predict the next traversing road. Unlike many prediction models, the designed models are compact and easy to be constructed, and can thus provide efficient solutions for prediction. Our proposals were compared with well-known prediction models and exhibiting quite promising results on two real- world datasets.

Keywords: Route prediction, dependency graph, compact prediction tree, lossless model, noise tolerance, time factor.

Résumé

La prédiction de route est la pièce manquante dans plusieurs idées proposées pour les véhicules intelligents et les villes intelligentes. Ce domaine a pris une grande importance récemment. En raison des différents problèmes rencontrés sur les routes, la spécificité du transport et de la nature des données de mobilité, divers modèles ont été proposés pour une prédiction efficace des routes. Par exemple, le modèle de Markov, les modèles séquentiels basés sur des règles… etc. À mesure que les problèmes de routes évoluent, l'application de ces méthodes classiques n'est plus suffisante.

Dans cette thèse, nous proposons deux nouveaux modèles de prédiction de route, à savoir, PreNext et PreGraph. Le premier modèle PreNext dépend du modèle CPT, il offre donc tous ses avantages, y compris sa propriété lossless ou sens perte qui permet de conserver toutes les données pour effectuer la prédiction, son faible besoin d'espace de stockage, la prédiction de cas rares avec une grande précision, etc. Notre deuxième modèle PreGraph est modèle basé sur un graphe de dépendance. PreGraph représente les routes sous forme d'un graphe qui est ensuite utilisé pour prédire la prochaine route à traverser. Contrairement à de nombreux modèles de prédiction, les modèles conçus sont compacts et faciles à construire, et peuvent ainsi fournir des solutions efficaces pour la prédiction. Nos propositions ont été comparées à des modèles de prédiction bien connus et ont présenté des résultats assez prometteurs sur deux jeux de données réels.

Mots clés : Prédiction de route, graphe de dépendance, arbre de prédiction compact, modèle sans perte, tolérance au bruit, facteur de temps.

الملخص

التنبؤ بالطريق هو القطعة المفقودة في العديد من الأفكار المقترحة للمركبات الذكية والمدن الذكية. اكتسب هذا المجال أهمية كبيرة في الآونة الأخيرة. بسبب المشاكل المختلفة التي تحدث على الطرقات وخصوصية النقل وطبيعة بيانات التنقل، تم اقتراح نماذج مختلفة للتنبؤ الفعال بالمسار. على سبيل المثال نجد نموذج ماركوف، النماذج المتسلسلة القائمة على القواعد ... إلخ. ولكن مع تزايد مشاكل الطرق، لم يعد تطبيق هذه الأساليب الكلاسيكية كافياً.

في هذه الرسالة، نقترح نموذجين جديدين للتنبؤ بالمسار وهما (PreNext) و (PreGraph). يعتمد النموذج الأول (PreNext) على نموذج CPT، وبالتالي فهو يقدم جميع مزاياه بما في ذلك خاصية عدم فقدان البيانات التي تسمح بالحفاظ على جميع البيانات لأداء التنبؤ، ومتطلبات مساحة التخزين المنخفضة، والتنبؤ بالحالات النادرة بدقة عالية، وما إلى ذلك. نموذجنا الثاني (PreGraph)هو نموذج قائم على الرسم البياني التبعية. يمثل الرسم البياني (PreGraph) الطرق كرسم بياني يُستخدم بعد ذلك للتنبؤ بالطريق التالي لسائق. ايضا على عكس العديد من نماذج التنبؤ، فإن النماذج المصممة مضغوطة وسهلة البناء، وبالتالي يمكن أن توفر حلولًا فعالة للتنبؤ. تمت مقارنة مقترحاتنا مع نماذج التنبؤ المعروفة جيدًا وأظهرت نتائج واعدة جدًا على مجموعتي بيانات من العالم الحقيقي.

الكلمات الأساسية: تنبؤ المسار، الرسم البياني للتبعية، شجرة التنبؤ المدمجة، نموذج بدون خسارة، تحمل الضوضاء، عامل الوقت.

Table of Content

Experimental evaluation ……………………………….Chapter IV

List of figures

List of tables

1. General Introduction

Understanding driver intent during his trip is an essential step for determining and predicting his next pathways and reduce the risks that may affect the driver and his car. Due to the increase in traffic accidents and the technical problems of cars, road prediction systems have evolved. This development is evident through Intelligent Transportation Systems (ITS) and Location-based Services (LBS). These systems and services mainly depend on many types of prediction systems that are being used like Dynamic routing which takes in real-time mobility data and analyses the routes and Eco-routing which mainly focuses on conserving the fuel by providing the best feasible route. The basis for all these systems is the data which are collected for days and months and then used to train the machines to understand the user's behaviors of travelling in a specific path.

In recent years, route prediction has become the focus of the researchers and challenge between them to cover all vehicle routing problem. This branch of science depends on the hypothesis that forecasting the next route of a driver or car is done by considering the spatial regularities in its movements. Each person has driving habits or preferences (e.g. The driver may prefer a long route with less congestion over a short route with more congestion) that influence his/her choice of a path to reach a destination. In addition, many people choose the same set of roads to go from the same source location to a specific destination (e.g. from work to shopping mall) or to destinations in the same area (e.g. from work to local restaurant). Thus, predicting the next location or road segment(s) that a driver can take is obtained by matching his current path with his previous paths. Therefore, the route prediction problem is considered as a sequence prediction that aims at using historical sequence information from set of users to predict the next value or values in the sequence.

2. Motivations

Nowadays, route prediction problem has been extensively studied. Each existing work differed from a scholar to other according to the techniques used, the type and size ,the final goal of prediction (routing, fuel consuming optimization ,etc.), the range of prediction (short and long-term).Considering the technique used, these works can be categorized into systems that are suing artificial neural networks, probabilistic models, trip matching, clustering, etc. For instance, *Epperlein* et al. [18] suggested using Markov chain model and *Tiwari* et al. [22] have built a route prediction using partial match PPM (Prediction by Partial Matching) algorithm. Besides, *Qiang* et al. [24] have proposed a data mining-based prediction system whereas *Terroso-Saenz* et al. [27] *Park* et al. [33] and *Tiwari* et al. [34] have adopted trip matching approaches.

Although the above models have achieved satisfactory results in some applications, an important drawback is their noise sensitivity toward the mobility patterns that they can learn. The smallest deviation in mobility data affects the prediction process result, and thus the prediction accuracy. This problem is exacerbated for noisy mobility datasets such as GPS trajectory data which are prone to several disturbances and inaccuracies. In addition, most of the proposed model are unable the continuous geographical distances between locations and time interval in modelling sequential data. For instance, for a given person who is accustomed to go the gym on the evening and rarely go to the cinema, the probability that he /she will go to the gym at this time is higher than going to the cinema relying on his/her interests and demands in this specific time .Moreover, these local temporal contexts have fundamental effects in revealing the characteristics of the driver and are helpful for behavior modeling. The behavior of person who go repeatedly to a gym or a stadium indicates that both the gym and the stadium have higher importance than other contexts. Furthermore, as some behaviors are periodical such as going to the mosque every Friday, or going to the market every weekend the effect of time interval becomes important for temporal prediction in such situations.

3. Contributions

In this thesis, we tackle the above challenges by proposing two novel models PreNext and PreGraph. While PreNext is based by an accurate and lossless prediction model called Compact Prediction Tree (CPT). PreGraph is built upon the Dependency Graph (DG) predictor to build a mobility prediction framework.

The underlying idea of the two proposals is that vehicle mobility is order dependent. A vehicle passes through a sequence of road segments to reach some desired destinations, by following a specific order and traversing a set of road segments in specific directions. Besides, the repeated behaviors and habits of a driver helps to predict her upcoming direction over time. Hence, to predict the next route that the driver may be visited in the future, PreGraph and PreNext utilize the current road of a driver and his historical data.

PreGraph builds a prediction graph, where nodes are roads and arcs are used to represent the visiting order of roads by drivers. Then, the prediction graph is used to suggest the next location of a driver by attempting to match its current mobility pattern with graph paths.

Furthermore, PreNext provides the CPT features such as the benefits of preserving all the information contained in training sequences to perform predictions. Yet, and regardless of CPT's lossless nature that allows it to preserve all relevant data, a significant data reduction is attained through its prediction structures and its *FSC (Frequent Subsequence Compression)* and *SBC (Simple Branches Compression)* advanced compression strategies [6].

In addition to its fast prediction process unlike several other prediction models that are noise sensitivity, CPT is noise tolerant. It copes with noise found in data by adopting a similarity measures mechanism that tolerates variations in mobility data while performing predictions. Moreover, PreGraph and PreNext predictors have been further extended to regard the temporal influence on prediction process. Rather than considering only the location in each sequence, the two proposals takes the spatial and temporal contexts into consideration, Thus, the temporal context associated to each sequence is considered to form time-extended sequences so-called: temporal sequences. Each element of sequence refers to each visited road and the visited time associated with this road. As a result, we are dealing in this thesis with temporal graph and temporal CPT.

The main contributions of this thesis are summarized as follow:

- We introduce two models for route prediction based on the compact prediction tree (CPT) and dependency graph (DG) models.
- Our proposals have been further extended to considerate the temporal context associated with location data.

- We evaluate our proposals and compare them with two state-of-the-art models using two large-scale and realistic mobility datasets.

4. Thesis's organization

This thesis is organized as follows:

- In the first chapter, we investigate the subject of probabilistic models on which most of the existing studies are depending on. To ensure better understanding of these model, a practical example is associated to the description of each presented model. At the end of the chapter, fast overview that highlights a set of probabilistic based approaches from different fields (networks, medical, social, etc.), is given.
- In the second chapter, we first give a brief review of some related work in the field of route prediction, and then we present a taxonomy that classifies these works according to the technique employed. In this same chapter, we summarize the key existing works and emphasize their positive and negative aspects.
- In the third chapter, we describe our basic route prediction models and present the extension of the models to handle the temporal context of mobility.
- In the fourth chapter, we evaluate the performance of our proposal with other models and present a comparison study of the two models by varying their parameters.
- Finally, we conclude the thesis with a conclusion.

Chapter I

Probabilistic Models

1. Introduction

Probabilistic models have taken great value because they are intimately related to the prediction studies. These models are applied in all situations where there is uncertainty about the outcome and we would like to precise a description of what could happen is needed. More specifically, probabilistic models have been widely used in several fields such as medical treatment outcomes prediction, weather forecasting, economic studies, elections...etc.

Relying on the theory of the fact that randomness plays a role in predicting future events, a probabilistic prediction mainly aims at assigning a probability degree of value to each possible outcome on an experiment.

This chapter begins with a general overview on some probabilistic models namely (Markov-Chain model, Hidden-Markov-model (HMM), Prediction by Partial- Match (PPM), Lempel-Ziv, Compact Prediction -Tree (CPT) and Dependency Graph (DG)). Next, a brief description with an illustrative example of each of these models is given. Finally, a sample of probabilistic based approaches in different fields are classified relying on the type of probabilistic models used is presented.

2. Probabilistic models

2.1. Markov chain model

Markov model is a stochastic model used to model randomly changing systems where it is assumed that future states depend only on the current state not on the events that occurred before it [1].We describe a Markov chain as follows: We have a set of states $S = \{\ S_1, S_2, \ldots, S_n\}$. The process starts in one of these states and moves successively from one state to another. Each move is called a step. If the chain is currently in state S_i, then it moves to state S_j at the next step with a probability denoted by P_{ij} and this probability does not depend upon which states the chain was in before the current state. The probabilities P_{ij} are called transition probabilities. The process can remain in the state it is in and this occurs with probability P_{ii} (loop). An initial probability distribution defined on *S* specifies the starting state. Usually, this is done by specifying a particular state as the starting state [2]. For the sake of illustration, let consider the example depicted in Fig.1.The latter represents three stores in small-town (*A*, *B* and *C*). On any given week, there are 180 people go to store *B* and

120 go to store C and 200 of them go to store A to do their shopping for the week. However, people don't typically go to the same store all the time because of their requirement. In other words, some customers go back to the same store also others are changing their destination. let's consider $X_0 \begin{bmatrix} A_0 \\ B_0 \\ C_0 \end{bmatrix}$ is the current state or matrix which contains the current probability of visiting the three stores and $X_1 \begin{bmatrix} A_1 \\ B_1 \\ C_1 \end{bmatrix}$ is the next state which we aim to achieve,$X_1 = PX_0$where P is the transmission matrix which contains the probability of moving from the store *i* to store *j*. In each row are the probabilities of moving from the state are represented by that row as follow: $P = \begin{bmatrix} A\ to\ A & B\ toA & Cto\ A \\ A\ to\ B & B\ to\ B & C\ to\ B \\ A\ to\ C & B\ to\ C & C\ to\ C \end{bmatrix}$

Thus $X_1$is calculated by multiplying P with X_0 (matrix multiplication). $\begin{bmatrix} 0.404 \\ 0.316 \\ 0.280 \end{bmatrix} = \begin{bmatrix} 0.80 & 0.2 & 0.1 \\ 0.1 & 0.7 & 0.3 \\ 0.1 & 0.1 & 0.6 \end{bmatrix} \begin{bmatrix} 0.40 \\ 0.24 \\ 0.36 \end{bmatrix}$ and $X_{1_{[1,1]}}$= 0.404 represents the probability of visiting the store A that we aim to find it and the number of customers that will visit the store A next week calculated as 0.404 x*500 =202.

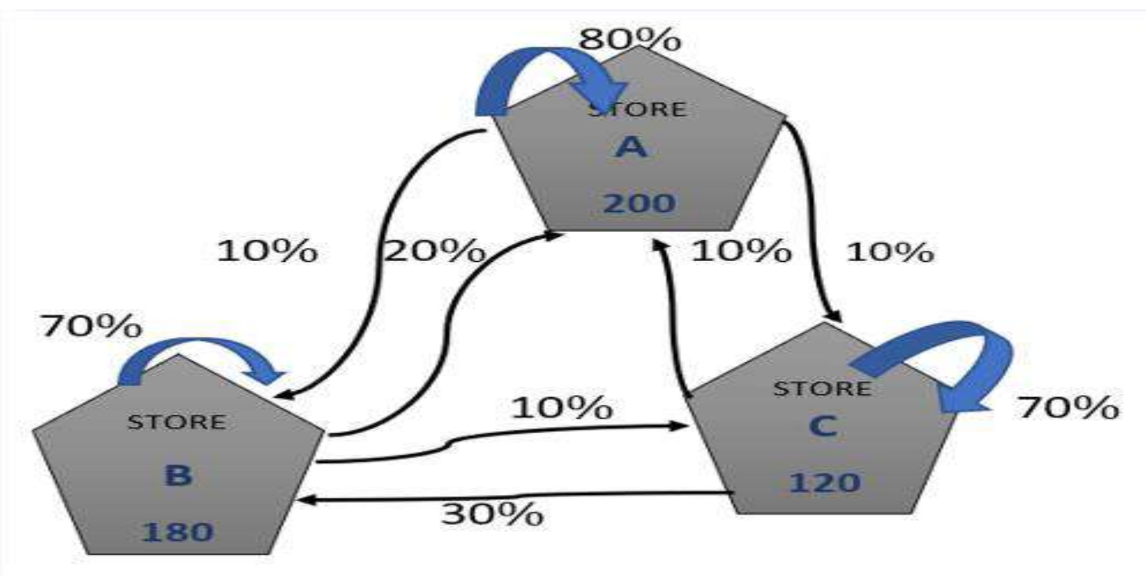


Fig.1 Markov chain model example.

2.2. Hidden Markov model

Hidden- Markov model (HMM) is a statistical model that can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable. The observed event is called a symbol whereas the invisible factor underlying the observation is called a state. The HMM consists of two stochastic processes, namely, an invisible process of hidden states and a visible process of observable symbols. The hidden states form a Markov chain, and the probability distribution of the observed symbol depends on the underlying state. For this reason, an HMM is also called a doubly-embedded stochastic process [3] and is named hidden as the events are not observed directly. For instance, in the speech recognition process, part-of-speech tags in a text are not observed. Yet, words are seen and the text must infrared from the word sequence. In this case, the tags are hidden because they are not observed. The HMM includes five components 1) a set of *n* state $Q = q_1 q_2 \cdots q_n$,2) a transition probability matrix *A* each a_{ij}representing the probability of moving from state *i* to state *j* $\sum_{j=1}^{n} a_{ij} = 1 \forall i$, $A = a_{11} \ldots \ldots a_{ij} \ldots a_{nn}$, 3) a sequence of *T* observation $O = O_1 O_2 \cdots O_T$, 4) a sequence of observation likelihoods also called emission probability *B = bi (Ot)*, 5) $\pi = \pi_1, \pi_2, \pi_3, \ldots \ldots, \pi_n$ an initial probability distribution over states π is the probability that Markov chain starts with. For instance, let us consider two states of weather *A* and *B* while A denotes rain and B represents dry weather. For the given sequence, S_k *<B, B,A,A>* of weather the probability *P ({B,B,A,A})*of S_k with the transition probabilities of weather states are given as *P(A|A) =0.3, P(B|A) =0.7,P(A|B) =0.2, P(B|B) =0.8*.Besides, the probability of rainy day is 0.4 and dry day is 0.6. The probability of S_k is calculated by applying the following formula:

$P(S_1, S_{I2} \ldots \ldots S_{IK}) = P(S_{IK} | S_{I1}, S_{I2} \ldots \ldots S_{IK-1})\, P(S_{I1}, S_{I2} \ldots \ldots S_{IK-1}) = P(S_{IK} | S_{IK-1})\, P(S_I, S_{I2} \ldots \ldots S_{IK-1}) = P(S_{IK} | S_{IK-1})\, P(S_{IK-1} | S_{IK-2}) \ldots \ldots P(S_{I2} | S_{I1})\, P(S_{I1})$. Thus, *P {(B, B, A, A)} = P(A|A) P(A|B) P(B|B) P(B) = 0.3*0.2*0.8*0.6=0.0288.*

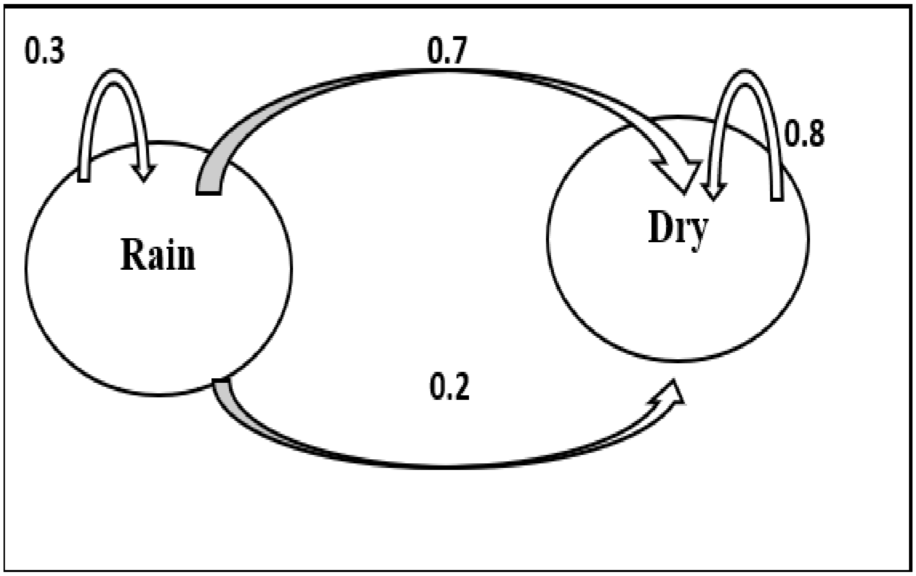


Fig.2 The weather states with probabilities.

2.3. Prediction by partial Matching (PPM)

Prediction by Partial Matching (PPM) is a sequence prediction model proposed by (Cleary &Witten (1984)) [4]. It is mainly used for data compression and sequence predictions. PPM is a variable-order Markov model that uses various lengths of previous contexts to build the prediction model. The basic algorithm initially attempts to use the largest context. The size of the largest context is predetermined if the symbol to be encoded has not previously been encountered in this context. An escape symbol is encoded and the algorithm attempts to use the next small context and if the symbol has not occurred in this context either. The size of the context is further reduced. This process continues until either a context that has previously been encountered with this symbol or no much of the context has been encountered. For instance, the Fig.3 represent the PPM prediction for the sequence *S=ABCDCECF-AECFCDCBA-AFCECDCBA-ABCECDC*.The context step(1) is *ECDC*. The symbols which had been appeared after this context along the sequence are just *B* so *B* is to be excluded and the escape symbols have encoded with probability1/2.The context reduced to *CDC* in step(2)and the process continues with the symbol after the new context *CDC* that is *E*,*E* excluded and the escape symbols encoded with probability2/3.The context in step (3) became *DC* . Here, no new symbols after *DC* would be found. The process continues in the last step (4). The new context is *D* and the symbols after *D* is just *F* and *F* encoded with probability 2/4+2+4=1/5. The process will be finished in this step as *F* has not been encountered in any context before.

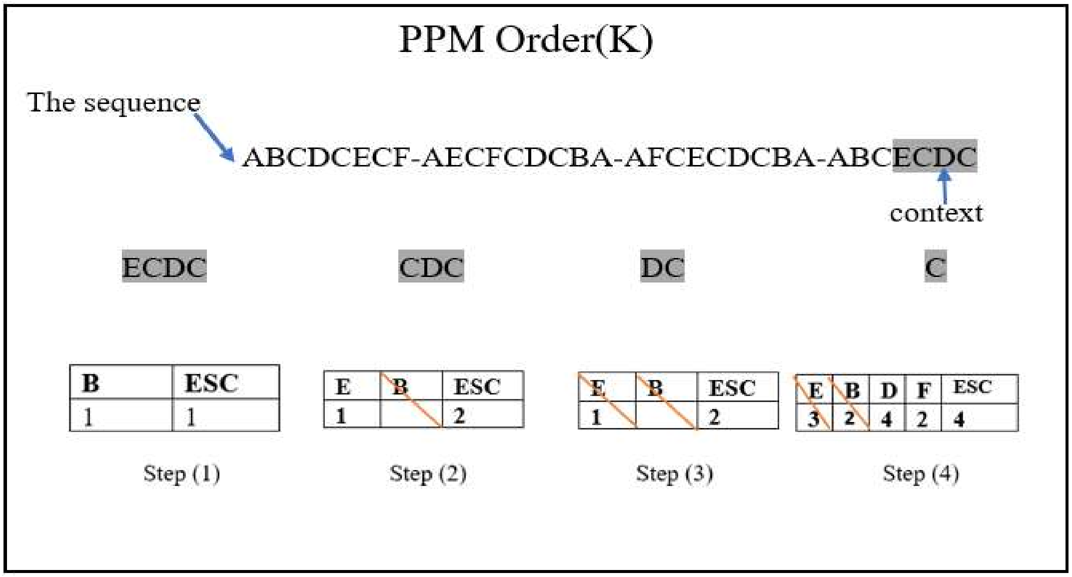


Fig.3 PPM example.

2.4. Lempel Ziv (LZ)

Around 1977, Lempel and Ziv [5] have developed the Lempel-Ziv popular class of adaptive dictionary data compression techniques. The prediction components of this algorithm were first discussed by Langdon and Rissanen (1983) [5] and then it was simplified and became known as LZ78.LZ78 is considered as a dictionary-based text compression algorithm that performs incremental parsing of an input sequence. This algorithm parses the given input string into substrings, and as it is a compression algorithm it also consists of [encoder\decoder] processes. In our case, and as we do not need to reconstruct the parsed sequence no need to consider encoder/decoder system. Yet, LZ is simply used as a system that split up a given sequence of states into phrases. From this standpoint let's consider the following sequence:

S = < *AABABBBABAABABBBABBABB*> as an input. LZ breaks up this sequence into the following set phrases: *A*|*AB*|*ABB*|*B*|*ABA*|*ABAB*|*BB*|*ABBA*|*BB*. The algorithm maintains statistic for all contexts seen within the phrases that are stored in the LZ parsing tree. For example, the context *A* appears six times at the beginning of the phrases *(A|AB|ABB|ABA|ABAB| ABBA)*, *BB* appears two times *(|BB|BB)* and so on. These context statistics are stored in a tree depicted in Fig.4.

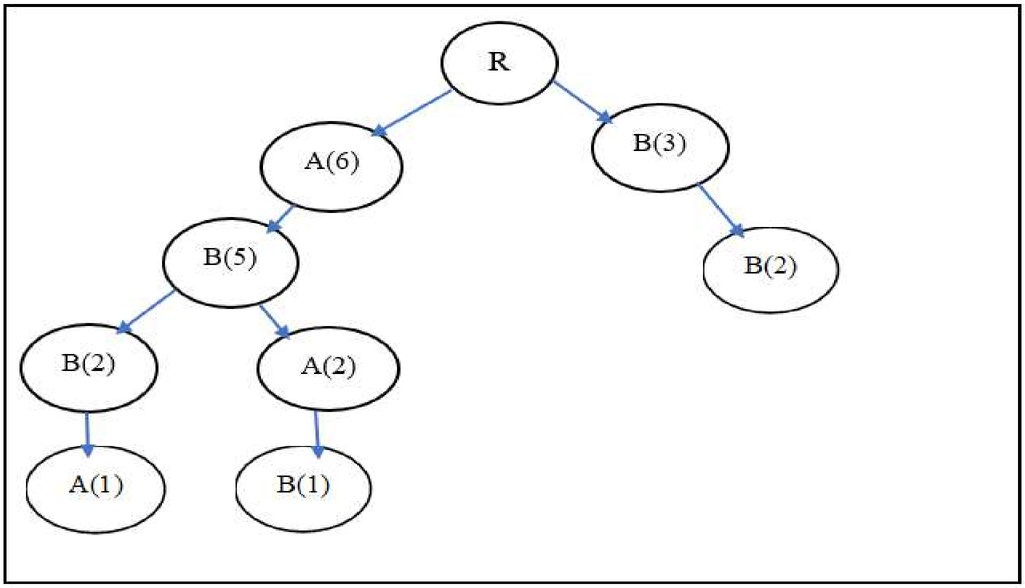
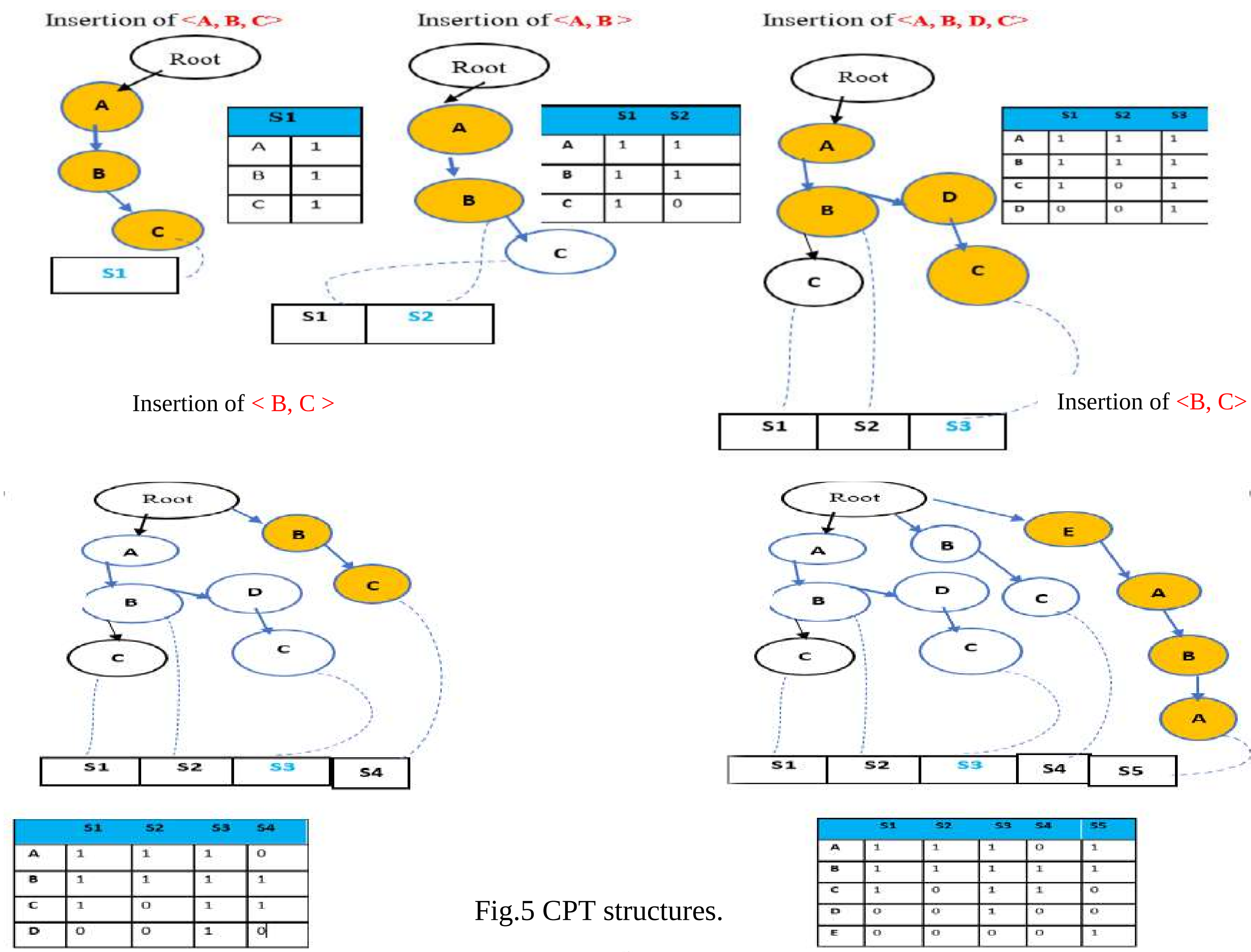


Fig.4 A sample of a tree constructed by the LZ78 parsing.

2.5. Compact prediction tree (CPT)

Compact prediction tree (CPT) is a prediction algorithm that proposes a new approach called *"lossless"* that means predicting without any loss of information. Similarly, to other prediction models, CPT is performing the training and prediction or testing phase. In the training phase, CPT construct three data structures namely:1) *Prediction Tree (PT)*, it's a type of prefix tree. It includes all training sequences. Each node in *PT* represents an item and each training sequence is represented by a path starting from the tree root and ending by an inner node or a leaf.2) *Lookup Table (LT)*, is an associative array which allows locating any training sequence in the prediction tree with a constant access time and 3) *Inverted Index (II)*, is a set of bit-vectors that indicate for each element the set of sequences that it contains [6].Fig.5 illustrates the construction of these structures by the successive insertion of the five sequences: $S_1 =< A, B, C >$, $S_2 =< A, B>$ $S_3 =< A, B, D, C >$,$S_4 =< B, C>$,$S_5 =< E, A, B, A >$.

Once the three structures are built, the prediction could take place. For instance, the prediction of the next symbols after the sequence *S* = <*A*, *B*> is done by finding all sequences that contain the items of S in any order and any position. These sequences are so-called *the sequences similar to S* or the matching sequences. To find these similar sequences the *Inverted index* structure in Fig.6 is used. *II* performs AND operation to find the sequences S_1, S_2 and S_3 that contain the items *A* and *B*. The *lookup table* is then used to allow traversing the matching sequences from the end to the start. From these sequences $S_1 =< C, B, A >$, $S_2 =< B, A>$, $S_3 =< C, D, B, A >$, the subsequences <*C*> and <*CD*> called *consequents* are retrieved. Each item of those consequents is then stored in a data structure named *Count Table* (CT). This structure comprises a list of possible candidate items and their respective score (i.e. number of times an item appears after the target sequence *S*). The item with the highest score within the *CT* is the predicted item. In our example, item *C* is retained because its score is 2 while *D* has a score of 1.



	S1
A	1
B	1
C	1

	S1	S2
A	1	1
B	1	1
C	1	0

	S1	S2	S3
A	1	1	1
B	1	1	1
C	1	0	1
D	0	0	1

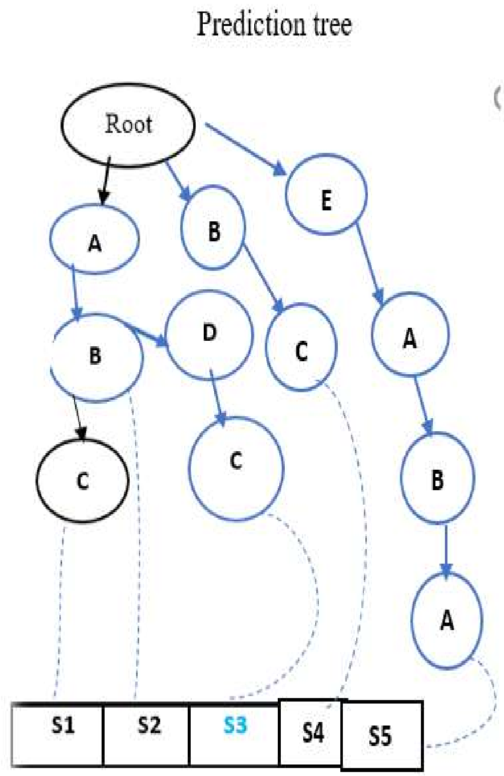
	S1	S2	S3	S4
A	1	1	1	0
B	1	1	1	1
C	1	0	1	1
D	0	0	1	0

	S1	S2	S3	S4	S5
A	1	1	1	0	1
B	1	1	1	1	1
C	1	0	1	1	0
D	0	0	1	0	0
E	0	0	0	0	1

Fig.5 CPT structures.

Inverted index

	S1	S2	S3	S4	S5
A	1	1	1	0	1
B	1	1	1	1	1
C	1	0	1	1	0
D	0	0	1	0	0
E	0	0	0	0	1



Count Table

Item	Score
C	2
D	1

Fig.6 CPT prediction.

2.6. Dependency Graph (DG)

2.6.1. Directed graph and Undirected graph

A directed graph is a pair $G = (V, A)$ where V is a set of v nodes, and A $\subseteq V \times V$ denotes a set of directed arcs. Each arc is an ordered pair $e = (u, v)$ that represents an asymmetric relationship between two nodes. Note that a direct graph can have self-loop (u, u) to the same node. Besides, an undirected graph is a pair $G = (V, E)$ where V is a set of nodes, and $E \subseteq V$ is a set of arcs. In contrast to directed graph, each arc in an undirected graph is an unordered pair $e= \{u, v\}$ (or equivalently {v, u}). By this definition, an undirected graph cannot have self-loops since {v, v} = {v}. Therefore, undirected graphs represent symmetric relationships.

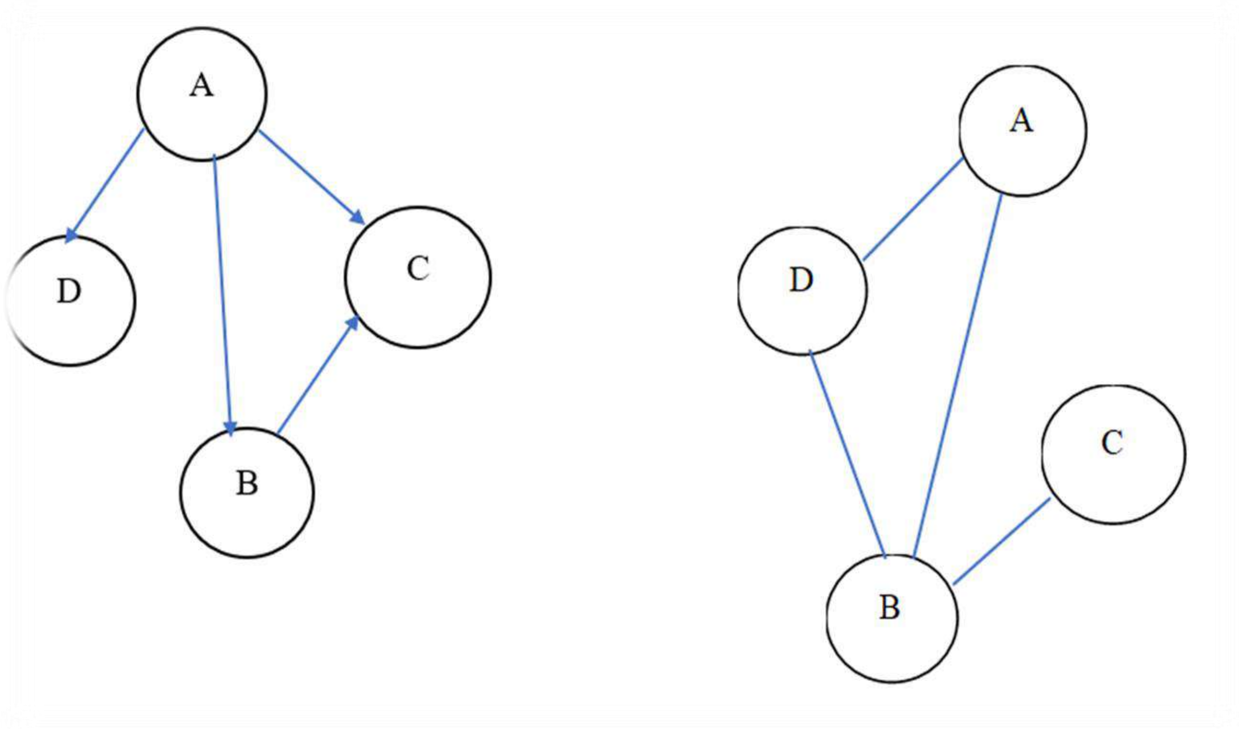


Fig.7 An example of a directed and undirected graph of 4 nodes.

2.6.2. Dependency Graph

A dependency graph is a directed graph representing dependencies of several objects towards each other [6]. Events in any system can be represented as random variables whereas the dependencies between them are modelled as probabilistic relations amongst the variables. The probability of a random variable is conditioned and learnt from the observed values of the random variables in the system.

In the dependency graph, random variables are represented as graph nodes with directed arcs being the conditional dependence of one variable on another. The model represents the inter-dependency of different random variables so it is a valuable framework to understand complex systems built from these variables. The other advantage of the dependency graph is that when the values of certain variables are observed, the conditional dependencies can be used to infer the values of the dependent variables. Hence, the framework of this graphical model can be used for diagnostics and a tool for prediction of unobserved variables [7]. One of the most important parameters that the dependency graph based on is the Lookup-Window. This latter determines the size of the window to retrieve the next symbols of a sequence.

For the sake of illustration, let us consider the DG depicted in Fig.8 constructed from two sequence

S_1 = <*A*, *B*, *C*, *A*, *C*, *B*, *D*> and S_2 = <*C*, *C*, *A*, *B*, *C*, *A*>. We aim to build a model to do some prediction, to find the probability of *A* followed by *B* and *A* followed by *C* with Lookahead-Window size 2 that mean look two symbols by two when constructing the graph.

As illustrated in Fig.8 four nodes are created as the number of symbols and every two successive nodes (retrieve by the size of lookup-window) are connected by an arc with a value which represents how many time *A* is followed by *B* in all sequences. Thus, $\mathrm{P(B|A) = 3|sup(A) = 3|4}$ and $\mathrm{P(C|A) = 3|sup(A) = 3|4}$.where sup(A) is how many time *A* appears in all sequences.

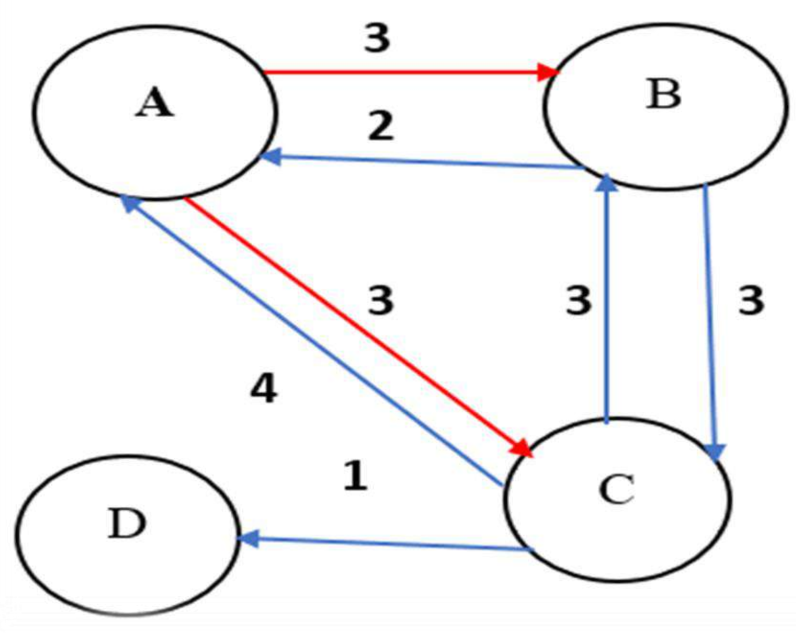


Fig.8 A sample of DG model with lookahead =2.

3. Taxonomy of some prediction works

To review some probabilistic based prediction approaches, we provide in this section a taxonomical classification that categorizes the prediction models according to the probabilistic model employed to perform prediction.

Work	Probabilistic model used	Task
Katsaros et al. (2009) [9]	Markov Chain model	Prediction of what in Wireless Networks by Markov Chains.
Zakaria et al. (2019) [10]	Markov Chain model	Forecasting Air Pollution Index of Miri, Sarawak
Qiao et al. (2017) [11]	Hidden-Markov model (HMM)	Predicting Social Unrest Events with Hidden Markov Models Using GDELT
Liu et al. (2017) [12]	Hidden-Markov model (HMM)	Big Data-Driven Hidden Markov Model-Based Individual Mobility Prediction at Points of Interest
Sael et al.(2010) [13]	Prediction by Partial Matching (PPM)	Binding Ligand Prediction for Proteins Using Partial Matching of Local Surface Patches
Tan et al. (2015) [14]	Lempel-Ziv (LZ)	FPGA-based hardware accelerator for the prediction of protein secondary class via fuzzy K-nearest neighbors with Lempel–Ziv complexity-based distance measure
Cui et al. (2016) [15]	Lempel-Ziv (LZ)	Double-dictionary matching pursuit for fault extent evaluation of rolling bearing based on the Lempel-Ziv complexity
Mallick et al. (2016) [16]	Compact Prediction Tree (CPT)	Weather prediction using CPT+ algorithm
Zimmermann et al.(2006)[8]	Dependency Graph (DG)	Predicting Subsystem Failures using Dependency Graph Complexities
Edakunni et al. (2015) [7]	Dependency Graph (DG)	probabilistic dependency Networks for prediction and diagnostic

Table1: Taxonomy of some prediction models.

4. Conclusion

Through this chapter, we have reviewed a small set of probabilistic models. For each of these models, brief descriptions with their key (properties and main concepts are given). We have also supported our explanation with illustrative examples for each model. To highlight the most important works that have been proposed relying on probabilistic models, a small set of approaches are presented at the end of this chapter.

Chapter II
Route Prediction

1. Introduction

Due to the rapid development of technology, movement prediction studies became the interest of researchers. The main challenge between them is to find solutions for all mobility problems such as social networking applications, reminder, aged people movement optimization, recommender systems, mobility of object (vehicles, human being, etc.).Predicting the future locations of vehicles is regarded as the most interesting topic where researchers seek to obtain more accurate and effective predictors to mainly improve the quality of ITS (Intelligent Transportation Systems). Hence, decrease congestion in traffic.

A route prediction system provides routes based on driving history and the driver requirements and desires to take a specific route. The driver may prefer a long route with less congestion over a short route with more congestion. Route prediction has many challenges constraints on which the routes are predicted, so many algorithms and methods can be used to give priority to a specific constraint. In literature, several models have been proposed. In this chapter, we aim to review related work on route and destination prediction problem.

2. Route prediction

Currently, many drivers use different kinds of route prediction systems to acquire better driving routes. Route prediction systems have been widely used and it offers extensive services for both the driver and his car such as intelligent transportation systems. It can be used to display traffic warnings, location-based services(e.g. systems that determine vehicles paths, recommendation route systems that play an important role in many applications like the display of targeted advertisements about points of interest and shops that a user is approaching).In addition, it was also contributed to reducing fuel consumption, therefore, it provides a major economic and environmental benefits. Route prediction consists of predicting the next road of a driver according to its current path and his previous so, it can be viewed as an instance of the problem of sequence prediction.

Definition1 (**Road segment)**

A road segment is a part of a vehicle or a driver location history. It is a directed edge between two junctions. As depicts in Fig.10 road segment *Rx* and *Ry* connecting two junctions [28].

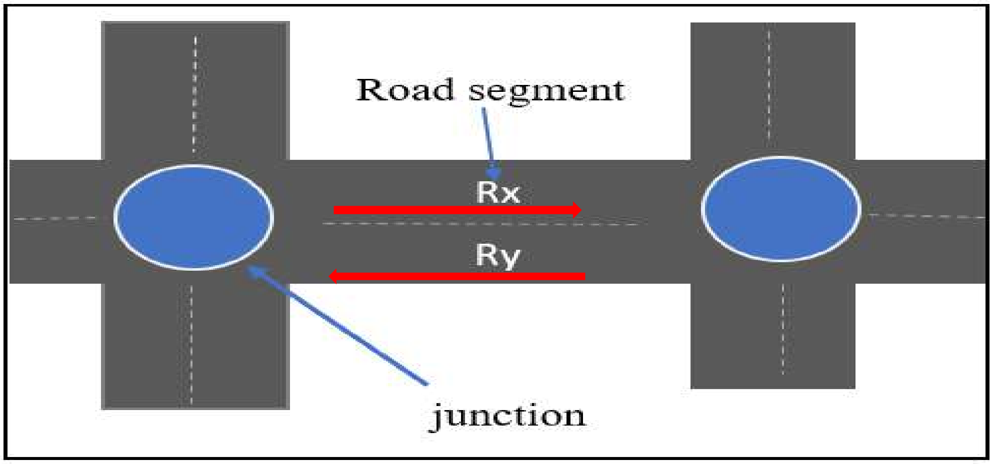


Fig.10 Road segments Rx and Ry connecting two junctions.

Definition 2 (Trajectory)

A path made by vehicle or driver through the road where it moves (GPS dots) $Trj=d_1 \rightarrow d_2 \rightarrow \dots \rightarrow d_n$, where d_i *(Longitude, Latitude, Timestamp)* $\in d$, and d is GPS points set.

Definition 3 (Stay point)

A stay point is a geographic area constituting where the person stayed for a period greater than a time threshold T_{thr} and distance between GPS trajectory points doesn't exceed a spatial threshold D_{thr}.

Definition 4 (Trip)

A trip is a GPS trajectory must be between stay points in the location history, so stay points are trip delimiters. Note that, the driver's starting point is also considered as stay point.

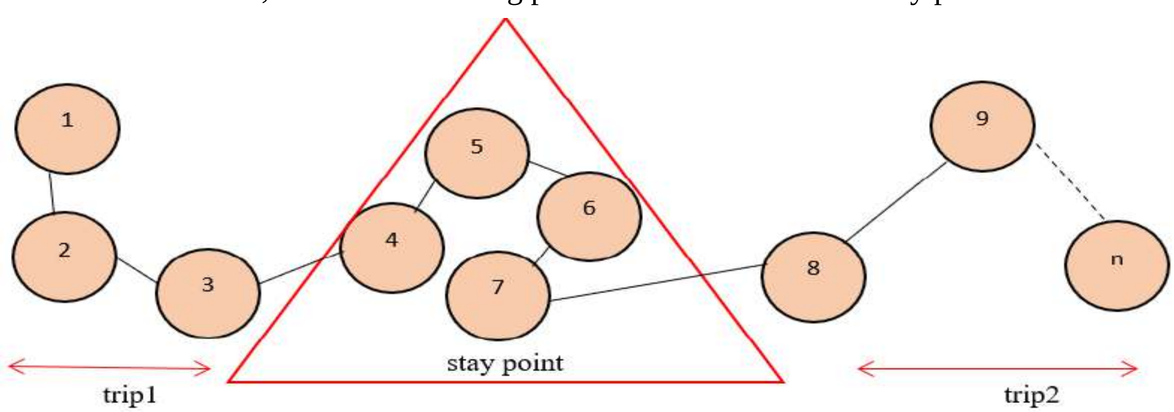


Fig.11 Example of trajectory, stay point and trip.

Definition 5 (Road network)

A road network is a directed graph *(N, RS)*, where *N* is a set of nodes representing the road junctions in a given area, and *RS* is a set of arcs, representing all the road segments joining those junctions.

Definition 6 (Location sequence)

A location sequence $S=L_i,L_{i+1},..,L_m$ is an ordered list of locations visited by a driver V_i during a time period *TP*.

Definition 7 (Road history)

The road history *(RH)* of a vehicle *Vi* is the set of all location sequences of *Vi*.

3. Related work

Several studies have addressed the problem of route predictions [18],[24],[26]. These works could be categorized according to 1) the range of prediction whether it is a short prediction (predicting the next location) [21], or a long one (future locations and destination) [23], or 2) the technique adopted. In this chapter, we will classify related work according to this last criterion. Considering the technique used, existing works can be classified into three main categories: 1) Probabilistic models approach 2) Artificial Neural networks approach 3) Clustering-based approach [17].

3.1 Probabilistic models

Probabilistic models have been widely used to mine frequent patterns for route prediction problems. *Epperlein* et al. [18] proposed Markov chains as models for the journey patterns, they explained how trips can be modelled as outputs of stochastic processes to obtain an estimate of the posterior probabilities of each known journey pattern. *Simmons* et al. [19] presented a Hidden Markov model (HMM) to predict a driver's intended route and destination through observations of the driver's habits. *Gambs* et al [20] improved a previously existed mobility model named *v-Mobility Markov Chain (v-MMC)*, to incorporate the v previous visited locations. They showed that prediction accuracy increases with v. They only considered the sequence of the significant locations instead of all locations to build higher order MM. *Rathore* et al. [21] proposed approach with a mixed Markov model (MMM)-based scheme and a trajectory clustering, called *NETSCAN* *based TP* method for both short- and long-term trajectory predictions. Because of the existing methods for frequent sequential pattern mining tend to be limited to predicting short-term partial

trajectories, *Tiwari* et al. [22] designed a scalable route prediction application based on a prediction by partial match (PPM) modelling of user travel data. Even, *Qiao* et al. [23] suggested a prefix-projection-based trajectory prediction algorithm called *Prefix-TP* for predicting long-term trajectories of connected vehicles.

3.2. Artificial neural networks approach

Neural Networks have been also applied for prediction. *Liu* et al. [24] compared between Personalized Markov-Chain (FPMC) and Tensor Factorization (TF) against Recurrent Neural Networks (RNN). They found that (RNN) model shows promising performance comparing with PFMC and TF, but all these methods have a problem in modelling continuous time interval and geographical distance. So, they extended RNN and proposed a novel method called *Spatial Temporal Recurrent Neural Networks (ST-RNN)*. ST-RNN can model local temporal and spatial contexts in each layer with time-specific transition matrices for different time intervals and distance-specific transition matrices for different geographical distances Moreover, *Mikuscak* et al. [25] discussed several algorithms and methods which have been used in intelligent transport systems (ITS). They proposed a route prediction method based on artificial neural networks using the past routes of a vehicle.

3.3. Clustering-based approach

Several researchers have relied on trajectory clustering in route prediction methods, which divide the trajectories into many clusters representing different motion patterns based on the trajectory similarity. *Cao* et al. [26] proposed a model called t*rajectory-clustered Markov* model (tra-MM) that exploits the similarity between trajectories. In tra-MM first, they clustered similar trajectories according to a given similarity metric and then for each cluster, they trained a variable-order Markov model using the trajectories contained. Also, *Terroso-Saenz* et al. [27] presented online route prediction based on clustering of meaningful velocity-change areas called *Prop-Turn*. The framework integrates route learning and the prediction algorithm in an on-line manner. By means of a thin-client and server architecture, it also puts forward a new concept for route abstraction based on the detection of spatial regions where certain velocity features of routes frequently change

4. Taxonomy of route prediction

To review the existing route prediction models, we provide, in this section, a taxonomical classification that categorizes the prediction models according to the technique used followed by a brief comparison between a set of works by highlighting their weakness and strength aspects.

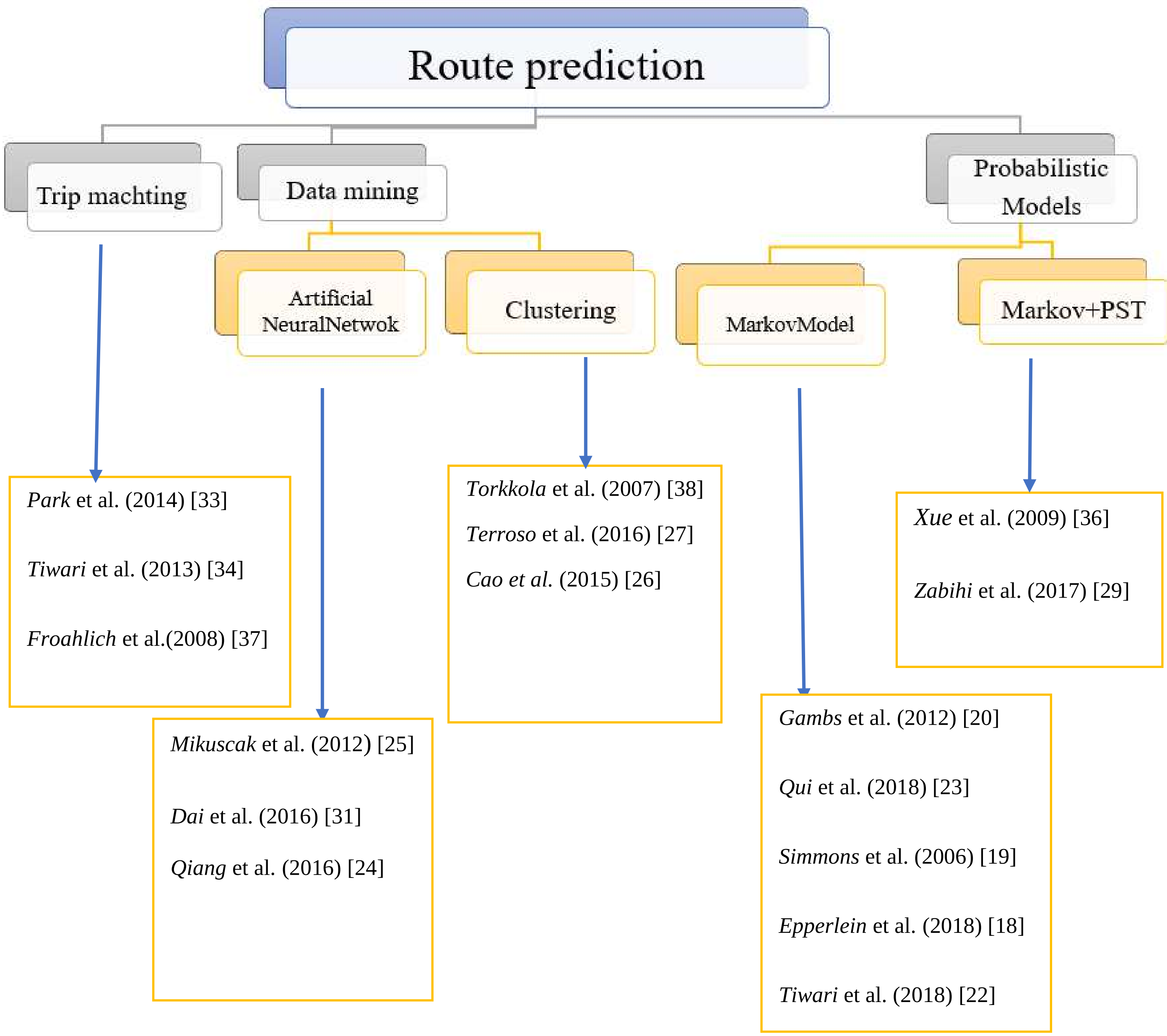


Fig.12 Taxonomy of route prediction

5. Literature review for route prediction studies

In this section, we present a table that clarifies the different route prediction algorithms used by researchers and its weakness and strength sides.

Author	Technique(s) Used	Strength	Weakness
Amirat et al. (2019) [28]	CPT model for accurate route prediction	Successful prediction ratio between 70% and 98% Failure prediction less than 30% and a high coverage of 80 to 93%	The size of CPT is verry huge
Zabihi et al. (2017) [29]	Autoregressive Input-Output HMM	Accuracy is 80% in real-time	
Gross et al. (2016) [30]	Random Decision Forests (RDF) Support Vector Machine (SVM)	RDF achieves a TP@5FP of 59 % at a TTI of 2.88 s which performs better than support vector machine where (TP@5FP) is a major performance measure and TTI is worst-case times to the reference node	Not applicable for real-world conditions and live testing
Dai et al. (2016) [31]	Machine learning for building personal driving route knowledge system First order Markov Re-direct	Redirect the driver's intended route when the driver deviates from the predicted route.	Information such as traffic flow, weather conditions and other conditions
Mingjun et al. (2014) [32]	Improved Dijkstra's Algorithm	Provides reasonable shortest route for drivers	
Park et al. (2014) [33]	Intelligent Trip Modelling System (ITMS) through Machine Learning	Performance of ITMS is robust when Cross-region variance is considered	It cannot be used where traffic sensors are unavailable
Tiwari et al. (2013) [34]	Road networks Map Matching	The algorithm runs faster, by two folds without affecting the accuracy of the output	

Goh et al. (2012) [35]	Hidden-Markov model (VSW-method)	Higher accuracy when compared to Fixed Sliding Window (FSW)	Interpolating trajectory points should be considered rather than the shortest path
Xue et al. (2009) [36]	Variable-order Markov models (VMMs) Probabilistic Suffix Tree (PST)	The enormous impact of different traffic conditions is taken into consideration	Poor scalability as the approach is centralized
Simmons et al. (2006) [19]	Hidden-Markov model (HMM)	Accuracy is above 98% for all transition that are Forced transitions	Accuracy drops to 70% when Unforced transition is considered

Table.2 Different route prediction algorithms.

6. Conclusion

In this chapter, we have presented an overall description of route prediction and its main applications in real-life. Then, we have given a review on related work on this topic and we have provided a taxonomy that classifies some works according to the technique used. To conclude, we have given a brief comparison between a set of existing works by highlighting their strong and weak aspects. In the next chapter, we will discuss how to add the temporal context in route prediction models.

Chapter III

PreNext and PreGraph models

1. Introduction

Spatial and temporal contextual information play key roles in route prediction studies and helps predicting where he or she will go next. These factors are fundamental to model human behaviors in practical applications. It is challenging and essential to predict where the driver will be at a given time with complex temporal and spatial information. Besides, most of prior proposals have only focused on spatial and social influences. For this reason, we propose in this chapter two novel route prediction models called PreNext and PreGraph respectively that are based on the temporal and spatial properties of driver movement.

This chapter consisted of two main parts. In the first part, the PreNext model that depends on CPT model is presented. PreNext thus offers all the advantages of CPT including: conserving all the data to perform prediction (i.e. lossless model), requiring less storage space, predicting rare cases with a high accuracy, and dealing with noisy mobility data (e.g. GPS) [6].

In the second part, the description of our second model PreGraph that is inspired by the dependency graph (DG) predictor, is given. PreGraph represents roads as a graph node that are used then to predict the next visiting road.

Both of the proposed models are expanded to consider the temporal context associated with each mobility sequence.

This chapter is organized as follow. First, a set of preliminaries that describe some interesting definitions related to the problem of route prediction is given then a presentation of PreNext and PreGraph models is provided.

2. Preliminaries

Definition 1 (Temporal mobility sequence)

The mobility sequence *Ms* = $\langle t_1 \rangle r_1, \langle t_2 \rangle r_2, \dots, \langle t_n \rangle r_m$ is a set of road segments traversed by a driver where each item in *Ms* represents the road traversed by the vehicle at a specified time. For example, Table.3 shows a sample of three mobility sequences performed by three vehicles *V: {V1, V2, V3}*, in three days, representing for each vehicle its own route represented as a list of a road segments. For instance, vehicle *V1* has visited the road r_1 at time t_1 followed passed by r_4 at time t_4.

Vehicle ID	Day	Mobility sequence
V_1	D_1	<$\langle t_1\rangle$>r_1, <$\langle t_2\rangle$> r_2, <$\langle t_3\rangle$> r_3, <$\langle t_4\rangle$> r_4.
	D_2	<$\langle t_1\rangle$>r_2, <$\langle t_2\rangle$> r_3, <$\langle t_4\rangle$> r_4, <$\langle t_5\rangle$> r_5.
	D_3	<$\langle t_1\rangle$> r_1, <$\langle t_3\rangle$> r_4, <$\langle t_4\rangle$> r_5, <$\langle t_5\rangle$> r_3.
V_2	D_1	<$\langle t_2\rangle$> r_2, <$\langle t_4\rangle$> r_4, <$\langle t_4\rangle$> r_3, <$\langle t_5\rangle$>r_5.
	D_2	<$\langle t_1\rangle$> r_1, <$\langle t_2\rangle$> r_2, <$\langle t_3\rangle$> r_3, <$\langle t_4\rangle$> r_4.
	D_3	<$\langle t_1\rangle$> r_4, <$\langle t_3\rangle$> r_2, <$\langle t_3\rangle$> r_3, <$\langle t_4\rangle$> r_4.
V_3	D_1	<$\langle t_1\rangle$> r_1, <$\langle t_2\rangle$> r_2, <$\langle t_3\rangle$>r_3, <$\langle t_4\rangle$> r_5.
	D_2	<$\langle t_1\rangle$> r_2, <$\langle t_2\rangle$> r_5, <$\langle t_3\rangle$> r_3, <$\langle t_4\rangle$> r_4.

Table.3. A sample of mobility sequences and histories.

Definition 2 (Time binning)

Time binning is the process that discretizes the time using time-slots (bins). It consists of dividing a day into multiple bins of equal length e.g. [8:00-10:00],]10:00-12:00] for 2-hour bins. For instance, Table.3 shows temporal mobility sequence generated using bins of two hours where

t_1= [6:00-8:00], t_2=]8:00-10:00], t_3=]10:00-12:00], t_4=]12:00-14:00], t_5=]14:00-16:00].

Definition 3 (Road pattern)

A road pattern Rp_i is a subsequence of some temporal mobility sequences in a road history *RH* that contains the set of all mobility sequences. For example, from road history depicted in Table 3, the sequence <t_1>r_1, <t_2> r_2 is a road pattern.

Definition 4 (Temporal graph)

A temporal graph $G = \{N, A\}$ is a directed graph where N is the set of nodes in G and A represents the set of arcs attaching the nodes. In the graph G, each node represents the road visited by each driver associated with its traversing time.

Definition 5 (Route prediction problem)

Let $TMs= \{Ms_1, Ms_2, Ms_3..., Mst\}$ be a set of training mobility sequences used to train a prediction model *M*. The problem of route prediction consists in predicting the next road segment with its traversing time $\langle t_{n+1}\rangle r_{n+1}$ segment with its traversing time of a given mobility sequence Ms_k=< $\langle t_1\rangle r_1, \langle t_2\rangle r_2, \dots, \langle t_n\rangle r_n$> by using *M*.

3. Compact prediction tree model (PreNext)

Our first model PreNext is inspired by the compact prediction tree CPT predictor but it is extended so it considers the temporal context. As previously mentioned, (in chapter 1), CPT is a lossless model that relies on two main processes: A) training phase, and B) prediction phase and that implements three main data structures: Prediction tree *(PT)*, Inverted Index *(II)* and Lookup Table *(LT)*.

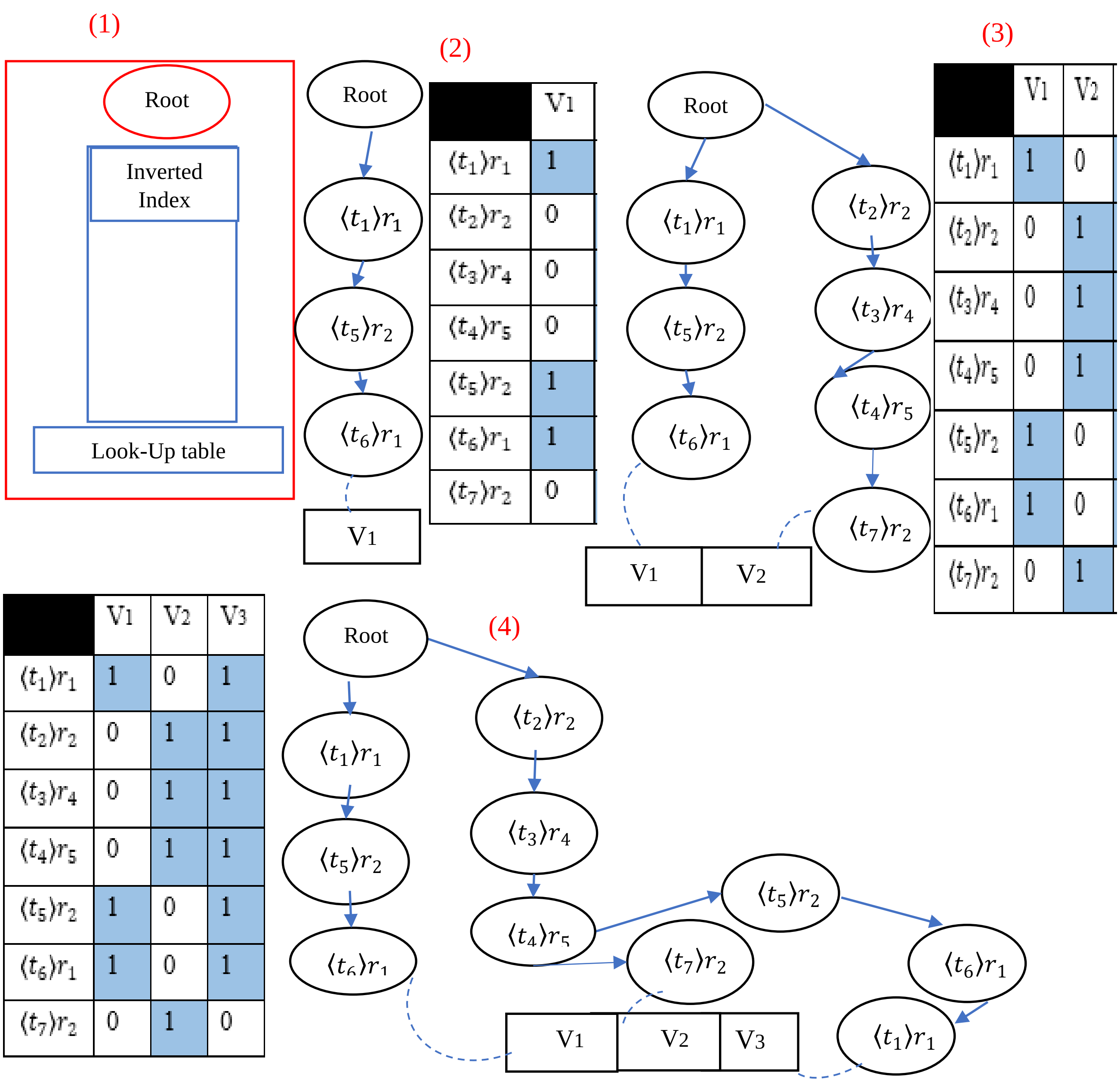
3. 1. Training phase

In this step, the three data structures are constructed by inserting the mobility sequences. Each node of the prediction tree *(PT)* represents a road segment whereas each path starting from the root to the last road visited by vehicle represents a mobility sequence. The Lookup Table cells are created whenever a new sequence is inserted in the *(PT)* where each leaf of *(PT)* related to cell of *(LT)* and the cells are formed by the order in which the sequences appear. Finally, the Inverted Index (II) is employed to indicate for each sequence if the road $\langle t_i\rangle r_j$ included in that sequence. Thus, the bit that corresponds to this road segment in the vector of the sequence in the *II* is set to 1 if the sequence contains the road segment and 0 otherwise. Hence, *II* is designed to quickly find in which sequences the given segment appears [1]. For sake of illustaion, Fig.13 depicts the training process of the following mobility sequences:

<$\langle t_1\rangle r_1, \langle t_5\rangle r_2, \langle t_6\rangle r_1 >$,$< \langle t_2\rangle r_2, \langle t_3\rangle r_4, \langle t_4\rangle r_5, \langle t_7\rangle r_2 >$ and <$\langle t_2\rangle r_2, \langle t_3\rangle r_4, \langle t_4\rangle r_5, \langle t_6\rangle r_1, \langle t_5\rangle r_2$>.

where step(1) represents the initial state with the three main structures *(PT)*,*(II)* and *(LT)* which were empty .Then, the steps 2,3, and 4 represent the model after the insertion of the mobility sequences one by one .Depending on the dataset used, CPT takes more or less space, however, the training process is really fast (O(n)),because of the following strategies are applied[6]. To further reduce the time and space complexity, CPT adopts two strategies namely *(Frequent Subsequence Compression)* and *SBC (Simple Branches Compression)*.

In many situations, sequences may share common sub-sequences. To deal with these cases, the *FSC* is applied to compress frequent sub-sequences which are the sequential patterns that are found in training sequences. Each sub-sequence found is replaced in (*PT)* by a single new segment identifier *Ri* stored in a new structure named *Subsequence Dictionary (DCF)* and associated with the subsequence that it replaces. Besides, the SBC strategy requires the replacement of each simple branch (defined as a branch leading to a single leaf) by a single node representing the whole branch. Fig.14 illustrates the application of these strategies on the *PT* of Fig.13. By applying *FSC*, the frequent subsequence $<\langle t_5\rangle r_2, \langle t_6\rangle r_1>$is replaced with one node *R₁*. Then the *(PT)* is further compressed by replacing the whole branch $(R_1, \langle t_1\rangle r_1)$ with a single node using *SBC* strategy.

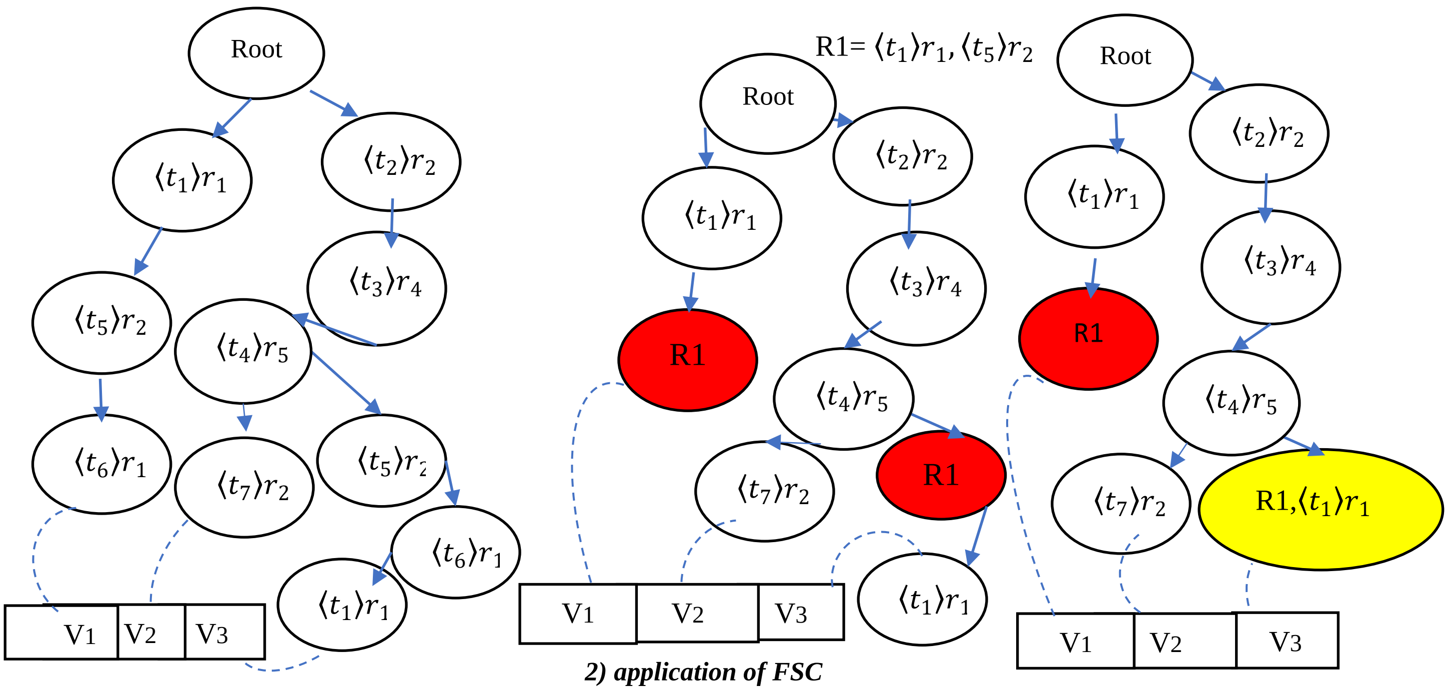


	V1
$\langle t_1\rangle r_1$	1
$\langle t_2\rangle r_2$	0
$\langle t_3\rangle r_4$	0
$\langle t_4\rangle r_5$	0
$\langle t_5\rangle r_2$	1
$\langle t_6\rangle r_1$	1
$\langle t_7\rangle r_2$	0

	V1	V2
$\langle t_1\rangle r_1$	1	0
$\langle t_2\rangle r_2$	0	1
$\langle t_3\rangle r_4$	0	1
$\langle t_4\rangle r_5$	0	1
$\langle t_5\rangle r_2$	1	0
$\langle t_6\rangle r_1$	1	0
$\langle t_7\rangle r_2$	0	1

	V1	V2	V3
$\langle t_1\rangle r_1$	1	0	1
$\langle t_2\rangle r_2$	0	1	1
$\langle t_3\rangle r_4$	0	1	1
$\langle t_4\rangle r_5$	0	1	1
$\langle t_5\rangle r_2$	1	0	1
$\langle t_6\rangle r_1$	1	0	1
$\langle t_7\rangle r_2$	0	1	0

Fig. 13 An example of the CPT training process



2) application of FSC

1) PT without

3) application of FSC and SBC

Fig.14 Application of *(FSC)* and *(SBC)*.

3. 2. Prediction phase

As previously mentioned in chapter 1, the prediction process of CPT predictor deals with the three data structures in addition to a new structure called Count Table (*CT*), which holds the candidate roads and their corresponding scores. For instance, let us consider the mobility sequences *Ms*=$< \langle t_1 \rangle r_1, \langle t_2 \rangle r_2, \dots, \langle t_m \rangle r_n >$ containing *n* road segments with their traversing time. We define *LTRS*$_L$*(Ms)*= $<\langle t_{n-L+1} \rangle r_{n-L+1}, \langle t_{n-L+2} \rangle\ r_{n-L+2},.., \langle t_n \rangle .r_n>$ as the Last Traversed Road Segments of *Ms* of size *L* with *1≤L≤n*. The *L* parameter determines the number of road segments in a path to be considered for predicting the next road segment. The prediction of the next segment of a sequence *Ms* is performed as follows. First, the similar sequences to *Ms* among all the mobility sequences that contain all the road segment of *LTRS*$_L$ *(Ms)* are retrieved by using the intersection of the bitset of *II* which contain the road segments of *LTRS*$_L$ *(Ms)*. The results obtained from the intersection indicates the set all similar mobility sequence to *Ms*. Second, the matching sequences are extracted by using the *LT* structure that helps getting access to these sequences.

Finding the similar sequence can consider as a sensitive task. However, and in contrast to many prediction models, CPT can deal with noise occurred in movement data by applying two strategies called respectively: *Recursive Divider (RD)* and *PNR (Prediction with improved Noise Reduction)*[6].

a) *Recursive divider (RD)*

For a given travelled mobility sequence *TMs*, the *RD* strategy tries to find similar sequences of each subset *SB∈TMs* where |*SB*|=*k* with *k = {1, 2,..,max-Level}* and *max-level* indicates the maximum subset splitting size allowed for *TMs*. If a prediction cannot be made at level *k*, *RD* explores the *k+1* level if and only if *k+1< max-level.*

b) *Prediction with improved Noise Reduction (PNR)*

To gain more flexibility with noisy data, this strategy could be also employed relying on the hypothesis that noisy routes appeared in training sequences are the ones with low frequency *(support)*. Therefore, PNR aims to remove the road segments having a low support.

The next step in the prediction process is finding the consequent of each similar sequence. The consequent is defined as the longest sub-sequence after the last occurrence of $LTRS_L(Ms)$ in a similar sequence. For instance, given a mobility sequence *Ms*, and its $LTRS_L(Ms)$, a mobility sequence similar to *Ms* can be divided into three sub-sequences: the context subsequence comes in the first position followed by the $LTRS_L(Ms)$ and in the third position the consequent subsequence (as illustrated in Fig.15).

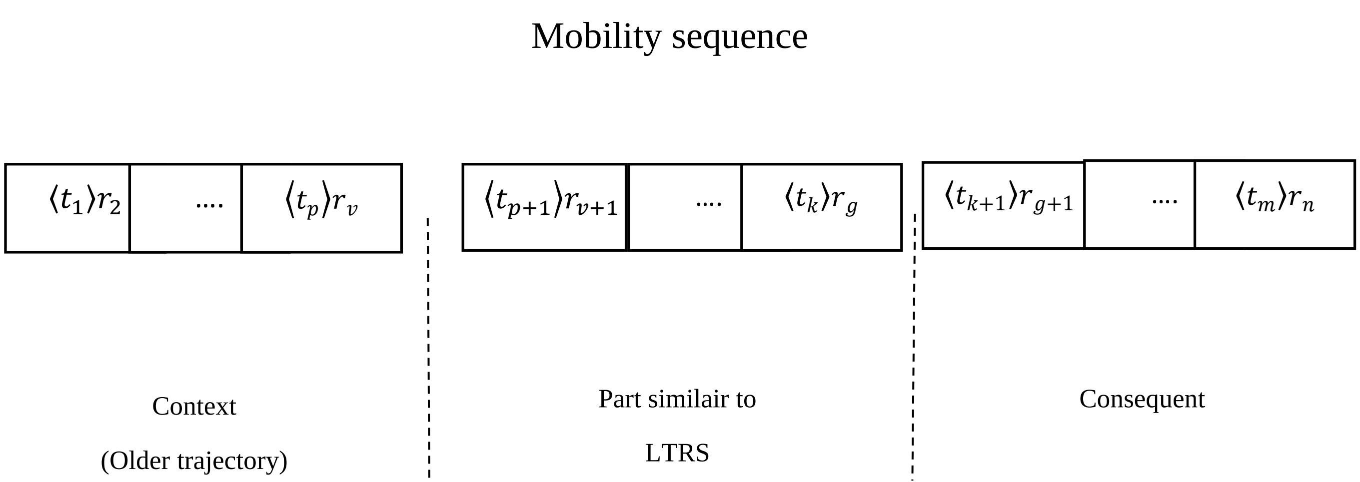


Fig.15 Mobility sequence consequent.

Each consequent of a sequence similar to *Ms* is stored in Count Table *(CT)* where road segments are keys and a score is associated as the value corresponding to each key. The score represents the support (frequency) of a given road segment$< \langle t_i \rangle r_j >$which is defined as the number of times it appears in the consequent of a mobility sequence similar to *Ms*. In the case where the supports of two road segments are equal, the confidence is used. It is defined as the support of$< \langle t_i \rangle r_j >$divided by the total number of sequences that contain $< \langle t_i \rangle r_j >$(the cardinality of the bitset of $< \langle t_i, r_j \rangle >$in the *II*).

4. System architecture

As shown in Fig.16, PreNext is composed of two main modules: pre-processing and route prediction.

4.1. Pre-processing

In this module, the GPS data (GPS trajectories) are transformed into mobility sequences by applying the three following steps. First, the set of stay points relaying on two predetermined space and time thresholds T_{thre} and D_{thre} is extracted. Stay points divide the trajectories into trips. The resulting trips are then converted into mobility sequences by map-matching GPS trajectories using a cloud map-matching based API [42].

4.2. Route prediction

To predict the next route of drivers, three processes are performed: training, predicting (or testing). Basically, the pre-processed data is divided into two subsets: one is used for training and the other for testing. The user of PreNext needs to set the size of the *LTRS* to be considered *(L)*. The prediction process will consider the last *(L+1)* road segments of each testing mobility sequence. The first *L* road segments are used to perform prediction based on CPT structures. The set of possible consequents are stored in the (*CT)* with their corresponding scores. The road segment having the highest score is then selected as the prediction result. Finally, the predicted segment is compared with the last road segment from the testing mobility sequence to determine if the prediction was a success or not [28].

The prediction model is adapted to consider both global (collective) and personal (individual) movement behaviors of drivers. Two models are proposed:

1) Global model *(GM)*

As people always show a degree of similarities in their movement while travelling from locations, this model is adapted to represent the global mobility behavior of persons. Therefore, the *GM* model is employed to perform predictions for a driver when no prior knowledge about his mobility patterns is found such as newly seen drivers in a geographic area by considering and using the mobility patterns of all drivers.

2) Personalized model *(PM)*

Sometimes, a single driver may also exhibit a specific personal movement relaying on her/his habits or desires. *PM* model consists of creating a mobility model for each driver comprising his previous trajectories. Accordingly, the prediction model is only trained with his mobility sequences rather than the data of all drivers.

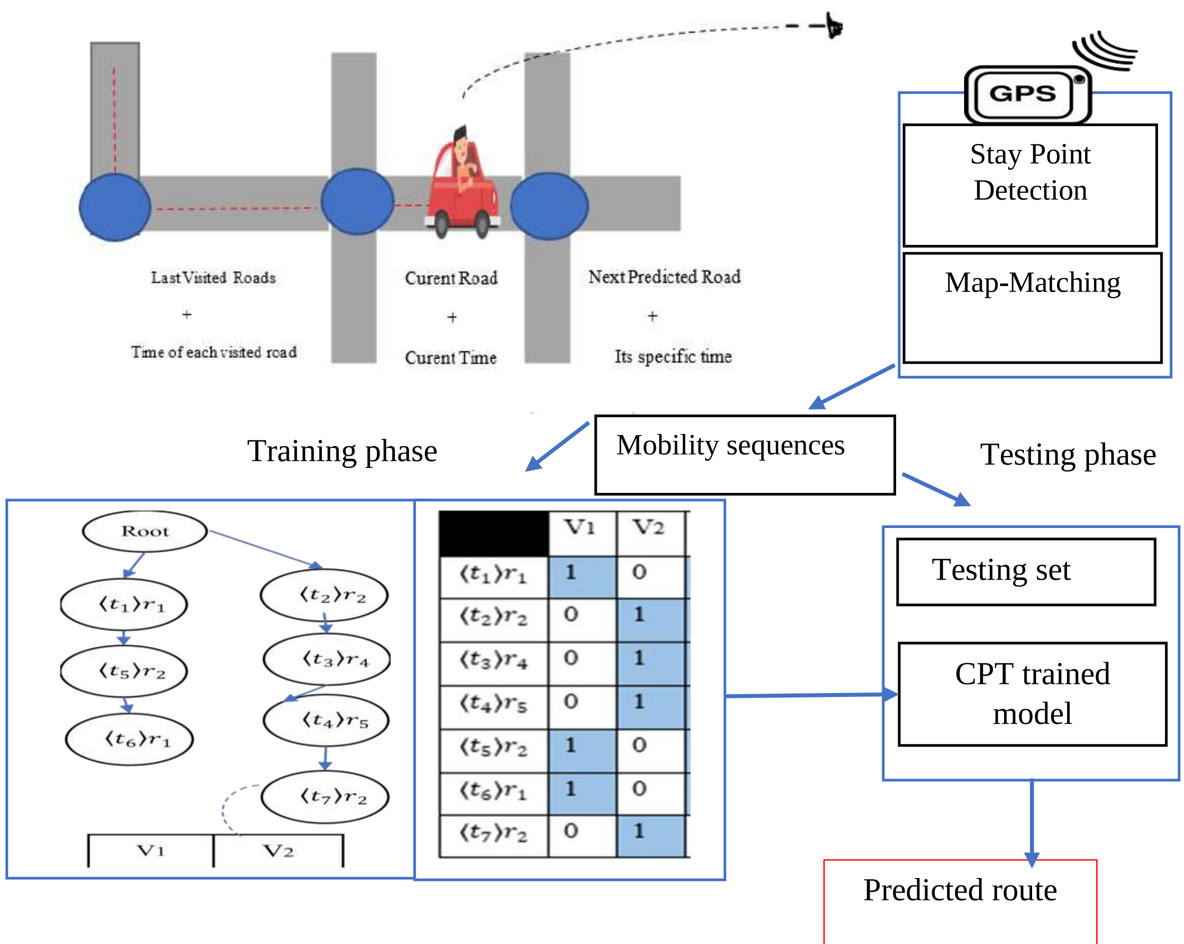


Fig.16 PreNext architecture.

5. Dependency graph model (PreGraph)

Our second proposed model is PreGraph. This latter adopts the Dependency graph (DG) predictor. PreGraph builds a graph structure named *PG*. Formally, the *PG* graph represents a pair of sets *(R, A)* where *R* is a set of road segments (nodes) and $A \in R \times R$ is a set of directional arcs (edges) representing movements on the road network. For a given arc (road segment) $a = R_x R_y$, the location from which the movement starts is called a *source* of *a* and is denoted as the *source (a)* = R_x, while the other node R_y is called *destination of* a, and is denoted as $Dest(a) = R_y$. In *PG*, an arc $a = R_x R_y$ arc is created if and only if R_y appears within *w* movement after R_x in a mobility sequence, where *w* is the *lookahead*, that determines the size of the window. Moreover, the weight value *(w(a))* is associated with each arc *a*, indicating the number of times that *Dest (a)* traversed by drivers after *Source(a)*. In the context of mobility prediction, *PG* allows representing order dependencies among traversed roads in mobility sequences by drivers where the source of a given arc $a_i \in A$ must appears before its destination. The mobility prediction graph is built by gradually inserting mobility sequences in the graph. In the case where sequences share common roads, the weights of the shared arcs are incremented rather than creating new arcs. Thus, significant space reduction can be achieved using the *PG* representation.

It is worth noticing that the *PG* graph is built using temporal mobility sequences that consider the traversing time of each road segment. This process is done by applying the time binning process that consists of dividing a day into multiple time-slots or bins of equal length. For instance, Fig.17 depicts the *PG* graph created using the set of mobility sequences of Table.4 with a lookahead window specified by *w* = 2 and using the *time_slot_size*=4.

6. System architecture

Depending on client-server architecture, PreGraph has been designed to perform route prediction where the client is the vehicle (driver) and the server is an authority infrastructure such as an RSU (road side unit). As depicted in Fig. 18, PreGraph comprises three main modules.

6.1. Data preparation

This module periodically collects driver location data (GPS records) and sends it to a server site. To transform this data into mobility sequences, three steps are required. These steps are the same presented in section (4.1. Pre-processing data for PreNext model). A sample of the output of this module is presented in Table.4

Vehicle	Mobility sequences
V1	$<\langle t_1\rangle r_1, \langle t_5\rangle r_2, \langle t_6\rangle r_1>$
V2	$<\langle t_2\rangle r_2, \langle t_3\rangle r_4, \langle t_4\rangle r_5, \langle t_7\rangle r_2>$
V3	$<\langle t_2\rangle r_2, \langle t_3\rangle r_4, \langle t_4\rangle r_5, \langle t_5\rangle r_2>$

Table .4 A set of time extended location sequences.

6.2. Graph construction

When the mobility sequences are extracted, the server gradually updates its mobility graph *PG*. At first the mobility graph is built and then extended by inserting new road segments as graph nodes, whereas vehicle movements between pair of road segment within the lookahead window size are represented by arcs. The weight of each new arc is set to 1. where a road segment appears on newly collected mobility sequences, the weight of the corresponding arc is incremented accordingly. Note that, in some situations where movement sequences are highly heterogonous and do not overlap (have any common road segments), the mobility graph is a disconnected graph. In this thesis, we assume that the mobility graph is connected, comprises a least two nodes, and has no isolated nodes. These assumptions hold in real-life for representing the mobility of vehicles in active urban areas.

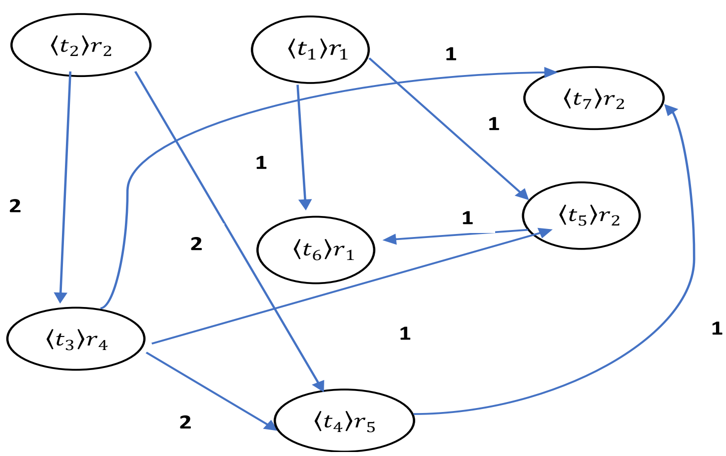


Fig.17 PG graph illustration.

6.3. Route prediction

Once the mobility graph is constructed, predictions can be performed using it. To predict the next route segment that will be visited by a driver *D*, its current trajectory called *CT (Current Trajectory)* is used. is used. *CT* comprises the current road segment where *D* is located in addition to its previous locations with their traversing time for the same trip. In this case, *CT* is written as *CT*= $\{\langle t_i \rangle r_i, \langle t_{i+1} \rangle r_{i+1}, \dots, \langle t_c \rangle r_c\}$.

Formally, let *RS*= $\{\langle t_1 \rangle r_1, \langle t_2 \rangle r_2, \dots, \langle t_m \rangle r_n\}$ be the set of all road segments in a road network *RN*. *CT*= $\{P_i, P_{i+1}, \dots, P_c\}$ is a sequence of road segments traversed by *D* where *Pc* is the current road segment of *D* with its traversing time. Having the trajectory *CT*, the prediction of the next route segment is performed in two steps.

6.3.1 Graph matching

Initially, the first step is to find a path *SP*= $\{S_i, S_{i+1}, \dots, S_m\}$ in the mobility graph that matches with *CT* where $S_i \in RS$. We say that *SP* matches *CT* if and only if each road segment in *SP* appears in the same order in *CT*, that is $\forall\ i, S_i = P_i$ and $m = c$. Note that graph matching is noise sensitive. This step can be challenging since erroneous positions may appear in location data. Using *PG*, built according to the lookahead window, more flexibility to handle noisy data could be obtained and this is one of the most important traits that this graph offers.

PG allows creating arcs not only between consecutive road segments (which may be noise) but also to the following road segments within the lookahead window. In the other hand, if the first road segment *Ne* that comes after a given node N_x in a mobility sequence *Ms* is considered as noise, another arc will be created that skips N_e and go directly to next road segment, given that $w \geq 2$. Unlike other Markov-based predictors such as PPM, the noise tolerance strategy achieved by using *PG* permits PreGraph to forecast the future route of *CT* that have not been previously seen in mobility sequences.

6.3.2 Next road extraction

The second step is to find the next road segment that a driver will visit according to *SP*, denoted as *Nr*. The latter is the road segment predicted as the destination of the arc having the highest weight emanating from the last road segment in SP.

For example, consider that the current trajectory of a driver is $CT = \{\langle t_2\rangle r_2, \langle t_3\rangle r_4\}$. Based on the mobility graph of Fig. 17, three candidate arcs are considered, which are a_1 $(\langle t_3\rangle r_4, \langle t_7\rangle r_2)$ and a_2 $(\langle t_3\rangle r_4, \langle t_4\rangle r_5)$, a_3 $(\langle t_3\rangle r_4, \langle t_5\rangle r_2)$. These arcs have weights of 2 and 1, respectively. Therefore, the next route segment is predicted to be $Nr = a_2$ (the destination of a_1). In case where the candidates' segments have equal weights, many selection criterions could be employed such as retaining the road with highest frequency in mobility.

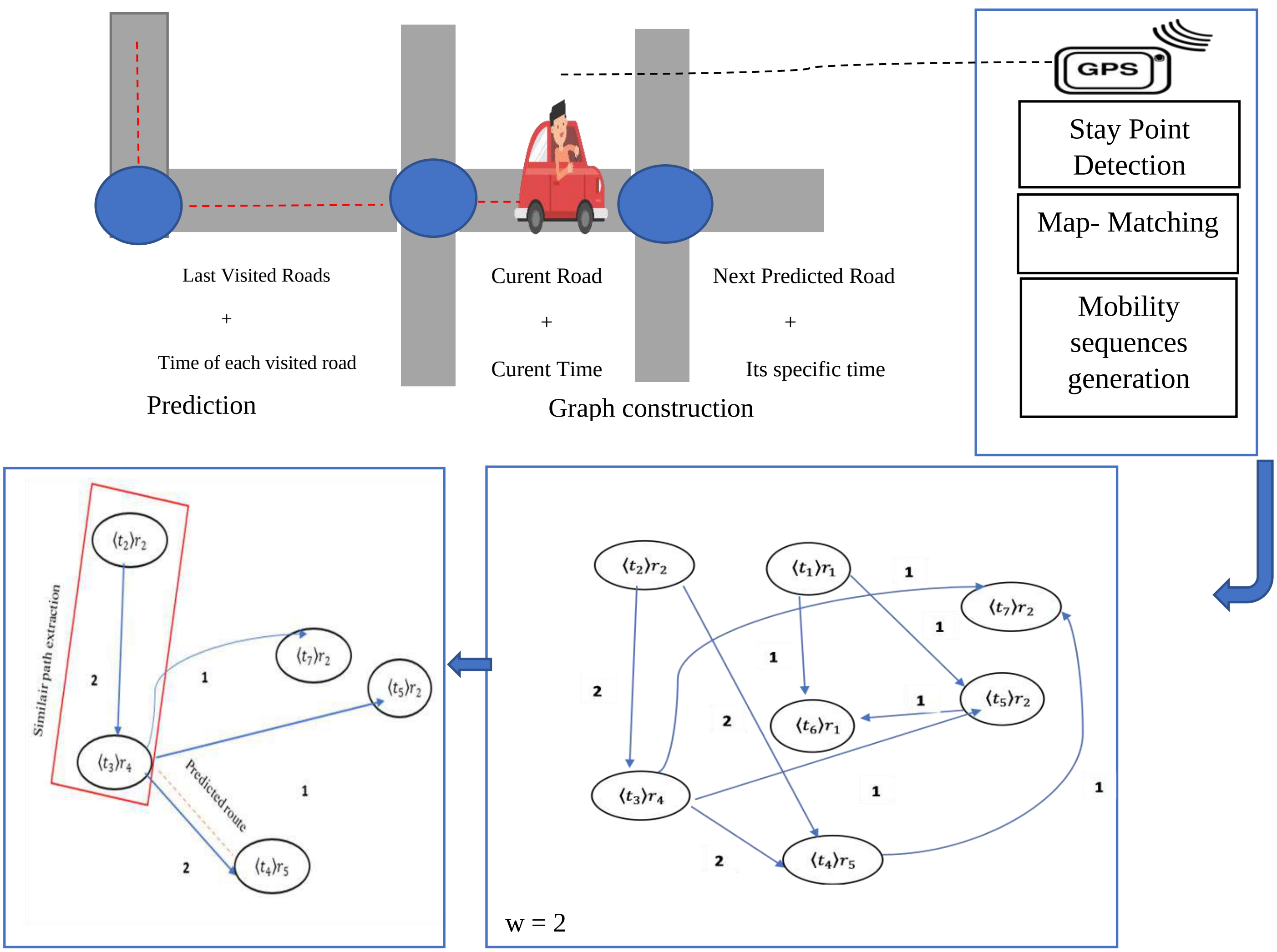


Fig .18 PreGraph architecture.

7.Conclusion

In this chapter, we have proposed two models called PreNext and PreGraph. While PreNext is basically depends on the CPT predictor; the PreGraph model is inspired of the dependency-graph predictor. These two models perform route prediction, and present the core components of our proposals. Note that in this thesis, we have only dealt with time information as a context. However, our proposals can be easily extended to consider other contextual data associated with navigation data.

Chapter IV

Experimental evaluation

1 Introduction

This chapter describes the evaluation of the proposed PreGraph framework, an extensive experimental evaluation was carried out. First, a small description of the datasets and experimental settings, followed by the evaluation metrics to measure the model's performance. Also, for better study, PreGraph performance is compared with first-order Markov PPM and the LZ predictor. Besides, the proposed model is compared also with the based model without time factor (DG).

The source code of the compared prediction models, as well as our models, can be downloaded as part of the SPMF library *Fournier-Viger* et al [41]. Finally, this chapter presents the conducted experiments and the corresponding results.

2. Datasets

Experiments were performed on two public large-scale real datasets from Brightkite [39] and Gowalla [40], respectively. These two datasets have been widely used for prediction and location recommendation. Each dataset was split between a training set and a testing set based on K-fold cross validation with k=10.From the set of the location of drivers, the location histories were generated each composed of several daily sequences. In each sequence, the repeated consecutive locations were removed. The cause is that such repetition is not a transition between two locations.

3. Experimental Settings

Experiments were performed on a computer equipped with a Dual-Core Intel CPU 1.60 GHz, 4GB of RAM and 250GB of Hard Disk. The proposed model was implemented in Java by using the DG implementation available in the SPMF open-source data mining library [41].

4. Evaluation Metrics

To evaluate the performance of the proposed prediction models, two popular and widely used measures were employed: accuracy and coverage.

- **Overall Accuracy**

It is defined as the number of successfully predicted routes, divided by the total number of testing mobility sequences.

$$Overall\ Accuracy = \frac{number \text{ of successful predictions}}{\text{number of testing mobility sequence}}.$$

- **Coverage**

It is the number of test mobility sequences where a matching path was found for each current trajectory of a driver divided by the total number of test sequences.

$$Coverage = \frac{\text{number of matching path}}{\text{number of testing mobility sequence}}.$$

5. Experiments

In this section, we present the evaluated models and conduct an analytical comparison between the PreGraph performance and these models. Also, the impact of varying some important parameters on the performance was shown.

5 .1 Evaluated models

PreGraph was compared with the following prediction models (see their detailed description in chapter 1):

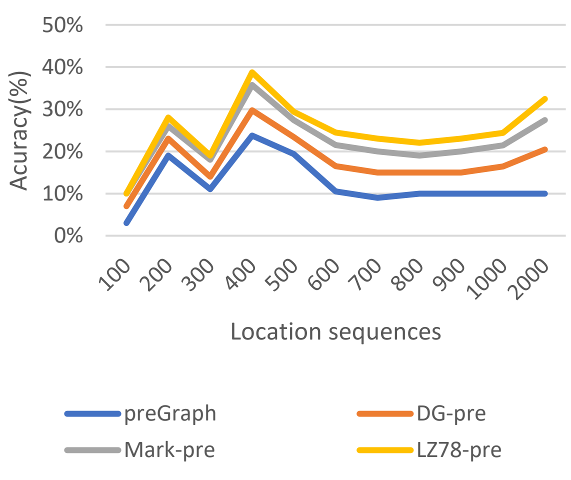
- ***DG-Pre (Dependency graph predictor).*** DG-pre is a standard graph-dependency model without temporal context consideration. This predictor was initially designed for web prefetching. It is used to predict the next webpage that a user will visit based on previously visited webpages by the user and other users [7].
- ***Mark-Pre (First Order Markov Predictor).*** This predictor utilizes first-order Markov chain which only uses the last road visited by drivers to predict the next road.
- ***LZ-Pre (Lempel-Ziv Predictor).*** This model is similar to k-order Markov predictor except that the k is a parameter that can grow to infinity.

Experiment 1. Time factor consideration and comparison with other models

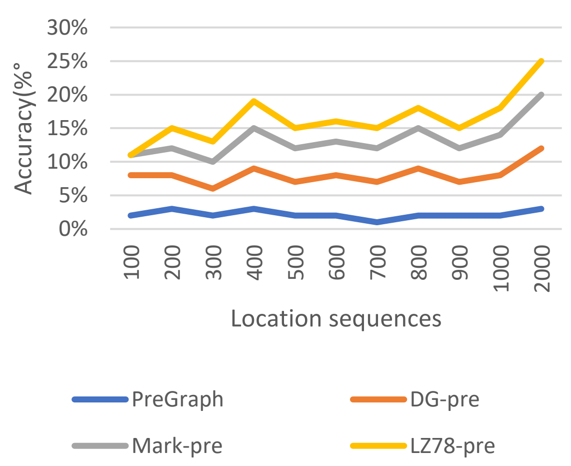
In this experiment, we assess the performance of our predictor with and without time consideration. In other words, our predictor PreGraph is compared with DG-pre model. In this experiment, we also aim to compare the performance of PreGraph with others models using Britgherkite and Gowalla datasets.

The result depicted in Fig.19 have shown that PreGraph accuracy is the lowest compared with DG-Pre and the other models in both cases with Britgherkite and Gowalla datasets. This may due to the fact that although human mobility is characterized by its temporal regularity, each person still has his own habits. A given person may visit the same set of locations already visited by other persons but not necessary at the same time or in the same order. For example, a person may visit a popular shopping mall just like some other people but he may not visit it at exactly the same time. In addition, the lower performance of PreGraph is also due to the frequencies of temporal patterns that are less than for standard location patterns. By extending location sequences to consider temporal data, the frequencies of location patterns decrease and thus, the prediction quality is influenced. As for the coverage, reults have demonstrated that PreGraph and other models have achieved good coverage values that increses by increasing the number of mobility data considred.

(a) Accuracy (Britgherkite dataset).



(b) Accuracy (Gowalla dataset).



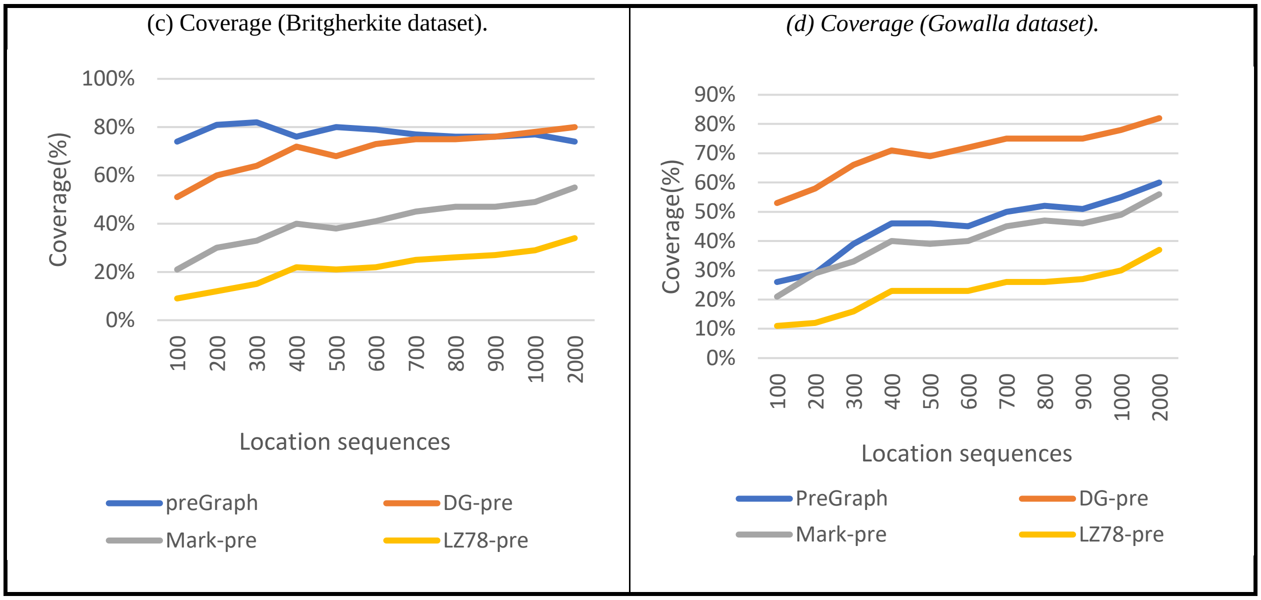


Fig.19 Comparing PreGraph with DG-pre, Mark-pre and LZ-pre.

5.2 Parameters Effect

This set of experiments assesses to evaluate the influence of varying the parameters: *time slot length* and *lookahead size*.

Experiment 2. Impact of varying the time-slot length

In this experiment, the size of time slots (bins) that used to construct temporal sequences were varied. Each day is divided into several periods of a predefined length. From the results depicted in Fig.20, it can be observed that overall, the prediction quality (accuracy and coverage) is improved when the size of time slots is increased. The reason behind these results is that by increasing the time slot length, more temporal flexibility is provided. Thus, the driver has more time to visit the predicted roads but not necessary in the same exact time. This finding helps the system to make more accurate prediction by providing tolerance to the time of taking a specific road segment. Consequently, the time bin that provides the best accuracy is one of the largest ones (time slot size=10 for Britgherkite and size=12 for Gowalla). As far as, the coverage is over than 30% for both cases. The coverage is an important metric as it measures how often a prediction model can make a prediction. It is best if a predictor can make a prediction even if it is a failure. This is because additional work could then be done to increase the accuracy of the model to fit the desired output or parameters to turn a failed prediction into a successful one.

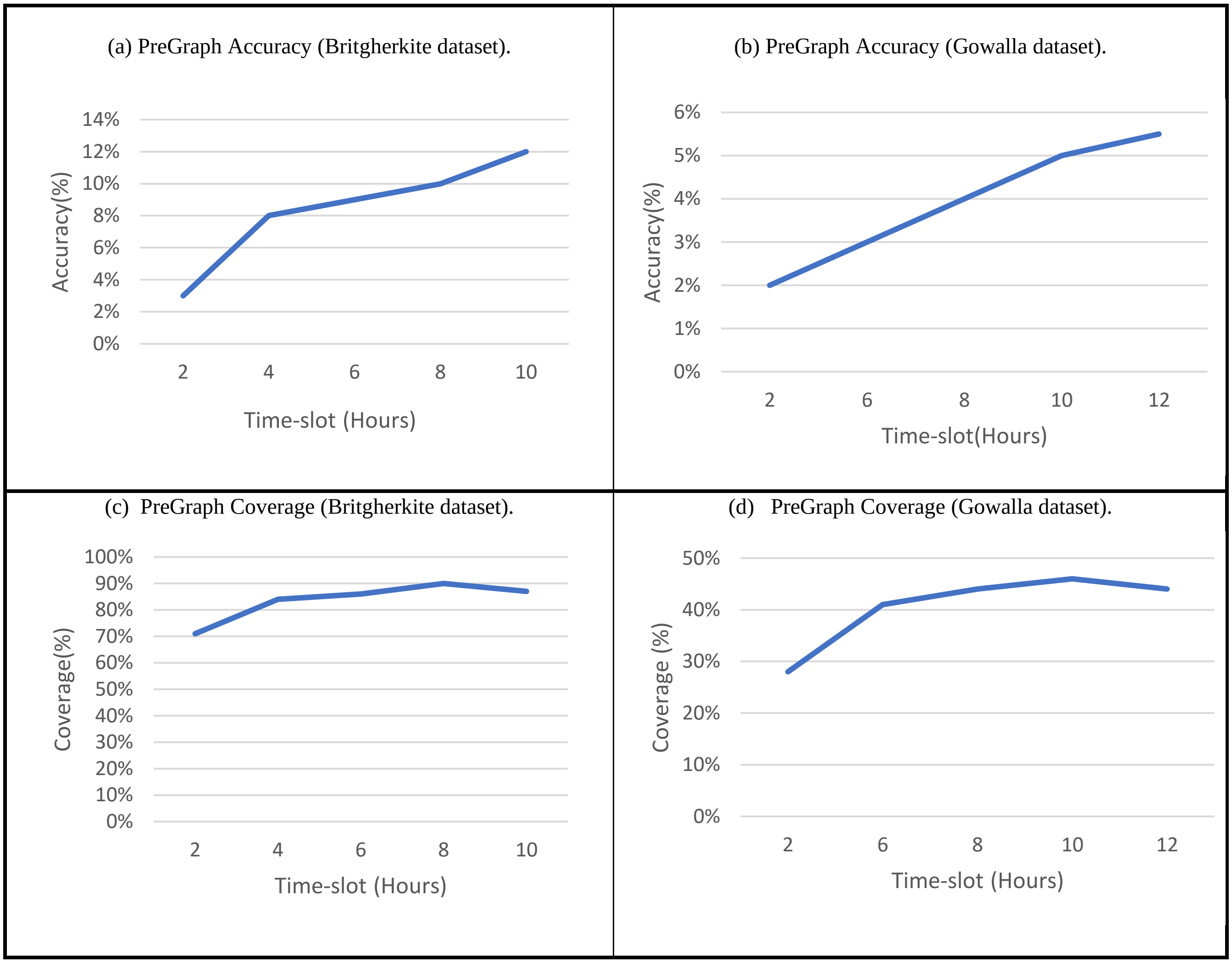


Fig.20 The impact of varying the time-slot.

Experiment 3. Impact of varying the lookahead size

In Fig.21, we evaluate the impact of varying the lookahead size on prediction performance. In this experiment, the window size was varied from 1 to 10. Increasing the lookahead size by 1 means that additional arcs are created not only to the next road segment but to a number of *lookahead* upcoming roads.

Overall, results indicate that setting the lookahead size to 2 gives the best results specially with Brightkite dataset and that by increasing the lookahead size a small improvement are obtained. This observation confirms the assumption that the system provides more accurate results in short-term predictions, comparing to the long-term, meaning that our system predicting the next route of the driver is more beneficial than may visit rather than later. Besides, results also show that *lookahead* transitions allow creating additional links between road segments, which often permits performing predictions by skipping segments otherwise no prediction could be done which boot the prediction coverage.

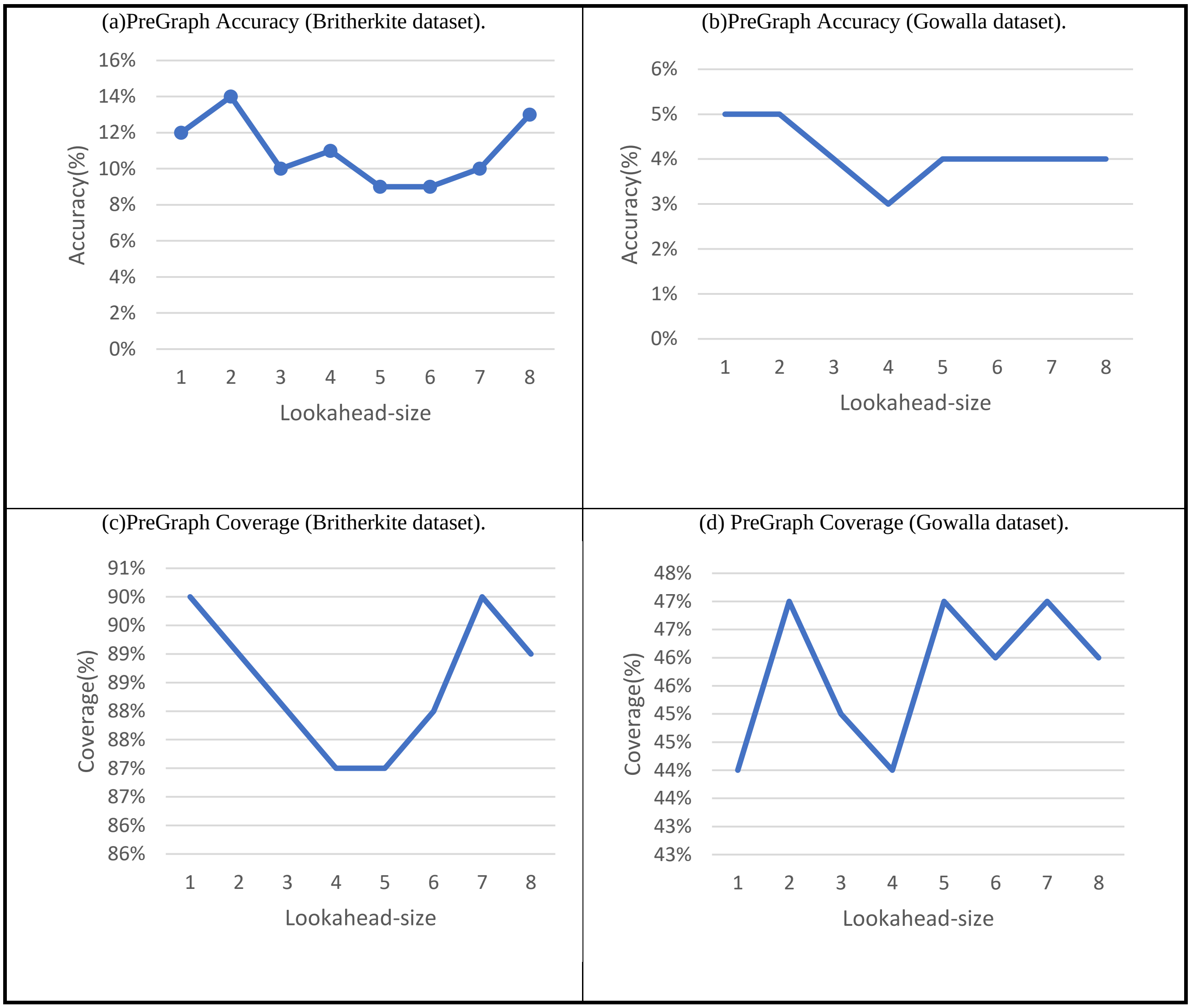


Fig.21 The impact of varying the lookahead-size.

6.Conclusion

Our experimental evaluation has demonstrated that PreGraph has shown acceptable and exhibits promising results that outperform a set of state-of-the-art models in all proposed experiments for both types of datasets (Britgherkite and Gowalla). Experiments also demonstrate the improvements in accuracy obtained by focusing on short term prediction. Therefore, for the future works, we are willing to predict not only the next location (short term prediction) but also other upcoming location (long-term prediction) in addition to the person's final destination. And several improvements will be considered such as extending the proposed mobility graph to regard day-of-week and other contextual factors (i.e. weather, traffic congestion level, etc.). So, it will become possible to distinguish between trips relying not only on spatial data, and time of day but also time and other contextual information.

General Conclusion

In this thesis, we have proposed two route prediction systems called PreNext and PreGraph. These two proposals are designed to consider both the spatial and temporal factors of human mobility.

The first proposed model PreNext is mainly depend on the CPT (compact prediction tree) model, thus it offers all its advantages including 1) its lossless property that allows conserving all the data to perform prediction, 2) its lower storage space requirement, and 3) its ability to predict rare cases with high accuracy.

Our second proposal (PreGraph) utilizes a graph representation of sequences where nodes are locations and each arc represent the visiting order between two locations. A distinctive characteristic of the proposed prediction graph structure is the creation of additional links to upcoming road segments using a lookahead parameter that allows increasing the prediction coverage.

In this thesis, an extensive experimental study was carried out using two real-world datasets. Results have shown that the proposed model has exhibits promising results compared to state-of-the-art models.

Despite the promising results, several improvements, additions, and prospects are possible. Therefore, as a future work, we plan to extend the PreGraph and PreNext models so they consider 1) more temporal information such as day-of-week, and 2) other contextual factors (i.e. weather, traffic volume for upcoming routes, etc.). In addition, we aim to extend the models so they perform long term prediction (to predict the later upcoming roads for a driver not only the next).

Bibliography

Bibliography

[1] E. Bas and E. Bas, "An Introduction to Markov Chains," *Basics Probab. Stoch. Process.*, pp. 179–198, 2019, doi: 10.1007/978-3-030-32323-3_12.

[2] E. Salinelli and F. Tomarelli, "Markov chains," *UNITEXT - La Mat. per 3 piu 2*, vol. 76, pp. 227–255, 2014, doi: 10.1007/978-3-319-02291-8_6.

[3] D. Jacobs, *Markov chains and Markov chains and HMMs*, vol. Markov Pro. 2013.

[4] J. Gutierrez, *Selected readings on telecommunications and networking*. 2008.

[5] K. GOPALRATNAM and D. J. COOK, "Active Lezi: an Incremental Parsing Algorithm for Sequential Prediction," *Int. J. Artif. Intell. Tools*, vol. 13, no. 04, pp. 917–929, 2004, doi: 10.1142/s0218213004001892.

[6] T. Gueniche, P. Fournier-Viger, and V. S. Tseng, "Compact prediction tree: A lossless model for accurate sequence prediction," *Lect. Notes Comput. Sci.*, vol. 8347 LNAI, no. PART 2, pp. 177–188, 2013, doi: 10.1007/978-3-642-53917-6_16.

[7] N. U. Edakunni, A. Raghunathan, A. Tripathi, J. Handley, and F. Roulland, "Probabilistic Dependency Networks for Prediction and Diagnostics," pp. 0–12, 2015.

[8] T. Zimmermann and N. Nagappan, "Predicting subsystem failures using dependency graph complexities," *Proc. - Int. Symp. Softw. Reliab. Eng. ISSRE*, pp. 227–236, 2007, doi: 10.1109/ISSRE.2007.19.

[9] D. Katsaros and Y. Manolopoulos, "Prediction in wireless networks by Markov chains," *IEEE Wirel. Commun.*, vol. 16, no. 2, pp. 56–63, 2009, doi: 10.1109/MWC.2009.4907561.

[10] N. N. Zakaria, M. Othman, R. Sokkalingam, H. Daud, L. Abdullah, and E. A. Kadir, "Markov chain model development for forecasting air pollution index of miri, Sarawak," *Sustain.*, vol. 11, no. 19, pp. 1–11, 2019, doi: 10.3390/su11195190.

[11] F. Qiao, P. Li, X. Zhang, Z. Ding, J. Cheng, and H. Wang, "Predicting Social Unrest Events with Hidden Markov Models Using GDELT," *Discret. Dyn. Nat. Soc.*, vol. 2017, 2017, doi: 10.1155/2017/8180272.

[12] Q. Lv, Y. Qiao, N. Ansari, J. Liu, and J. Yang, "Big Data Driven Hidden Markov Model Based Individual Mobility Prediction at Points of Interest," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5204–5216, 2017, doi: 10.1109/TVT.2016.2611654.

[13] L. Sael and D. Kihara, "Binding Ligand Prediction for Proteins Using Partial Matching of Local Surface Patches," *Int. J. Mol. Sci.*, vol. 11, no. 12, pp. 5009–5026, 2010, doi: 10.3390/ijms11125009.

[14] Y. T. Tan and B. A. Rosdi, "FPGA-based hardware accelerator for the prediction of protein secondary class via fuzzy K-nearest neighbors with Lempel-Ziv complexity based distance measure," *Neurocomputing*, vol. 148, pp. 409–419, 2015, doi: 10.1016/j.neucom.2014.06.001.

[15] L. Cui, X. Gong, J. Zhang, and H. Wang, "Double-dictionary matching pursuit for fault extent evaluation of rolling bearing based on the Lempel-Ziv complexity," *J. Sound Vib.*, vol. 385, pp. 372–388, 2016, doi: 10.1016/j.jsv.2016.09.008.

[16] D. B. Mallick and K. Raj, "Weather prediction using CPT+ algorithm," *Int. J. Eng. Comput. Sci.*, vol. 5, no. 5, pp. 16467–16470, 2016, doi: 10.18535/ijecs/v5i5.23.

[17] V. Koolwal and K. K. Mohbey, "A comprehensive survey on trajectory-based location prediction," *Iran J. Comput. Sci.*, vol. 3, no. 2, pp. 65–91, 2020, doi: 10.1007/s42044-019-00052-z.

[18] J. P. Epperlein, J. Monteil, M. Liu, Y. Gu, S. Zhuk, and R. Shorten, "Bayesian classifier for Route prediction with Markov chains," *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC*, vol. 2018-November, pp. 677–682, 2018, doi: 10.1109/ITSC.2018.8569895.

[19] R. Simmons, B. Browning, Y. Zhang, and V. Sadekar, "Learning to predict driver route and destination intent,"

IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC, pp. 127–132, 2006, doi: 10.1109/itsc.2006.1706730.

[20] S. Gambs, M. O. Killijian, and M. N. Del Prado Cortez, "Next place prediction using mobility Markov chains," *Proc. 1st Work. Meas. Privacy, Mobility, MPM'12*, pp. 0–5, 2012, doi: 10.1145/2181196.2181199.

[21] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami, and J. C. Bezdek, "A Scalable Framework for Trajectory Prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3860–3874, 2019, doi: 10.1109/TITS.2019.2899179.

[22] V. S. Tiwari, A. Arya, and S. Chaturvedi, "Scalable prediction by partial match (PPM) and its application to route prediction," *Appl. Informatics*, vol. 5, no. 1, 2018, doi: 10.1186/s40535-018-0051-z.

[23] S. Qiao, N. Han, J. Wang, R. H. Li, L. A. Gutierrez, and X. Wu, "Predicting Long-Term Trajectories of Connected Vehicles via the Prefix-Projection Technique," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2305–2315, 2018, doi: 10.1109/TITS.2017.2750075.

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

[25] T. Miklušćák, M. Gregor, and A. Janota, "Using neural networks for route and destination prediction in intelligent transport systems," *Commun. Comput. Inf. Sci.*, vol. 329 CCIS, pp. 380–387, 2012, doi: 10.1007/978-3-642-34050-5_43.

[26] T. Cao, E. P. Lim, Z. H. Zhou, T. B. Ho, D. Cheung, and H. Motoda, "Advances in knowledge discovery and data mining: 19th Pacific-Asia Conference, PAKDD 2015 Ho Chi Minh City, Vietnam, May 19–22, 2015 proceedings, Part II," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9078, no. May, pp. 409–421, 2015, doi: 10.1007/978-3-319-18032-8.

[27] F. Terroso-Saenz, M. Valdes-Vela, and A. F. Skarmeta-Gomez, "Online route prediction based on clustering of meaningful velocity-change areas," *Data Min. Knowl. Discov.*, vol. 30, no. 6, pp. 1480–1519, 2016, doi: 10.1007/s10618-016-0452-3.

[28] H. Amirat, N. Lagraa, P. Fournier-Viger, and Y. Ouinten, "NextRoute: a lossless model for accurate mobility prediction," *J. Ambient Intell. Humaniz. Comput.*, no. 0123456789, 2019, doi: 10.1007/s12652-019-01327-w.

[29] S. M. Zabihi, S. S. Beauchemin, and M. A. Bauer, "Real-Time driving manoeuvre prediction using IO-HMM and driver cephalo-ocular behaviour," *IEEE Intell. Veh. Symp. Proc.*, no. Iv, pp. 875–880, 2017, doi: 10.1109/IVS.2017.7995826.

[30] F. Gross, J. Jordan, F. Weninger, F. Klanner, and B. Schuller, "Route and stopping intent prediction at intersections from car fleet data," *IEEE Trans. Intell. Veh.*, vol. 1, no. 2, pp. 177–186, 2016, doi: 10.1109/TIV.2016.2617625.

[31] Y. Dai *et al.*, "Dynamic prediction of drivers' personal routes through machine learning," *2016 IEEE Symp. Ser. Comput. Intell. SSCI 2016*, pp. 1–8, 2017, doi: 10.1109/SSCI.2016.7850094.

[32] M. Wei and Y. Meng, "Research on the optimal route choice based on improved Dijkstra," *Proc. - 2014 IEEE Work. Adv. Res. Technol. Ind. Appl. WARTIA 2014*, pp. 303–306, 2014, doi: 10.1109/WARTIA.2014.6976257.

[33] J. Park, Y. L. Murphey, R. McGee, J. G. Kristinsson, M. L. Kuang, and A. M. Phillips, "Intelligent trip modeling for the prediction of an origin-destination traveling speed profile," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 3, pp. 1039–1053, 2014, doi: 10.1109/TITS.2013.2294934.

[34] V. S. Tiwari, S. Chaturvedi, and A. Arya, "Route prediction using trip observations and map matching," *Proc. 2013 3rd IEEE Int. Adv. Comput. Conf. IACC 2013*, pp. 583–587, 2013, doi: 10.1109/IAdCC.2013.6514292.

[35] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet, "Online map-matching based on hidden Markov model for real-time traffic sensing applications," *IEEE Conf. Intell. Transp. Syst. Proceedings, ITSC*, vol. 117543, pp. 776–781, 2012, doi: 10.1109/ITSC.2012.6338627.

[36] G. Xue, Z. Li, H. Zhu, and Y. Liu, "Traffic-known urban vehicular route prediction based on partial mobility

patterns," *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, pp. 369–375, 2009, doi: 10.1109/ICPADS.2009.129.

[37] Froehlich J, Krumm J (2008) Route prediction from trip observations. In: Proceedings of SAE world congress, vol 2193, p 53.

[38] Torkkola K, Zhang K, Li H et al (2007) Trafc advisories based on route prediction. In: Proceedings of workshop on mobile interaction with the real world, MIRW 2007, pp 33–36

[39] https://snap.stanford.edu/data/loc-Brightkite.html (visited 2020/08/23)

[40] https://snap.stanford.edu/data/loc-gowalla.html (visited 2020/08/23)

[41] Fournier-Viger P, Gomariz A, Gueniche T et al (2014) SPMF : a Java open-source pattern mining library. J Mach Learn Res 15:3569–3573

[42] F. Marchal, J. Hackney, and K. Axhausen, "Efficient map matching of large global positioning system data sets: Tests on speed-monitoring experiment in zürich," Transp. Res. Rec. J. Transp. Res. Board, vol. 1935, pp. 93–100, Jan. 2005.