UNIVERSITAT ROVIRA I VIRGILI

CONTRIBUTIONS TO TRAJECTORY ANALYSIS AND PREDICTION: STATISTICAL AND DEEP LEARNING TECHNIQUES

Abdulrahman Qasem Al-Molegi

UNIVERSITAT
ROVIRA i VIRGILI

Contributions to Trajectory Analysis and Prediction

Statistical and Deep Learning Techniques

ABDULRAHMAN QASEM YAHYA AL-MOLEGI

DOCTORAL THESIS
2019

Contributions to Trajectory Analysis and Prediction

Statistical and Deep Learning Techniques

DOCTORAL THESIS

Author:
Abdulrahman Qasem Yahya Al-Molegi

Advisors:
Dr. Antoni Martínez-Ballesté
Dr. Agusti Solanas Gómez

Departament d'Enginyeria Informàtica i Matemàtiques

Universitat Rovira i Virgili
Tarragona
2019

UNIVERSITAT
ROVIRA I VIRGILI

Departament d'Enginyeria Informàtica
i Matemàtiques

Av. Paisos Catalans, 27
43007 Tarragona
Tel. +34 977 55 95 95
Fax. +34 977 55 95 97

We STATE that the present study, entitled "Contributions to Trajectory Analysis and Prediction: Statistical and Deep Learning Techniques", presented by Abdulrahman Qasem Yahya Al-Molegi, for the award of the degree of Doctor, has been carried out under our supervision at the Departament d'Enginyeria Informàtica i Matemàtiques.

Tarragona, April 2019.

Doctoral Thesis Supervisors,

Dr. Antoni Martínez-Ballesté

Dr. Agusti Solanas Gómez

To my wife Samar, my sons Wesam, Wasem and Ellen, my parents, brothers and sister

Abstract

Due to the relationship between people's daily life and specific geographic locations, the historical trajectory data of a person contains lots of valuable information that can be used to discover their lifestyle and regularity. The generalisation in the use of mobile devices with location capabilities has fueled trajectory mining: the research area that focuses on manipulating, processing and analysing trajectory data to aid the extraction of higher level knowledge from the trajectory history of a user. Based on this analysis, even the person's next probable location can be predicted. These techniques pave the way for the improvement of current location-based services and the rise of new business models, based on rich notifications related to the right prediction of users' next location. This thesis addresses location prediction as well as the discovery of significant regions in person's movement area. It proposes various models to predict the future state of people movement, based on different machine learning techniques (such as Markov Chains, Recurrent Neural Networks and Convolutional Neural Networks) and considering different input representation methods (embedding learning and one-hot vector). Moreover, the attention technique is used in the prediction model, aiming at aligning time intervals in people's trajectories that are relevant to a specific location. Furthermore, the thesis proposes a time encoding scheme to capture movement behavior characteristics. In addition to that, it analyses the impact of Space-Time representation learning through evaluating different architectural configurations. Finally, trajectory analysis and location prediction is applied to real-time smartphone-based monitoring system for seniors.

Keywords: Location Prediction, Deep Learning, Time Encoding Scheme, Trajectory Analysis, Attention technique, Embedding representations learning, Regions-of-interest discovering, Wandering, Smart Health, Mild Cognitive Impairments, Dementia, Monitoring system.

Resum

A causa de l'estreta relació entre la vida de les persones i determinades ubicacions geogràfiques, les dades històriques sobre trajectòries d'una persona contenen informació valuosa que es pot utilitzar per descobrir els seus estils de vida i hàbits. L'ús generalitzat de dispositius mòbils amb capacitat de localització ha impulsat la mineria de trajectòries (trajectory mining), la qual se centra en la manipulació, el processament i l'anàlisi de dades de trajectòries per facilitar l'extracció de coneixement a partir de l'històric de les trajectòries d'una persona. Basant-nos en aquesta anàlisi, fins i tot es pot arribar a predir quina serà la probable propera localització d'una persona. Amb aquestes tècniques, s'obre la porta a la millora dels actuals serveis basats en la ubicació i a l'aparició de nous models de negoci, basats en notificacions riques relacionades amb la predicció adequada de les futures ubicacions dels usuaris. Aquesta tesi tracta sobre la predicció de la ubicació i el descobriment de regions significatives a les zones de moviment de les persones. Proposa diversos models de predicció, basant-se en diferents tècniques d'aprenentatge automàtic (com ara les cadenes de Markov, les xarxes neuronals recurrents i les xarxes neuronals convolucionals), tot considerant diferents mètodes de representació d'entrada (embedding learning i one hot vector). A més, el model de predicció utilitza la attention technique (tècnica d'atenció), que té com a objectiu alinear els intervals de temps en les trajectòries de les persones que són rellevants per a una ubicació específica. La tesi també proposa un esquema de codificació temporal per capturar les característiques del comportament del moviment. Addicionalment, analitza l'impacte de l'aprenentatge de la representació espacial-temporal mitjançant l'avaluació de diferents arquitectures. Finalment, l'anàlisi de la trajectòria i la predicció de localització s'apliquen a la monitorització en temps real per a persones grans.

Resumen

Debido a la estrecha relación entre la vida de las personas y determinadas ubicaciones geográficas, los datos históricos sobre trayectorias de una persona contienen información valiosa que se puede utilizar para descubrir sus estilos de vida y hábitos. El uso generalizado de dispositivos móviles con capacidad de localización ha impulsado la minería de trayectorias (trajectory mining), la cual se centra en la manipulación, el procesamiento y el análisis de datos de trayectorias para facilitar la extracción de conocimiento a partir de el histórico de las trayectorias de una persona. Basándonos en este análisis, incluso se puede llegar a predecir cuál será la probable próxima localización de una persona. Con estas técnicas, se abre la puerta a la mejora de los actuales servicios basados en la ubicación y en la aparición de nuevos modelos de negocio, basados en notificaciones ricas relacionadas con la predicción adecuada de las futuras ubicaciones de los usuarios. Esta tesis trata sobre la predicción de la ubicación y el descubrimiento de regiones significativas en las zonas de movimiento de las personas. Propone varios modelos de predicción, basándose en diferentes técnicas de aprendizaje automático (como las cadenas de Markov, las redes neuronales recurrentes y las redes neuronales convolucionales), considerando diferentes métodos de representación de entrada (embedding learning y one hot vector). Además, el modelo de predicción utiliza la attention technique (técnica de atención), que tiene como objetivo alinear los intervalos de tiempo en las trayectorias de las personas que son relevantes para una ubicación específica. La tesis también propone un esquema de codificación temporal para capturar las características del comportamiento del movimiento. Adicionalmente, analiza el impacto del aprendizaje de la representación espacial-temporal mediante la evaluación de diferentes arquitecturas. Finalmente, el análisis de la trayectoria y la predicción de localización se aplican a la monitorización en tiempo real para personas mayores.

Acknowledgements

The author was supported by a PhD grant from URV in 2015 (Martí Franquès program).

I would like to express my gratitude to my supervisors Dr. Antoni Martínez-Ballesté and Dr. Agusti Solanas for their useful guidance, insightful comments, and considerable encouragement to complete this thesis. Special thanks to Mohamed Jebreel for his cooperation and support.

Contents

List of Figures

List of Tables

List of Symbols

The next list describes several symbols that will be later used within the thesis.

Symbol	Description
$R@N$	Recall@N
$P@N$	Precision
$\mathcal{U}$	Set of users
u	User
L_u	Real visited locations by a user u
$PL_{N,u}$	Top N predicted locations
$\mathcal{M}_i$	Movement
$\mathcal{R}$	Set of RoIs
r_i	RoI vector
$\hat{r}_{i+1}$	Predicted RoI
N	Number of RoIs
Re	RoI embedded matrix
re_i	RoI embedded vector
d_r	RoI embedded vector dimensionality
W_r	RoI weight matrix
$\mathcal{T}$	Set of time intervals
t_i	Time vector
M	Number of different time intervals
Te	Time intervals embedded matrix
te_i	Time embedded vector
d_t	Time embedded vector dimensionality
W_t	Time weight matrix
w	Number of visited RoIs taken as inputs to the model
h_i	Hidden layer vector
d_h	Hidden embedded vector dimensionality
$\hat{y}_i$	Prediction layer vector
$W_{h_{i-1}}$	Previous hidden weight matrix
b_h	Hidden layer bias
W_h	Hidden weight matrix
b_o	Output layer bias
y	Real next RoI

$\hat{y}$	Predicted next RoI probability
ε	Maximum distance between
$minPts$	Minimum number of points
β	Fixed vector
W_β	Weight matrix
p	Number of pooling function
e_i	Entering time vector
l_i	Leaving time vector
Ee	Entering time intervals embedded matrix
Le	Leaving time intervals embedded matrix
ee_i	Entering time embedded vector
le_i	Leaving time embedded vector
d_e	Entering time embedded vector dimensionality
d_l	leaving time embedded vector dimensionality
W_e	Entering time weight matrix
W_l	Leaving time weight matrix
J	Number of different weekday types
q_i	Weekday vector
qe_i	Weekday embedded vector
d_q	Weekday embedded vector dimensionality
Qe	Weekday embedded matrix
W_q	Week weight matrix
x	Time encoding vector
xe_i	Time encoding embedded vector
Xe	Time encoding embedded matrix
W_x	Time encoding weight matrix
$\bar{\eta}$	Attention vector
Ω	Context vectors
α	Alignment weight vector
C	Content-based function
W_C	Content-based weight matrix
F	Set of convolution filters
k	Convolution filter height
R_{conv_i}	RoI convolution vector
T_{conv}	Time convolution vector
RT_{concat}	RoI and Time concatenated vector

δ	Distance
τ	Stay time
rd	Region density
Nxt_r	Next RoI
Cur_r	Current RoI
Cur_t	Current time
Prv_r	Previous RoIs
Prv_t	Previous times
$\mathcal{S}$	Sequence of directions values
s_i	Direction vector
Se	Direction embedded matrix
se_i	Direction embedded vector
d	Direction embedded vector dimensionality
W_s	Direction weight matrix

Part I

Introduction

CHAPTER 1

Introduction

Human behaviour is very complex and diverse. It can consist of a large number of attributes in which only some of those can be predicted from historical trends. Breaking human behaviour into different elements such as mobility patterns, shopping habits, etc., and investigating each element separately aims to reduce the complexity of the problem into a manageable subset. Mobility, as a component of human behaviour, is also complex, but its variability is lower and could be studied with more focused pattern-recognition approaches. In most cases, human mobility is analysed with the goal of predicting future behaviours. Mobility prediction is defined as the prediction of people's next location in the region that they usually move in.

With the proliferation and widespread of mobile devices, such as tablet PCs, smartphones, smart watches, etc., the availability of these devices to the general public, together with the rapid enhancement of their data collection technologies either indoor (Bluetooth, Wi-Fi) or outdoor (Global Positioning System (GPS), Global System for Mobile communications (GSM)) and finally mobile access to the Internet, a massive amount of mobility data from moving objects (*e.g.,* people and vehicles) can be obtained at a very low cost. Furthermore, the availability of large amounts of collected mobility data along with the development of location-based applications and services in mobile devices, have received considerable interests from both the industry and the research community towards building efficient methods for analysing and extracting knowledge from this data to predict user's next movement or location (Rashidi and Cook, 2010; Löwe et al., 2012; Lin and Hsu, 2014).

Many companies track users' daily visited locations and record these large amount of data for many different applications. Portable devices with localization systems such as GPS are used to collect mobility data. Users carry smartphones and wearables most of the time. Literature shows that companies can track such data from those devices even when location based applications are not enabled. Additionally, users' locations might be tracked even when smartphone Wi-Fi and/or data connections are disabled.

Location-based services (LBSs) can be considered as one of the most stimulating fields that impact both e-commerce and classical businesses. Users can be offered services relevant to where they are, right now. LBSs focus on enhancing smartphones' applications with the ability to know the current and historical locations of phones' users. LBSs are developed to be part of a smart city architecture to improve people's quality of life and also to empower marketing, becoming much more focused and relevant.

The main requirement to provide LBSs is to continuously be able to determine where the users are during their daily activities. This can be divided into three parts, where each part can be relevant to specific marketing strategies:

- **Historical locations.** Some applications like to know where user has been in the past, how often, for how long, etc.
- **Current Location.** Some applications like to know where the user is right now.
- **Future locations.** This is the focus of our research. Future locations can be

also divided into: next or the very near future possible location and the future location in general (*e.g.*, in the next summer vacation, holidays, etc.)

LBSs can be improved by predicting the most probable next location the user will visit. This prediction is used to make proactive offerings or services based on those predicted locations.

1.1 Motivation

How much can one know about individuals behaviour just by looking at the way they move from one place to another? Understanding human mobility at the level of daily lives is crucial to a broad-domain ubiquitous computing applications such as healthcare applications (Solanas et al., 2014a), network management (Vranova and Ondryhal, 2011), personal positioning (Fang et al., 2009), human computer interaction, public safety assurance, socio-economic modeling for urban planning, public transportation planning (Gonzalez et al., 2008; Gudmundsson et al., 2012; Zheng et al., 2014; Xia et al., 2014a), location-based services and advertisements (Rao and Minakakis, 2003; Kim et al., 2015), ubiquitous advertising (Krumm, 2011), crime prediction (Gerber, 2014), location recommendation systems (Quercia et al., 2010; Casino et al., 2015; Yu and Chen, 2015), epidemic prevention (Kleinberg, 2007), route planning, carpooling, meeting planners, and many others (Asgari et al., 2013).

Next location prediction can be useful in developing the vehicle system intelligence. For example, next location prediction can enhance the navigation system performance by providing information related to predicting the habitual places and routes of the driver. Such valuable information can substantially contribute to safer and more efficient driving (Wevers et al., 2010), reducing of fuel consumption and consequently, exhaust emissions (Ericsson et al., 2006; Lee et al., 2008; Ganti et al., 2010). In addition to that, combining the predicted routes the driver is going to take with real-time roads information obtained from several sources by vehicle's onboard systems, can be used to warn the driver about traffic conditions and unsafe sections of a route (Wevers et al., 2010).

In the marketing environment, next location prediction can play an important role in providing mobile users with appropriate services or advertisements related to the location is going to be visited (Barwise and Strong, 2002; Barnes and Scornavacca, 2004). For example, if the model knows that the user will go for lunch, then it can recommend a list of restaurants together with useful information like today's menu, availabilities, etc. Providing such services ensures that users will not be disturbed at inappropriate times and thereby not losing their interest in the services. In addition to that, this could help the advertisers to reduce cost and maximum possible gain from the service.

Location prediction can play an important role in crime suppression and rehabilitation where electronic monitoring and parole is needed. Instead of imprisoning the persons specially who have committed fairly minor crimes, monitoring them electronically helps reduce costs and ease prison overcrowding in addition to reduce the risk of corrupting them when imprisoning them with harder criminals (Perusco and Michael, 2007). Further, a next location model can predict the location and time where a certain victim and offender may potentially meet in order to avoid unwanted encounters. The local authorities can increase police presence at the predicted locations and thus avoid the occurrence of criminal acts.

Disaster relief can also be supported by next location prediction (Gao et al., 2011a,b; Zook et al., 2010). Currently, incidents and requests are reported during a disaster by volunteers and victims who have a phone or other communication device. There are no applications for forecasting future requests. However, in case of limited communication abilities for example in black-out areas, responders, governmental and non-governmental organisations face difficulties in deploying aid and rescue capabilities (Gao et al., 2011a). Scientific data about floods, earthquakes, and other phenomena together with users data from volunteers, victims and non-governmental organisations employees can be used to build models to predict the location of future requests and needs. While data collection from disaster scenes is a challenging task, timely and accurate data enables government and non-governmental organisations to respond appropriately.

Location prediction can be used in wireless mobile ad-hoc networks (H. Kaaniche, 2010; Dekar and Kheddouci, 2008). The mobile node in the ad-hoc networks performs as a host as well as a router and communicates either directly or via other nodes of the network by establishing routes. However, these routes are prone to frequent break up due to the mobility of the nodes. If the future location of the mobile can be predicted, the resources reservation can be made before be asked, which enables the network to provide a better quality of service. Furthermore, location prediction can be used to estimate the expiration time of the links connecting the nodes enabling them to select the most stable paths which improves routing performances.

Location prediction is also crucial in cellular communication network to increase the efficiency of the network (Yavaş et al., 2005). Using the location prediction information, the system can effectively allocate resources to the cells that are most likely a mobile user will move to rather than blindly allocating excessive resources in all neighbour cells to the current cell (Yavaş et al., 2005; Gohil, 2014; Kumar et al., 2015). Effective allocation of resources to mobile users would reduce the latency in accessing the resources and improve resource utilization. In addition to that, this prediction can be used to automatically update the location information of the mobile user which reduces the traffic load as a result of location update and paging in cellular communication networks (Parija et al., 2013c,a,b; Leca et al., 2015; L. Vintan and Ungerer, 2004; Daoui et al., 2012; Kumar and Venkataram, 2002).

Location prediction can potentially benefit many other areas, such as saving energy in residential buildings (Mozer, 1998, 2004; Gupta et al., 2009). In the USA, the residential sector uses 21% of the total U.S. energy consumption, while heating and cooling are responsible for 46% of the total energy consumed in U.S. residential buildings (Gupta et al., 2009). The residential energy generally can be saved using prediction model that learn the behaviour and needs patterns of the inhabitants by monitoring them over a period of time. Next location prediction model can provide a better estimation of the time when an inhabitants return home. Different home comfort systems such as thermostats, lighting can be then simply turned on.

In the field of public transportation network, location prediction can be used to provide beneficial strategies for passengers and taxi drivers (Yuan et al., 2011; Guang et al., 2015). The vacant taxis cruising on roads generate additional traffic, increase the exhaust pollution and waste gas and time of taxi drivers. However, to improve the utilization of the vacant taxis and reduce the energy consumption effectively, while enabling people to take a taxi more easily, a recommender system based on location prediction can be proposed for both passengers and taxi drivers. The system will suggest the taxi driver with a location which he/she is most likely to pick up a passenger. The prediction information helps the taxi driver to reduce the time of cruising without a fare, and thus, saves energy consumption and eases the exhaust pollution as well as maximizes the profit. In addition to that, people are most likely to find a taxi within a walking distance.

Users' location histories can be used to learn extensive knowledge about their behaviour and preferences. Friend recommendation systems can be developed by combining multiple users' social network and their location histories (Ye et al., 2009; Yang et al., 2017). It allows for detecting the locations for the users sharing the same interests and activities.

Next location prediction can predict the time when a person is going to be present at particular locations (Krumm and Brush, 2011). In other words, it can be used to detect anomalous behaviour such as when a disabled patient or child is expected to be at a certain location but is not. Further action can be initiated such as an emergency call. In addition to that, presence prediction could be useful in other scenarios such as an intelligent postal service.

Due to society ageing, the number of age-related diseases, such as dementia and Mild Cognitive Impairments (MCI), are gaining increasing importance. People suffering from this kind of diseases notice a slight deterioration in their cognitive abilities and they often have difficulties in navigational tasks. Next location prediction can support real-time assistive services for those people in order to provide active cognitive aids and improve their quality of life. Different assisted Cognition systems are developed (Shimizu et al., 2000; Patterson et al., 2002, 2004; Hossain et al., 2011) that helps to reduce spatial disorientation indoor and outdoor.

1.2 Contributions

The main contributions of this thesis are the following:

1. We propose a model to predict the future state of people movement. We use the embedding learning technique to effectively discover adequate internal representations of data input features enabling the model to capture the embedded semantic information about the users's behaviour more effectively. Meanwhile, Recurrent Neural Networks (RNN) is used in order to keep track of user movement history which allows to discover more meaningful dependencies. Furthermore, we propose another location prediction model in which time encoding scheme is proposed to capture movement behaviour characteristics. We also explore a set of neural pooling functions in order to extract rich features. Moreover, we study the impact of using different Space-Time input data in location prediction model with different architectural configurations.

   The results of the previous studies have been published in the following conferences:

   - **Abdulrahman Al-Molegi**, Mohammed Jabreel, and Baraq Ghaleb, "STF-RNN: Space-Time Features-based Recurrent Neural Network for predicting people next location", in IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 1-7, 2016.
   - **Abdulrahman Al-Molegi**, Antoni Martínez-Ballesté, and Mohammed Jabreel, "Geo-Temporal Recurrent Model for Location Prediction", in 20th International Conference of the Catalan Association for Artificial Intelligence, 126-135, 2017.
   - **Abdulrahman Al-Molegi**, Antoni Martínez-Ballesté, "The Effect of Space-Time Representation Learning in Predicting People's Next Location", in 21st International Conference of the Catalan Association for Artificial Intelligence, 64-73, 2018.

2. We propose an attention-based neural network model, called Move, Attend and Predict (MAP), for the problem of predicting people's next location. We show that the proposed model is essentially able to learn which time interval in the

trajectory sequences are relevant regarding a specific location.

The result of the previous study has been published in the following journal:

- **Abdulrahman Al-Molegi**, Mohammed Jabreel and Antoni Martínez-Ballesté, "Move, Attend and Predict: An Attention-based Neural Model for People's Movement Prediction", Pattern Recognition Letters, 112, 34-40, 2018, ISSN: 0167-8655 (1.952, Q2).

3. We propose a new approach to discover and predict people's next location based on their mobility patterns, while being computationally efficient. The first step in the approach proposed is to discover the Regions-of-Interest (RoIs) in people's historical trajectories. Then, we propose a model for predicting a user's next location based on Markov Chain (MC) to overcome the drawback of classical MC. Moreover, we use a general transformation function to transform the n-order MC into first order which helps to make more abstraction on n-order.

   The result of the previous study has been published in the following journal:

   - **Abdulrahman Al-Molegi**, Izzat Alsmadi and Antoni Martínez-Ballesté, "Regions-of-interest discovering and predicting in smartphone environments", Pervasive and Mobile Computing, 47, 31-53, 2018, ISSN 1574-1192, (ISI JCR 2016 2.349, Q2).

4. We address the assessment of wandering detection methods from different perspectives. First, we review the available datasets in the literature that can be used for benchmarking wandering detection methods. Moreover, we analyse the available datasets using a well-known wandering detection technique, so as to obtain the number of abnormal trajectories with respect to the whole dataset and each individual. Finally, we discuss the properties that a dataset for benchmarking should fulfill.

   The result of the previous study has been published in the following conference:

   - Antoni Martínez-Ballesté, **Abdulrahman Al-Molegi**, and Edgar Batista, "On the Detection of Wandering using Trajectories Datasets", in the 9th International Conference on Information, Intelligence, Systems and Applications IISA, IEEE (2018).

5. We propose a real-time smartphone-based monitoring system, called SafeMove, to discover and predict elderly people behaviours by analysing outdoor trajectories. We use Convolutional Neural Network (CNN) in order to keep track of elder movement history and then, predict the most popular locations he/she might visit in the next time. Moreover, we develop a model called Abnormal Behaviour Detection (ABD) using RNN that is able to detect the different abnormal behaviours scenario in real-time.

   The result of the previous study has been submitted to the following journal:

   - **Abdulrahman Al-Molegi**, Antoni Martínez-Ballesté, "*SafeMove*: Monitoring Seniors with Mild Cognitive Impairments using Deep Learning and Location Prediction", Expert System with Application.

1.3 Thesis Organisation

The thesis is divided into the following four parts:

- Part I: Introduction
  - Chapter 1: *Introduction*

    This chapter introduces in general trajectory data analysis. It then describes the motivation behind the thesis and its contributions to improve location prediction models.
  - Chapter 2: *Background*

    In this chapter, we describe the background to trajectory mining, prediction models evaluation and mobility datasets.
- Part II: Contributions to deep learning models for location prediction
  - Chapter 3: *Preliminary Matters*

    In this chapter, we have described the main location prediction steps. In the first step, data preprocessing is used to remove possible noise. In the second step, RoIs located in a user movement region are discovered. The third step is the process of building a prediction model. In the last step, the prediction model is evaluated using a testing dataset.

- Chapter 4: *Recurrent Neural Network for Predicting People's Next Location*

  In this chapter, STF-RNN model is proposed to predict the future state of people movement. The embedding learning technique is used to effectively discover adequate internal representations of space and time input features. The recurrent structure is incorporated with space and time interval sequences in order to discover long-term dependencies.

- Chapter 5: *The Effect of Different Architectural Configurations in Location Prediction Model*

  In this chapter, we study the performance of location prediction model through evaluating different architectural configurations. First, we study the impact of using a time encoding scheme that provides the model with more information related to the movement time. To extract the features of the context data, we use embedding representation technique. To obtain rich features, a set of neural pooling functions are explored. Second, we evaluate the impact of many different data inputs on the model final prediction performance. Third, we investigate the impact of input representation techniques on the prediction performance using both embedding representation learning and one-hot vector representation (*i.e.* static vectors).

- Chapter 6: *An Attention-based Neural Model for People's Movement Prediction*

  In this chapter, we propose MAP model in which an attention technique is used to provide the learning model with more information, related to the movement time. The proposed model essentially learns which time interval in the trajectory sequences are relevant regarding a specific location.

- Chapter 7: *Convolutional Neural Network for Predicting People's Next Location*

In this chapter, we propose a prediction model, called ST-CNN . We use CNN architecture in order to keep track of a user movement history and then, predict the most popular locations his/her might visit in the next time.

- Part III: Contributions to specific applications
  - Chapter 8: *RoI Discovering and Predicting in Smartphone Environments*

    This chapter presents an approach to discover RoIs in the users movement area and then predict their future locations, which play a key role in the success of advanced location-based services. Based on MC, we propose a model for predicting a user's next location. Moreover, we use a general transformation function for the trajectory to include the space and time context, which helps make more abstraction on n-order.
  - Chapter 9: *Monitoring Seniors with Mild Cognitive Impairments using Deep Learning and Location Prediction*

    In this chapter, we propose a real-time smartphone-based monitoring system based on the analysis of elderly people's trajectories to help them move independently and safely. Firstly, we analyse the elder's mobility data previously collected using CNN in order to keep track of his/her movement history and then, predict the most popular locations the elder might visit in the next time from his/her current location. We then develop a model based on RNN which is able to detect the different abnormal behaviours scenario in real-time.
- Part VI: *Conclusion*
  - Chapter 10: Concluding remarks.

    This chapter presents the conclusions of the thesis and some lines of future research.

CHAPTER 2

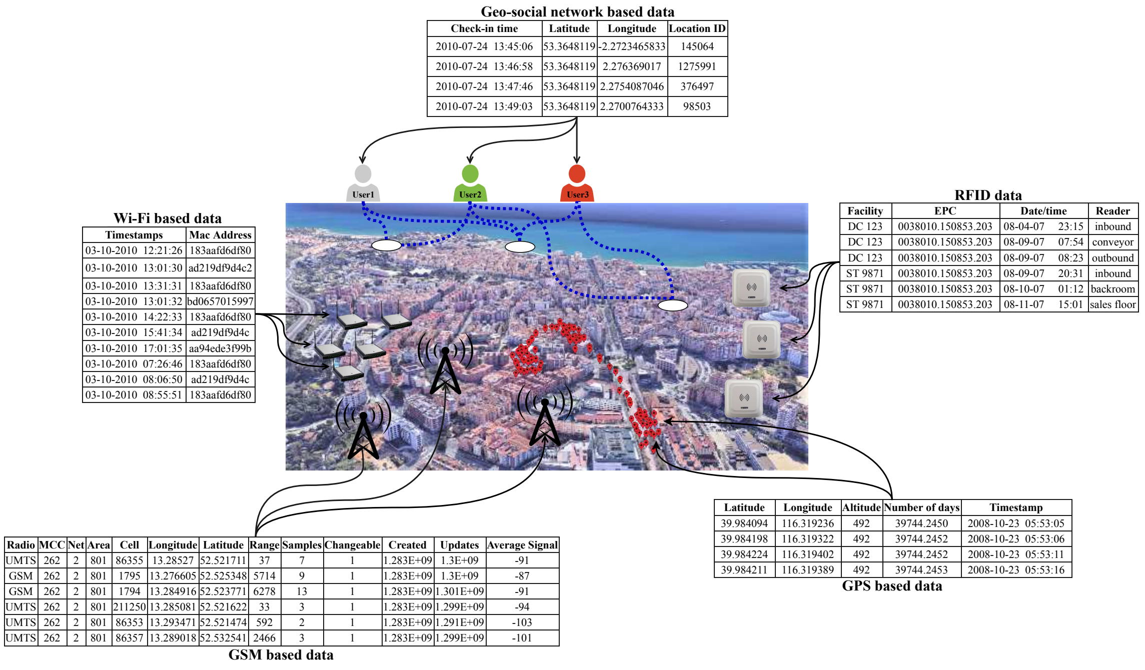
Background

2.1 Introduction

The advances in mobile computation and the self-location capability of devices such as smart phones and wearables boost the generation of various large-scale trajectory data which track the traces of moving objects. A trajectory of a moving object is typically represented by a sequence of timestamped locations.

In the past decade, a variety of trajectory data mining tasks have been proposed, such as RoI discovery, trajectory pattern mining, location prediction, outlier detection, movement behaviour analysis and trajectory classification to name a few. As a result, there exist a broad range of applications that can benefit from trajectory data mining. Trajectory data mining is application driven that depends on what we want to reveal from that data.

In this chapter, we present the background concepts that form the basis of the



Geo-social network based data

Check-in time	Latitude	Longitude	Location ID
2010-07-24 13:45:06	53.3648119	-2.2723465833	145064
2010-07-24 13:46:58	53.3648119	2.276369017	1275991
2010-07-24 13:47:46	53.3648119	2.2754087046	376497
2010-07-24 13:49:03	53.3648119	2.2700764333	98503

Wi-Fi based data

Timestamps	Mac Address
03-10-2010 12:21:26	183aafd6df80
03-10-2010 13:01:30	ad219df9d4c2
03-10-2010 13:31:31	183aafd6df80
03-10-2010 13:01:32	bd0657015997
03-10-2010 14:22:33	183aafd6df80
03-10-2010 15:41:34	ad219df9d4c
03-10-2010 17:01:35	aa94ede3f99b
03-10-2010 07:26:46	183aafd6df80
03-10-2010 08:06:50	ad219df9d4c
03-10-2010 08:55:51	183aafd6df80

RFID data

Facility	EPC	Date/time	Reader
DC 123	0038010.150853.203	08-04-07 23:15	inbound
DC 123	0038010.150853.203	08-09-07 07:54	conveyor
DC 123	0038010.150853.203	08-09-07 08:23	outbound
ST 9871	0038010.150853.203	08-09-07 20:31	inbound
ST 9871	0038010.150853.203	08-10-07 01:12	backroom
ST 9871	0038010.150853.203	08-11-07 15:01	sales floor

Latitude	Longitude	Altitude	Number of days	Timestamp
39.984094	116.319236	492	39744.2450	2008-10-23 05:53:05
39.984198	116.319322	492	39744.2452	2008-10-23 05:53:06
39.984224	116.319402	492	39744.2452	2008-10-23 05:53:11
39.984211	116.319389	492	39744.2453	2008-10-23 05:53:16

GPS based data

Radio	MCC	Net	Area	Cell	Longitude	Latitude	Range	Samples	Changeable	Created	Updates	Average Signal
UMTS	262	2	801	86355	13.28527	52.521711	37	7	1	1.283E+09	1.3E+09	-91
GSM	262	2	801	1795	13.276605	52.525348	5714	9	1	1.283E+09	1.3E+09	-87
GSM	262	2	801	1794	13.284916	52.523771	6278	13	1	1.283E+09	1.301E+09	-91
UMTS	262	2	801	211250	13.285081	52.521622	33	3	1	1.283E+09	1.299E+09	-94
UMTS	262	2	801	86353	13.293471	52.521474	592	2	1	1.283E+09	1.291E+09	-103
UMTS	262	2	801	86357	13.289018	52.532541	2466	3	1	1.283E+09	1.299E+09	-101

GSM based data

Figure 2.1: Trajectory data forms.

trajectory analysis and prediction. We begin with defining the concept of trajectory data and trajectory mining. Then, we study three different mining tasks that can be discovered from a trajectory data. The important basic concepts and ideas of deep learning are introduced in the next section. Finally, the evaluation and datasets used in our proposals are described.

2.2 Trajectory Data Mining

There are various trajectory data forms which depends on the technology used to collect, see Figure 2.1. Spinsanti et al. (2013) differentiated GSM, GPS and geo-social network based trajectory data. Pelekis and Theodoridis (2014) added two other forms: Wi-Fi based and RFID (radio frequency identification) based data. The collected trajectory data is categorized into two types based on their representation:

- Continuous, such as GPS coordinates.
- Discrete, such as cellular station IDs and Wi-Fi access points MAC addresses.

Generally, continuous data are converted into discrete positions for the sake of manageability. GPS based data, also called spatial-temporal trajectory, is a trace

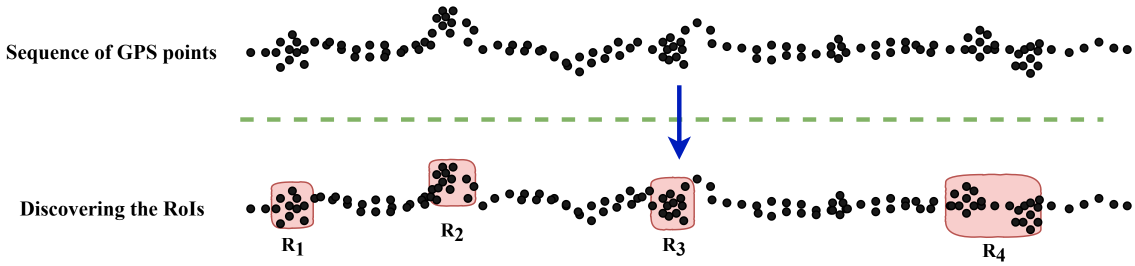


Figure 2.2: Discovery of significant places.

generated by a GPS-enabled device carried by moving object in geographical spaces. It is composed of chronologically ordered sequences of geographic coordinates.

Due to the tight relationship between people's daily life and geographic locations, the historical trajectory data contains lots of valuable information that can be used to discover their lifestyle and regularity (Ye et al., 2009). Trajectory mining emerges as the research area that focuses on manipulating, processing and analysing trajectory data to aid the extraction of higher level knowledge from the trajectory history of a user.

2.2.1 Discovery of Significant Places

In the last few years, there has been an explosion of mobility data collected by various GPS enabled devices, emerging the necessity of efficient techniques to analyse these data in different application domains. However, useful information may only be extracted from mobility data when the significant and important places are discovered. The goal of discovery of significant places is to detect a user's specific important locations in his/her movement area. In this thesis, a significant place will be referred to RoI, (*i.e.* a region where the user usually goes and waits, slows or stays for a while in order to complete important activities) such as home, restaurants, places of work, train stations, shops, etc.

RoI Discovery is the process that takes a sequence of GPS points from user's mobility data in order to produce a set of RoIs. Figure 2.2 shows an example of a trajectory where the discovering method must specify. It can be observed that, in general, the density of GPS points in any RoI is higher than in other regions because

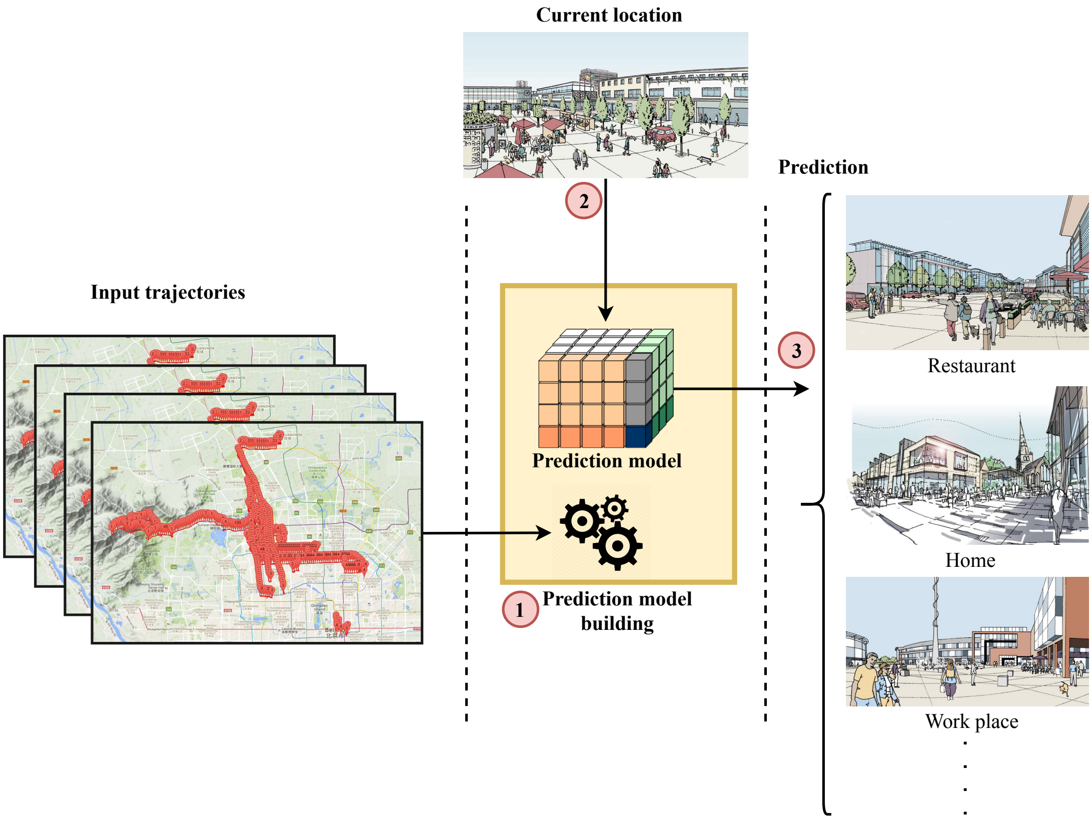


Figure 2.3: Location prediction: a big picture.

people tend either to move slowly or not to move at all. Moreover, the interest places that are closest to each other can be considered as one RoI, as in R_4. Some places are not considered as interest regions such as the place between R_3 and R_4. Here, it should be mentioned that there is a set of conditions such as the distance, spent time, etc. that any region must meet in order to be considered as a RoI.

2.2.2 Location Prediction

Location prediction is the most common research area based on the location histories of users. Predicting user's next location is mainly a proactive function that takes user current location in addition to other information (such as previous locations or user daily, weekly or monthly movement times and trends, etc.), and predict the next location or destination that will be visited, Figure 2.3 [1].

Generally, users can be classified into three different types according to the

[1] Pictures source: http://www.stephenpeart.co.uk/

predictability of their daily routine: predictable users, expected users and random users (Pollini and Chih-Lin, 1997). Predictable users follow regular routines between their home place and workplace during their working days. For this type of users, it is easy to accurately predict their next locations. The last two types are characterized by highly complex movement and usually a larger number of visited locations. Therefore, it is difficult to accurately predict their next locations. Hence, a certain likelihood of being in a predicted location is provided. Towards building efficient prediction methods for these users, their mobility data must be studied with more sophisticated approaches.

2.2.3 Wandering Detection

The advances in mobile computation and the self-location capability of devices such as smart phones and wearables have paved the way for a variety of health and well-being related mobile apps. Moreover, highly sensorised environments (such as smart cities and buildings) and the use of advanced artificial intelligence lead towards cognitive computation. As a result, the concept of smart health will play a key role in the future society (Solanas et al., 2014b).

One of the areas where computer scientists and engineers are being concentrated on is the welfare of elderly people, specially those suffering from MCI (Maioli et al., 2007). Such people, while still being able to cope with their habits, can suffer temporal episodes of disorientation and, even in some cases, get lost. Hence, tracking and monitoring systems have emerged as a necessary solution to assist elderly people during their mobility issues.

Most monitoring/alarm tracking systems only inform about the patient's location or when exiting from safe areas (the patient's neighbourhood, the park next to their house, etc). However, such systems do not consider the detection of wandering episodes: abnormal behaviours and erratic routes that can cause a great risk for the patients (Algase et al., 2007). Indeed, the appearance of wandering and the level of dementia are closely related (Algase et al., 2001; Lai and Arthur, 2003; Cipriani et al., 2014). The literature counts with a number of techniques to determine whether the elderly people are behaving abnormally with respect to their mobility patterns, and to identify abnormal or wandering behaviours [2].

2.3 Related Work

Mining trajectory data generated by a mobile user has been the subject of much research. This section aims to survey the literature on a set of related research that I considered in trajectory analysis and prediction: discovering RoI methods, location prediction models and wandering detection methods.

2.3.1 Methods for RoI Discovery

Several research papers and studies discussed the problem of discovering RoIs. The proposal in Marmasse and Schmandt (2000) represents an early work in this area. Their method uses the loss and regain of GPS signals within a certain radius to infer indoor places as buildings. Density-Joinable (DJ) (Gambs et al., 2012) and k-means (Ashbrook and Starner, 2003; Kang et al., 2004) clustering algorithms were used to discover RoIs. k-means clustering algorithm is not tailored for geolocated data where the grouping depends on the mean of all points that belong to the same cluster. Thus, using k-means to discover RoIs could cause missing some RoIs. Additionally, some locations that do not carry semantic meaning might be detected as a RoIs.

The time-distance threshold method was used in Huang et al. (2013) to extract the stay point (a sequence of GPS records in a spatial limited area). An area could be considered as a stay point if the stay time was greater than or equal to the time threshold and the distance between that area and the centre point of the GPS records was smaller than the distance threshold.

In Li et al. (2008); Yuan et al. (2013), the authors proposed a stay points detection method, called LiSPD in this thesis. The stay points were detected by seeking the region where the user has spent a period more than a predetermined threshold providing that the distance between the start and end points of the region is under

[2] In this thesis, for the sake of brevity, such variety of erratic behaviours will be referred to as "wandering".

a specific threshold. For each detected stay point, the mean coordinate, arrival and leaving time were computed. Giving two points in the trajectory, if the thresholds are not satisfied, the whole region between these points was ignored and consequently some information was lost. In our proposed method, these points are considered as a path and are associated with the nearest RoI, see Section 8.2.1.

2.3.2 Location Prediction Models

During the recent years, predicting people's next locations has received a rapid and increasing attention from the research community and industry. Many studies have discussed the usage of machine learning techniques in building the prediction model. We will select a significant subset of those contributions (*i.e.* seminal papers or those appearing in relevant journals). The models proposed in the literature can be classified into two types: (i) probabilistic models, such as Markov model and (ii) supervised learning models, such as association rules, Support Vector Machine (SVM) and Neural Networks (NN).

2.3.2.1 Probabilistic Models

A probabilistic model is a statistical analysis tool that provides a distribution of possible outcomes on the basis of past data. MC model is used in Gambs et al. (2012); Asahara et al. (2011) and Ashbrook and Starner (2003) to predict people's next location. The concept of n-Mobility Markov Chain (n-MMC) is proposed in Gambs et al. (2012) which depends on the sequence of n previous visited locations. Three different datasets were used to test the model: Phonetic (Killijian et al., 2010), GeoLife (Zheng et al., 2010) in addition to a synthetic (simulated) one. The model started by discovering RoIs using DJ clustering algorithm. Then, n-MMC was constructed based on the probabilities of transition among the discovered RoIs. The results showed that the prediction accuracy obtained from GeoLife dataset was 69% and it was achieved when the number of previously visited locations was two ($n = 2$).

Using a Mixed Markov-Chain Model (MMM) is proposed in Asahara et al.

(2011). MMM takes into account the effects of people's previous status as observable parameter and the user's own movement as an unobservable parameter. The proposed approach was evaluated using simulated and real tracing datasets, the latter was collected using indoor-GPS devices for 691 participants in a shopping mall. Pedestrians were classified into groups based on their mobility traces. For each group, different Markov models were generated. The pedestrian's next location approach works by first identifying the group a particular pedestrian belongs to and then the location most likely to be visited. However, relying solely on the last location to predict the next one greatly affects the prediction results. In order to better predict user's next location in a more precise manner, a model should typically take into consideration a *sequence of the recent locations*, and a higher-order Markov model should be modeled.

The previous models focus on predicting user's next location using only locations sequences but did not consider or incorporate information about movements daily, weekly or monthly time (*i.e.* when in the day, the week or the month they often occur).

Hidden Markov Model (HMM) is utilized in Mathew et al. (2012) to propose a hybrid method for predicting a person's next location. A real-life mobility dataset obtained from GeoLife project was used to evaluate the proposed method. Location characteristics were considered as an unobservable parameter, while an person's previous status was considered as an observable parameter. Unlike the previous models, the collected data were divided according to the associated timestamps into three clusters: day-time weekday, night-time weekday and weekend. A Hierarchical Triangular Mesh approach was used to divide earth surface into a set of triangular regions, where each region was represented by a single numeric ID. The locations in each cluster were converted into discrete codes associated to specific regions. Each cluster was used to train one specific HMM. The probabilities of all possible next locations were calculated and next location with the highest probability was returned.

Gao et al. (2012) proposed a location prediction model that considers the spatio-temporal contexts of a user's trajectories. The proposed model was evaluated using a mobile dataset provided by Nokia Mobile Data Challenge which contains 80 users over one year of time. The authors divided the temporal context into hour of a day and day of the week, and estimated their distributions using Gaussian distributions. The probability of visiting a given location depends on the distributions of the hour and day in addition to the current visited location. Unlike our proposed models in this thesis, in their paper, the authors did not consider the previous visited times and locations, which have strong relationship with the next location prediction.

Finally, both location and activity transitions of users were taken into account in Huang et al. (2013). A dataset of 100 users, using GPS devices for two weeks, was used to test the proposed model. The stay points were extracted from the collected data using time-distance threshold method. The activities of the users were considered as unobservable states, while the locations were considered as observable states. The destination the user will visit is predicted by estimating the next activity first.

2.3.2.2 Supervised Learning Models

Supervised learning models such as association rules, SVM and NN have been applied to predict people's location.

Morzy (2007) used association rules to predict the next location of a moving object. The Network-based Generator of Moving Objects was used to generate different instances of synthetic moving objects datasets. The moving objects' trajectories were obtained by dividing the movement area into a set of rectangular regions (cells) of fixed sizes. Each cell was assigned a discrete coordinate to identify the position of the cell in the movement area. Traj-PrefixSpan algorithm (a modified version of PrefixSpan algorithm (Pei et al., 2001)) was used to discover frequent trajectories that were used to generate the movement rules. An FrequentPattern-Tree index structure (Han et al., 2000) was used to speed up the process of looking for the trajectories discovered by Traj-PrefixSpan algorithm. To determine the next location, different matching strategies were used between the trajectories of the moving object and the generated movement rules. The results showed that the accuracy was close to 80%.

In Monreale et al. (2009), the authors used the frequent sequential pattern algorithm (Agrawal and Srikant, 1995) to extract the frequent movement patterns, called Trajectory-patterns (T-patterns). The T-patterns were ordered in a prefix tree. Each node in the tree contains the dimensions of the visited region ID, support and the pointers to another regions appear sequentially in the trajectories. The best T-pattern path that matches the given moving object trajectory was obtained. The score for each node in the relative path was calculated based on support value. Then the children of the best node are selected as next possible locations. A dataset obtained from GeoPKDD project (Giannotti et al., 2009) was used. The dataset includes trajectories of 1,700 cars which were equipped with GPS receiver for over a week.

In Daoui et al. (2013), the authors proposed a technique to predict movement in mobile network based on association rules mining. The histories of mobile user's movement that were recorded in the core base station network were used for evaluation purposes. These histories contain the dimensions of mobile user id, source cell, destination cell and travel date. For each mobile user, the movement history was arranged in a descending order from the newest to the oldest movements. A list of elements was generated where the first cell was the current location of the mobile user and the last cell was the first location recorded. Apriori algorithm (Agrawal and Srikant, 1994) was applied to extract the association rules that satisfy the specified minimum support and confidence. The best accuracy results were obtained for the association rules of order three. Thus, the next cells of the mobile user could be predicted if the two last cells crossed were known. As a result of the prediction, the network resources were effectively managed. Instead of reversing the resources in the cells that the mobile user might not cross, the network could reserve the resources only in the appropriate time when mobile user visits the predicted cells.

Another relevant contribution in this area is Kedia (2012), which proposed an object tracking approach for wireless sensor network using association rules technique. A simulation model was designed to generate an object and a set of randomly distributed sensors. The sensor nodes were used to record and transmit the object movement information across the monitoring environment to the sink. Apriori algorithm was applied to discover association rules from the transaction database recorded in the sink. The discovered association rules were used to predict the next location of the object. Kedia proved that the approach could save significant amount of energy where only the sensor nodes in the predicted location must be active and the other nodes can stay in sleep mode.

Ryan and Brown (2012) applied the association mining rules to the problem of occupant location prediction. The proposed method was evaluated using three types of datasets: simulated, University College Cork (UCC) and Augsburg dataset. In the association rules combination phase, Apriori algorithm was initialized with all itemset of attributes size + 1 instead of standard 1-itemsets. In their paper, the authors assumed that a visited locations sequences always occurs at the same time. Thus, the existence of shifts in the transitions time were ambiguous for their model.

A major shortcoming of the previous methods was that if no appropriate rule or route was matched, they failed to predict the next locations.

Location/contact traces (location, stay duration and social contacts) were utilized in Vu et al. (2014) to build a prediction framework for people's future contextual information movement. A scanning system based on Wi-Fi and Bluetooth scanners was built to collect Wi-Fi access point MAC addresses and Bluetooth MAC addresses, respectively. The sensing traces dataset was collected for 123 participants for 6 months. University of Illinois Movement (UIM) clustering technique (Vu et al., 2011) was used to cluster Wi-Fi traces data into locations. A regular location was considered if the user visits the location at the same time for at least a specific number of days. The dataset rows that belong to the same period of sensing time were assigned to the same locations a user visits. These records were used to train supervised model such as SVM, Naive Bayesian, and k-NN. The experiments showed an increase in performance.

NN has shown to be useful in location prediction model (Parija et al., 2013c,a,b).

The authors used NN technique to predict mobile user's movement in cellular communication network. The evaluation of the prediction scheme was conducted on a dataset obtained from the Mobile Host(MH) history of movement pattern. The dataset contains two mobiles MH1 and MH2 with regular and random movement patterns, respectively. This prediction was used to automatically update the location information of the mobile user which reduces the traffic load in cellular communication network. Unlike their model, where the time factor is not considered, the models proposed in this thesis have incorporated space and time interval sequences with the recurrent structure resulting in much more prediction accuracy.

Building local and global predictors to predict a person's next movement based on NN is proposed in L. Vintan and Ungerer (2004). Movement histories of four persons of the research group at the University of Augsburg are used to evaluate the neural predictor. In their model, they use the simplest multi-layer perceptron with one hidden layer trained with Backpropagation algorithm. The bit encoding is used to represent the rooms and the persons. Their evaluations showed that the local predictor overcomes the global predictor with accuracy of 92.32% and 87.3% respectively.

Some previous studies are based on RNN which are useful with sequential data due to the neurons' internal memories that are used to maintain information about the previous input. H. Kaaniche (2010) introduced a mobility prediction model in wireless Ad Hoc networks. Random Waypoint Mobility (RWM) model was used to generate location time series dataset for evaluating the efficiency of the mobility predictor. The location prediction could be used to estimate the expiration time of the links connecting the nodes enabling them to select the most stable paths which improves routing performances. Unfortunately, the author did not present details on the results.

Another recent work in which RNN was used is introduced in Liu et al. (2016). This work proposed a global prediction model called Spatial Temporal Recurrent Neural Network (ST-RNN) for predicting where users will go next. Two typical datasets called Global Terrorism Database (GTD) and Gowalla dataset were used to evaluate the effectiveness of ST-RNN model. The recurrent structure was utilized to capture not only the local temporal contexts but also the periodical ones. The spatial and temporal values were divided into discrete bins in order to produce the time-specific and distance-specific transition matrices. The corresponding transition matrix was calculated for each specific temporal value in one time bin and similarly for each specific spatial value. Unlike ST-RNN, in our proposed models, the space and time features were fed directly into the network and the network itself is responsible for learning their internal representation. Apart from prediction accuracy, the computational complexity of NN and RNN is extremely high. In contrary, some other models such as MM and Lempel-Ziv (LZ) attracted more attention due to the low resource consumption and complexity (Song et al., 2006).

Recent research in computer vision has successfully addressed the challenge of predicting the future locations of objects using RNN. In Alahi et al. (2017), multiple Long Short-Term Memory (LSTM) were generated for each individual to learn the movement pattern. Accordingly, a social pooling layer was introduced to share the information between the individuals' LSTMs. Lee et al. (2017) proposed using a single end-to-end trainable neural network model that predicts the future trajectory of multiple interacting objects. Finally, Vemula et al. (2017) describes an attention-based trajectory prediction proposal based on a RNN to model both spatial and temporal aspects of trajectories in human crowds. In this proposal, many model parameters are used and, hence, the computational complexity increases.

The main drawback in most of these models is that only the space context is considered, missing other information of trajectories (such as timestamps) which is important to build accurate prediction models. Additionally, discovering RoIs in people's mobility data has not been considered in most of previous mobility prediction studies. Because of the intensive computations of some techniques to train, in addition to mobile devices restrictions on resource consumption, some algorithms may sacrifice prediction accuracy for a higher prediction speed.

2.3.3 Wandering Detection Methods

In this section, we describe a selection of research publications proposing methods and systems that detect wandering behaviour and support elderly people during their outdoor movements. Studies vary in depth between methods and heuristics focusing on locations and trajectories, assessed using trajectories datasets, whereas some others address real implementations of prototypes and products.

The earliest works in this field began to exploit the combination of mobile phones and GPS receivers to track elderly people. Patterson et al. (2004) proposed Opportunity Knocks (OK), to guide and assist people with MCI when they are hesitant about their destinations. Smartphones together with physically separated sensor beacon devices were carried by patients. Firstly, they were asked to specify where they wanted to go. A Hierarchical Dynamic Bayesian Network model was used to predict the on-going route using their previous routes.

In Shimizu et al. (2000), a location system to help caregivers find the patient's current position on a map was developed. The main difference from the previous systems is the way to build the trajectory models and the predicting methods.

The iRoute system (Hossain et al., 2011) was proposed to track people with dementia during their outdoor movements and assist them in case of disorientation. The system was capable of learning new routes and guide the patients towards learned routes if they were lost. The system followed a Belief-Desire-Intention agent model using the preferences and historical records of wanderers.

A simpler solution addressing dementia is OutCare (Wan et al., 2011), which raised alarms when significant deviations from the daily routines were found. The system was tested with dozens of participants, but aged under 50 (not from elderly groups) and the deterioration capabilities were not mentioned. However, the lack of elderly participants leaves questions regarding the system validation.

The SIMPATIC project (Martínez-Ballesté et al., 2015) focused on the development of an autonomous system that monitored real-time trajectories from people with dementia, also counting with an application for the caregivers that received alarms under certain circumstances. A server that processed the locations received, extracted features from the on-going trajectory, and raised alarms to the caregivers' application when needed. The system was tested with 16 patients diagnosed with early or middle stages of dementia from the area of Tarragona (Catalonia, Spain).

Lin et al. (2012) presented a method to determine if the people with MCI were wandering by searching sharp changes of directions along with their GPS traces. This work was based on the assumption that inefficient patterns (*e.g.*, random, pacing and lapping) have a loop-like locomotion nature, and the direction changes are highly frequent in this kind of patterns.

In Sposaro et al. (2010), Bayesian theory was used to calculate a wandering probability. The authors implemented the iWander application, which asked the wanderer if he/she was disoriented when a possible wandering behaviour was detected. In case of disorientation, the application guided the patient to a safe area and then notified caregivers. In contrast to the SIMPATIC solution, iWander needed the interaction of the wanderer with button prompts, which may pose some trouble for elderly.

LaCasa (Hoey et al., 2012) used Markov decision process and contextual information to provide wandering assistance, whereas learned from the trajectories of the wanderers using Bayesian methods. The authors assumed that the individuals were at a known location as long as their smartphones were connected to a known Wi-Fi. This assumption was not always true where individuals might be wandering even in well-known areas.

Lin et al. (2015) proposed a method called Isolation-Based Disorientation Detection method to detect abnormal trajectories. Patient's trajectories previously collected were modeled as a graph in which the vertices are the frequent visited locations, and the edges are the routes among those locations. Presence of looping inside the graph or deviation of a defined route were considered as potential instances of disorientation.

While tracking elderly people is considered as a violation of privacy rights and loss of independence, the relatives and caregivers consider such tracking as a solution to keep elderly people safe. In this context, researchers proposed balanced solutions to support and consider the views of all parties (Landau and Werner, 2012; Doyle et al., 2014).

2.4 Deep Learning

Deep learning is a subfield of machine learning that uses the structures of the human brain to process data and gain certain types of knowledge. Generally, deep learning is used to tackle one of the most challenging task related to the automatic learning of feature representations which can then be readily used for prediction tasks. Deep learning architectures have achieved great success in many sequence modeling tasks such as image and speech processing (Krizhevsky et al., 2012; Hinton et al., 2012), time series prediction (Yümlü et al., 2005; Barbounis and Theocharis, 2006), machine translation (Luong et al., 2015), sequential click prediction (Zhang et al., 2014) and natural language processing (Mikolov et al., 2011), to name a few.

In the next subsection, we explain what embedding representations learning is. Then, two of the most common architectures of deep learning methods are described: RNN and CNN.

2.4.1 Embedding Representations Learning

Hand-crafted features, designed beforehand by human experts, are time-consuming task. Features are often incomplete and over-specified. Furthermore, the hand-crafted features design often involves finding the right trade-off between computational efficiency and accuracy. If machine learning approaches could learn features automatically, the entire learning process could be more easily and many more tasks could be solved. Deep learning provides the ability to learn features automatically.

Representation learning (Mikolov et al., 2013), also called embedding learning, is a set of techniques that learn features by transforming raw data input to real valued vectors that take place in machine learning tasks. By using the features, the

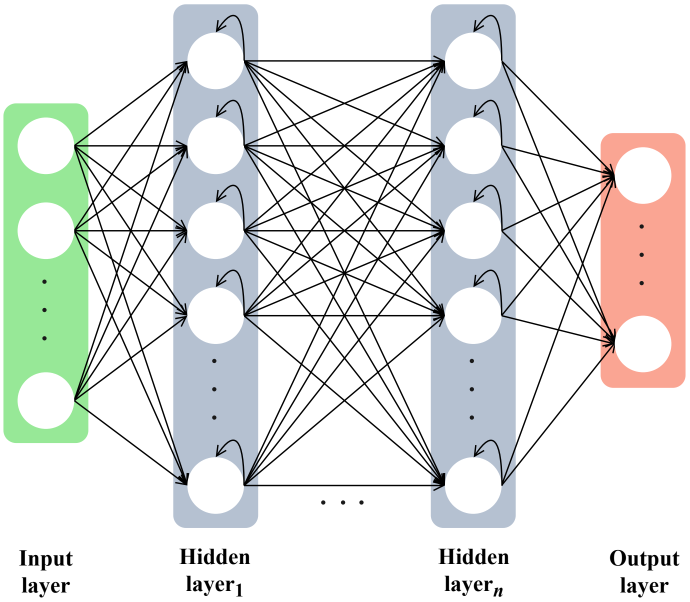


Figure 2.4: General RNN architecture.

machine learning model can learn a specific task and learn the features themselves as well. The embedding learning technique is used in this thesis to effectively discover adequate representations of trajectory data features to capture the embedded semantic information about the people's movement.

2.4.2 Recurrent Neural Networks

RNN is a type of advanced artificial neural network which designed to model sequences due to the ability to remember important things about the received input in its internal memory. RNN is designed to extract the embedded semantic information of the sequential data and then the patterns that used to predict the next likely scenario. In this thesis, the recurrent structure is incorporated with trajectory data sequences in order to keep track of people's movement history which allows to discover more meaningful dependencies (see Chapter 4).

The general architecture of RNN is as follows, see Figure 2.4. A RNN consists of three main layers: an input, an output and a hidden layers. RNN receives a data sequence as input and transforms them through a series of hidden layers. Every layer is made up of a set of neurons that have learnable weights and biases. Each layer is fully connected to all neurons in the previous layer. Lastly, the last fully connected

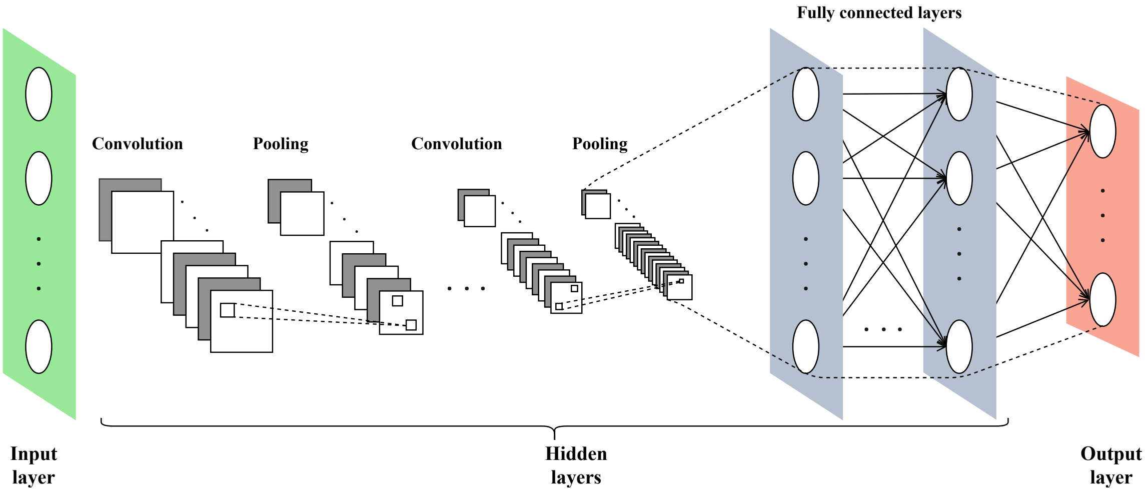


Figure 2.5: General CNN architecture.

layer is the output layer that used to represent the predictions.

2.4.3 Convolutional Neural Networks

CNN, also known as a ConvNet, are a class of deep neural networks which are especially adapted to various computer vision tasks because of their ability to abstract representations with local operations. The use of CNN has achieved significant success in various applications such as computer vision, speech and natural language processing. Although CNN is originally designed to cope with image data, it can be used for sequence modeling tasks such as location prediction (Chapter 7).

Figure 7.1 shows the CNN architecture which is a bit different from RNN architecture. Like other types of artificial neural networks, a CNN has an input, output and various hidden layers. However, a hidden layer can include multiple types of layers: convolutional, pooling, fully connected and normalization layers. Unlike RNN, the layers have neurons organised in three dimensions: height, width and depth. The neurons in one layer connect to a small region of the next layer not to all its neurons. Finally, the final output will be reduced to a single vector of probability scores, organised along the depth dimension.

2.5 Evaluation

In this section, we address the basics of evaluation, that will be used in our proposals in this thesis.

2.5.1 Evaluation of Location Prediction Models

The prediction evaluation procedure starts with reading the previously visited locations (represented by their IDs) in each trajectory sequentially. Then, the next locations to be visited are predicted based on these readings. Finally, the predicted locations are mapped into the real data.

Recall and Precision are employed as evaluation metrics in all experiments in order to assess the efficiency of the prediction models. The larger the value, the better the performance is. The **Recall@N** is defined as the ratio between the number of correct predictions (*i.e.* locations) over the total number of real visited locations. The **Precision@N** is defined as the ratio between the number of correct predictions over the total number of predictions.

To compute the Precision and Recall scores, first, a ranked list is populated with all potential next locations arranged in a descending order according to their probabilities. Then, the percentages of the times in which the real next location was found in the top-N most probable locations within the ranked list are calculated. For each user, the model predicts a list of locations N, given his/her last w-visited location and timestamps as input. In our study, we report $N = 1$, 2 and 3. We denote the recall and precision as R@N and P@N, respectively.

Supposing that L_u denotes the set of correspondingly real visited locations by a user u in the test data, $PL_{N,u}$ denotes the set of top N predicted locations and U is the set of users, the definitions of R@N and P@N are formulated as below:

$$R@N = \frac{1}{|U|}\sum_{u\in U}\frac{|L_u \cap PL_{N,u}|}{|L_u|} \tag{2.1}$$

$$P@N = \frac{1}{|U|}\sum_{u\in U}\frac{|L_u \cap PL_{N,u}|}{|PL_{N,u}|} \tag{2.2}$$

2.5.2 Evaluation of RoI Discovery Methods

Soundness and completeness properties are used as metrics to evaluate the effectiveness of the methods used to discover RoIs.

Definition 1 ***Soundness property.*** *The method to discover RoIs is sound only if it can find or discover interest regions.*

Definition 2 ***Completeness property.*** *The method to discover RoIs is complete if it can find all interest regions.*

Soundness and completeness are defined in the Equations: 2.3 and 2.4:

$$completeness = \frac{|correct\ RoI\ discovered|}{|all\ real\ RoIs|} \tag{2.3}$$

$$soundness = 1 - \frac{|incorrect\ RoI\ discovered|}{|all\ RoIs\ discovered|} \tag{2.4}$$

2.5.3 Wandering Detection Methods Evaluation

When applying wandering detection method given a user movement, it returns a prediction with two possible outcomes: (i) the movement contains normal behaviour which is the default situation, or (ii) the movement contains abnormal behaviour. However, we know in advance whether the trajectory actually contains abnormal movement or not based on the labels associated with each trajectory. So, we validate whether the classification is correct, by comparing the prediction of the method and the trajectory's label. For this reason, we apply a binary classification. Since the goal of the method is to detect abnormal movement, we consider that the "positive" class is "abnormal", and the "negative" class corresponds to "normal". In practice, a classified trajectory falls into one of the four categories (see Table 2.1):

1. **True Positive (TP)**: an abnormal trajectory successfully classified as abnormal.
2. **False Positive (FP)**: a normal trajectory classified as abnormal.
3. **True Negative (TN)**: a normal trajectory successfully classified as normal.

Table 2.1: Confusion matrix.

		Real value	
		Abnormal	Normal
Prediction value	Abnormal	True Positive	False Positive
	Normal	False Negative	True Negative

4. **False Negative (FN)**: an abnormal trajectory classified as normal.

From the evaluation perspective, we consider some statistical measures that can be derived from the confusion matrix (a technique for summarizing the performance of a classification method) after classifying each trajectory. In our evaluation, we analyse the following measures:

1. **Recall or Sensitivity**: Proportion of actual abnormal trajectories that are predicted as abnormal.
2. **Precision**: Proportion of abnormal predictions that are actually abnormal.
3. **Specificity**: Proportion of actual normal trajectories that are predicted as normal.
4. **Accuracy**: Proportion of trajectories that are predicted correctly, both abnormal and normal classes.
5. **F1-score**: Harmonic mean between recall and precision. This measure is widespread used as an indicator of the test's accuracy.

Table 2.2 presents the mathematical expressions of sensitivity, specificity, precision, accuracy and F1-score.

Table 2.2: The mathematical expressions of sensitivity, precision, specificity, accuracy and F1-score

Metric	**Expression**
Recall or Sensitivity	$TP/(TP + FN)$
Precision	$TP/(TP + FP)$
Specificity	$TN/(TN + FP)$
Accuracy	$(TP + TN)/(TP + TN + FP + FN)$
F1-score	$2TP/(2TP + FP + FN)$

Table 2.3: Datasets description.

Name	Number of individuals	Acquired locations	Trajectories	Time span of the collection	Sampling rate (s)
GeoLife (Zheng et al., 2010)	182	24,876,978	17,621	April 2007 to August 2012	1-5
Gowalla (Cho et al., 2011)	196,591	6,442,890	N/A	February 2009 to October 2010	N/A
SIMPATIC (Smart Health Research Group, 2014)	18	251,708	653	December 2013 to June 2016	180
OpenStreetMap (Open Street Map, 2018)	1,000	2,772,798	5,152	Over one month	N/A

2.5.4 Cross Validation

Cross validation has been widely used to evaluate machine learning techniques. It is used to give an indication of how well the model will predict unseen data. Cross validation is done by partitioning a dataset and using a subset to train the method and the remaining data to test it. The common cross validation technique that we use is called **k-fold cross validation**. In this technique, the data are divided into k folds. One of the folds is used for testing and the remaining folds for training the method. This scenario is repeated k times.

2.6 Mobility Datasets

There are many online publicly available real-world datasets that contain data on people's locations. Table 2.3 summarizes the datasets used in this thesis. It shows the number of contributing individuals of the dataset, the GPS points and trajectories they contain, they time span of location acquisition and the sampling rate. N/A is used when this information is not available.

- **GeoLife** dataset (Zheng et al., 2010) is an open-source large real life GPS trajectory dataset belongs to GeoLife project (Microsoft Research Asia). The dataset was collected by 182 users in a period of over five years (from April 2007 to August 2012). It contains 24,876,978 recorded points and 18,670 trajectories with a total duration of 50,176 hours and a total distance of 1,292,951 kilometers. The collected dataset covers different cities located in China, USA and Europe but the majority of the data was from Chinese cities. These trajectories were recorded by different GPS-phones and GPS loggers with high sampling ($5 \sim 10$ meters or every $1 \sim 5$ seconds). The GPS trajectories are represented by a sequence of time-stamped points, each of which contains the dimensions of latitude, longitude, altitude and other information. This dataset is widely used in many research fields, such as mobility pattern mining, user activity recognition and location privacy, among others. However, as we have observed, its usage in wandering detection is scarce (Martínez-Ballesté et al., 2018).

- **Gowalla** (Cho et al., 2011) is an open-source data which provides a check-in records contain a location identifier, the corresponding check-in timestamp and other information. This dataset contains 6,442,890 check-ins over the period from February 2009 to October 2010.
- **SIMPATIC** project dataset (Smart Health Research Group, 2014), which contains the daily trajectories (from end 2013 to mid 2016) of 18 Catalan individuals suffering from mild cognitive impairments (MCI), gathered during the course of the SIMPATIC project. The dataset contains around 2000 trajectories with low sampling (3-minute rate), for design reasons (Batista et al., 2015).
- The **OpenStreetMap** dataset (Open Street Map, 2018) is a collaborative project aiming at creating and distributing free geographical data worldwide, which already contains more than one million trajectories from thousands of individuals around the world since 2005. Most of the trajectories have high sampling (1-10 seconds) and, hence, are detailed trajectories. For the sake of feasibility, for the further analysis, we have randomly selected a subset of 1,000 individuals from more than one million individuals in the original OpenStreetMap dataset.

2.7 Summary

In this chapter, we have presented a background on trajectory analysis and prediction methods, defining the most basic form of deep learning, the evaluation procedures and mobility datasets. In the next chapters, we present several methods for predicting people's trajectories based on deep learning technique.

Part II

Contributions to Deep learning Models for Location Prediction

CHAPTER 3

Preliminary Matters

3.1 Introduction

Different common machine learning techniques have been used to predict people's next location such as MC, LZ family algorithms (LZ, LeZi Update and Active LeZi), rule-based approaches and NN. One of the significant challenges while predicting people movements is related to how to adapt machine learning techniques in the context information of movements. Additionally, building a (one-size for all) accurate prediction model for all users is hard and sometimes impossible because predicting next location is a user-specific problem. Even if the visited locations are similar for many users, a trajectory of one user visiting different locations is most likely unique. Thus, building one prediction model for each user could be desirable. Usually, building the model, *i.e.* discovering the frequent trajectories and locations, is performed off-line while the prediction itself is performed on-line.

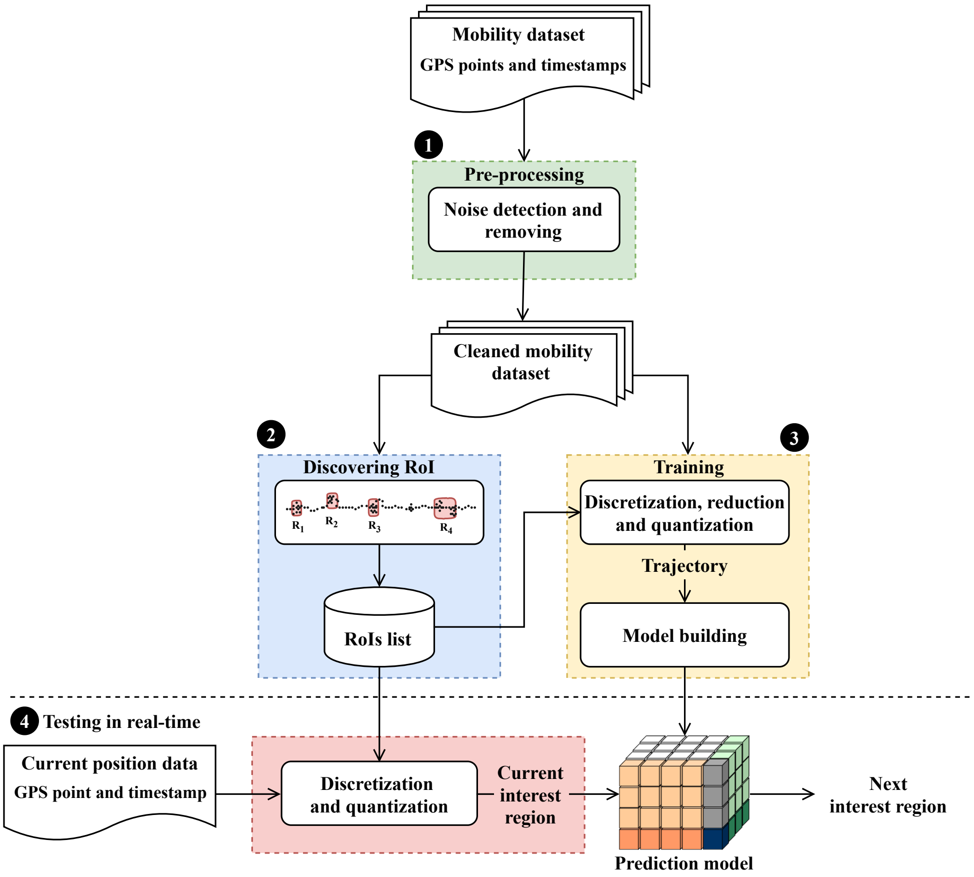


Figure 3.1: General location prediction model architecture.

In this chapter, we describe the main location prediction steps: data preprocessing, discovering RoIs, building a prediction model and model evaluation. All these steps, except building the prediction model, are carried out in the same manner in all the prediction models proposed in this thesis.

3.2 Location Prediction Model Architecture

Figure 3.1 shows our proposed approach which includes four main steps:

- First, the dataset is preprocessed to detect and remove possible noise.
- In the second step, RoIs located in a user movement region are discovered.
- The third step involves several sub-steps: discretization, reduction and quantization of the training dataset, and then building a prediction model.
- The last step involves evaluating the prediction model using a testing dataset.

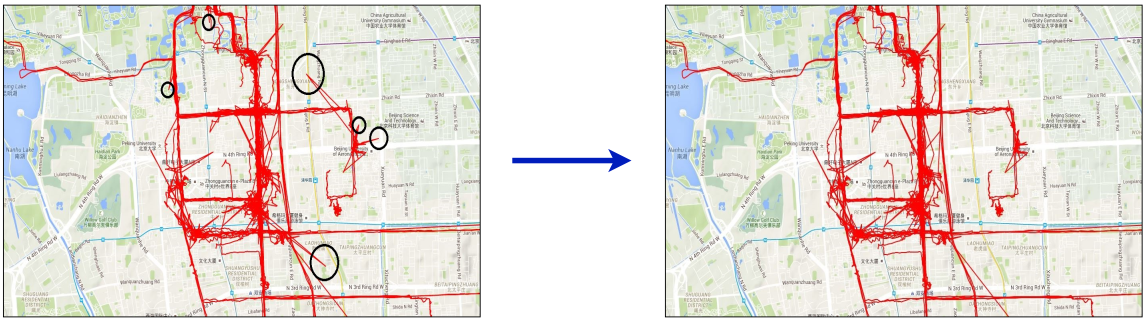


Figure 3.2: Noise detection and removal

Definition 3 ***GPS Points.*** *A GPS point is a tuple* $G = (x_i, y_i, t_i)$ *where* (x_i, y_i) *denotes a GPS coordinate,* x_i *is latitude,* y_i *is longitude, and* t_i *is the timestamp when GPS coordinates were recorded.*

3.2.1 Preprocessing

In mobility data preprocessing stage, dataset is converted into a suitable form for further processing. Noises that may come from different sources are removed. Such noise can be due to mispositioning software or hardware system, or losing GPS signals due to indoor usage or mobile device battery drain or problems. Once the dataset is cleaned, RoIs can be discovered.

In general, mobility data are not accurate and consistent. For example, in a short period of time, the distance between two adjacent points could be large. The inconsistencies negatively affect the process of discovering RoIs and consequently the prediction process as a whole.

One of the mobility dataset we have used in this chapter was collected from various cities in China, USA, and Europe, but the majority of the data was from China (its cities are assumed to be our target cities for prediction model). The preprocessing step on this particular dataset is conducted as it follows:

- Measuring the user travel speed between each two adjacent points based on time and distance. As shown in Figure 3.2, if the speed is high (*e.g.*, 300km/h), the point is removed (Zheng, 2015).
- Calculating the distance between every point in the mobility data and the centre point of China (Figure 3.3), then the points that are located outside the

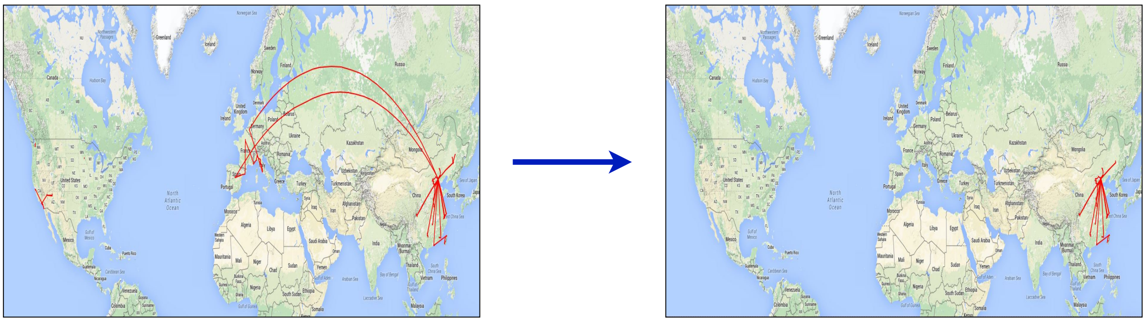


Figure 3.3: Removing the points outside our target boundary.

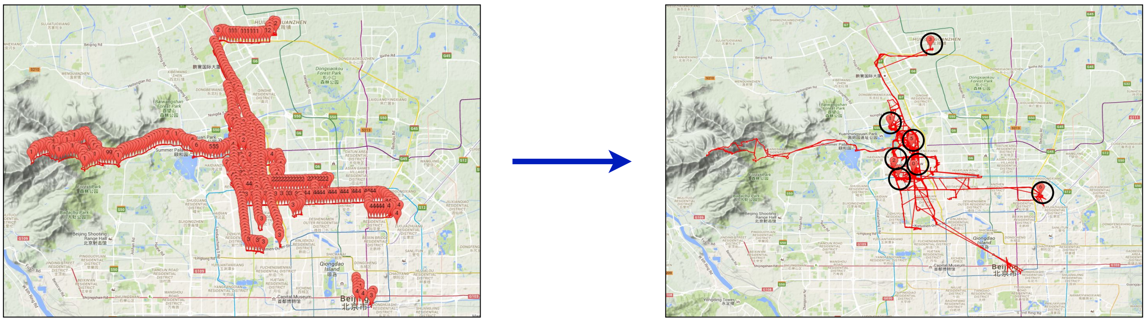


Figure 3.4: LiSPD method.

target boundary are removed.

3.2.2 RoI Discovery

Typically, mobility data should be processed first before it can be used in prediction models and, thus, each GPS coordinate in a user mobility data should first be converted into discrete values associated to specific RoI. The RoIs that are used to describe the movements of a certain user must be identified and eventually can be used to build a user's trajectories.

The GPS logs of each user are converted into trajectories (sequence of RoIs) by firstly detecting the *interest region*. To detect the interest region, we use the method proposed by Li et al. (2008); Yuan et al. (2013). The interest region represent those spatial regions where the user has stayed for more than a pre-determined time threshold providing that the distance between the start and end points of the region is under a specific threshold. The obtained region then are clustered into several geo-spatial regions using Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996). Each region is given a unique identifier.

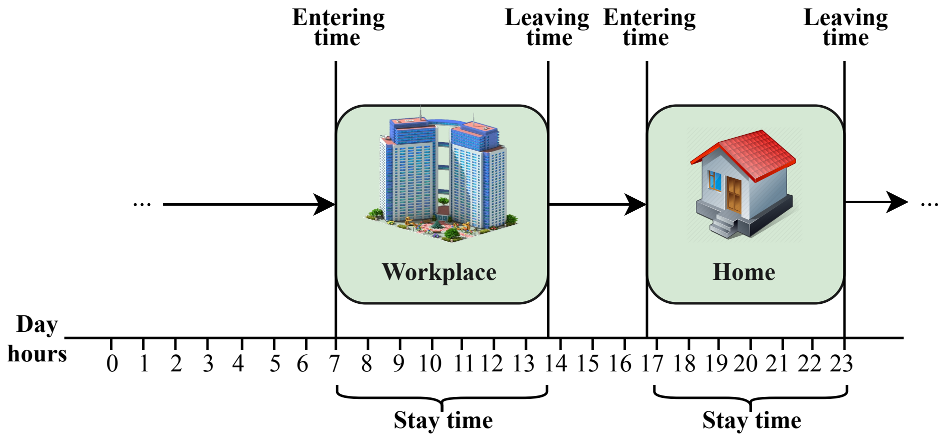


Figure 3.5: Movement time can be expressed as entering, staying or leaving times.

Finally, to formulate the region history of each user, the GPS points located in the same region are replaced by their identifiers.

Definition 4 ***Trajectories****. We denote the set of users by $\mathcal{U} = \{u_1, u_2, \ldots\}$, the set of RoIs by $\mathcal{R} = \{r_1, r_2, \ldots\}$ and the set of time intervals by $\mathcal{T} = \{t_1, t_2, \ldots\}$. A user trajectories are represented as a sequence of movements $\mathcal{M}^u = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, where $u \in \mathcal{U}$ and n is the length of the trajectories. Each movement $\mathcal{M}_i$ is represented as a tuple (r_i, t_i) where $r_i \in \mathcal{R}$ is the RoI identifier associated with its GPS coordinates (x_{r_i}, y_{r_i}) and $t_i \in \mathcal{T}$ is the time interval obtained from the GPS timestamp.*

3.2.3 Prediction Model Building

The third step is to present the process of building the prediction model. Various sub-steps are involved before building the prediction model: discretization, reduction and quantization. In the discretization, the GPS points of training dataset are converted to the RoI identifiers they belong to. In other words, using the RoI list obtained from the previous step, the continuous values of the GPS training data are converted from a series of spatial points into discrete values that represent a sequence of RoIs that the user visits. With the discrete values, many types of calculations are more easily performed than the continuous one. After GPS points discretization, the RoI identifiers may possibly be repeated. Thus, the reduction is used to remove the similar successive RoIs.

Timestamp can be categorized according to the location into three different types: entering, staying and leaving times, as shown in Figure 3.5. We assume that the leaving time has the most impact on predicting the next location. The rationale behind this assumption is that the next location a user will visit depends on the current location and the time at which he/she leaves. This assumption is confirmed by the experiment results in the next chapters. Therefore, in the reduction, the timestamps associated with GPS points are ignored except the time before the movement to the next RoI (*i.e.* the leaving time from the current RoI). The obtained timestamps are then quantized into a specific time interval such as splitting the day into different timeslots or the week into weekdays and weekend etc. We used different time intervals such as the hour, weekday type, month of the year and timeslots in Chapter 5.

Predicting people's future location can be viewed as a sequence generation problem. Given a user u in a RoI r_i at time t_i, the task is to predict user's future RoI $\hat{r}_{i+1}$ on the basis of his/her historical movement records, which is processed in a sliding window from $\mathcal{M}^u_{i-w}$ to $\mathcal{M}^u_i$, by modeling:

$$P(\hat{r}_{i+1} = r_j | \mathcal{M}_i, \mathcal{M}_{i-1}, \ldots, \mathcal{M}_{i-w}) \tag{3.1}$$

where r_j is a RoI $\in \mathcal{R}^u$ and w is the number of visited RoIs taken as inputs to the model.

3.2.4 Prediction Model Testing

Before predicting the next location, user's current position (in real-time) or trajectories previously collected (in testing) are preprocessed. The GPS coordinates are discretized by converting them into their RoI identifiers using the RoI list, while the timestamps are quantized into specified time intervals. Then, the trajectory will be represented as a single or sequence of tuples (r_i, t_i) where r_i is the RoI and t_i is the time interval.

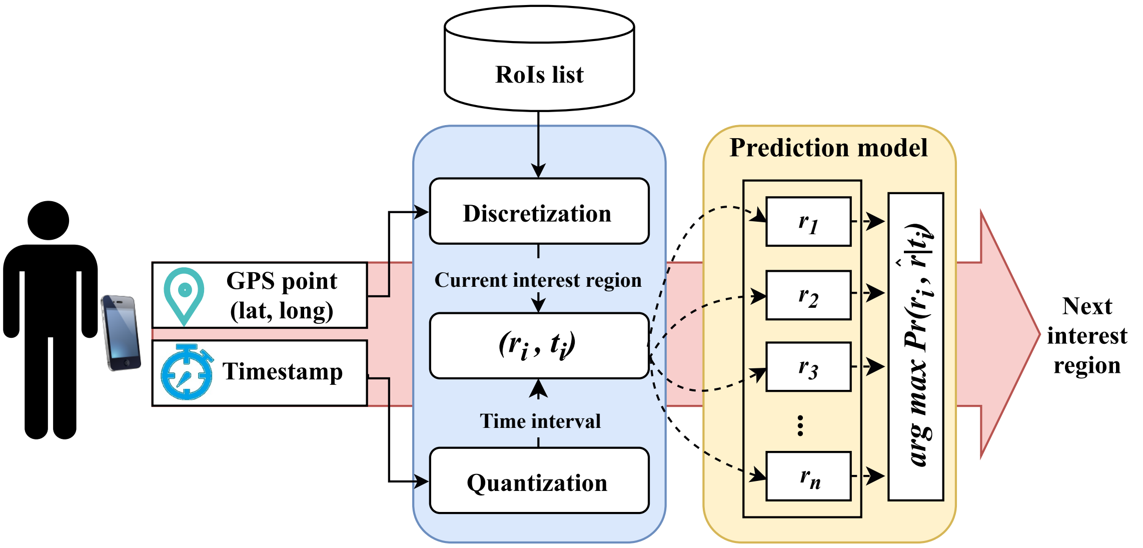


Figure 3.6: Next location prediction testing architecture.

To make prediction, as the final step, the test data is passed to the location prediction model. Then, a set of probability values is obtained. The RoI with the highest probability value is predicted to be the most likely next RoI. Figure 3.6 shows the overview of location prediction testing part.

3.3 Summary

In this chapter, we have described the main location prediction steps. The first step is data preprocessing that is used to remove possible noise. In the second step, RoIs located in a user movement region are discovered. The third step is the process of building a prediction model. In the last step, the prediction model is evaluated using a testing dataset.

CHAPTER 4

Recurrent Neural Network for Predicting People's Next Location

4.1 Introduction

Deep and neural learning methods have achieved remarkable results in many sequence modeling tasks such as image and speech processing (Krizhevsky et al., 2012; Hinton et al., 2012), time series prediction (Yümlü et al., 2005; Barbounis and Theocharis, 2006), sequential click prediction (Zhang et al., 2014) and network language modeling (Mikolov et al., 2011). Moreover, authors have reported that RNN achieve promising performance in comparison with the traditional counterpart. Due to the ability of RNN structure to represent the sequences, it is utilized in the proposed model in order to keep track of user movement history. Representation learning (Mikolov et al., 2013), also called embedding learning, is a set of techniques that learn a feature by transforming raw data input to real valued vectors that take place in machine learning tasks. By using the features, the machine learning model can learn a specific task and learn the features themselves as well. Embedding learning technique proved its ability to capture the embedded semantic information in people's mobility data in order to improve the performance of next location prediction models (Liu et al., 2016; Feng et al., 2017).

Some studies have focused on using a set of features to obtain a good prediction performance. For instance, in Vu et al. (2014); Do and Gatica-Perez (2014), a variety of hand-crafted features have been used. Such models lack the capability of extracting the embedded semantic information from people's mobility data. To overcome this drawback, a deep learning model has been utilized for automatically learning the best internal representations of the space and the time features.

The main contributions of this chapter are as follow. First, we use an embedding learning layer to discover adequate internal representations of the space and time input features while avoiding man-made representations. The space represents a specific location that has been visited by the user while the time represents the location visiting time. Second, we use RNN to model people movement and successfully incorporate the recurrent structure with space and time features into enhancing the model efficiency. Third, an extensive set of experiments is conducted on two large real-life mobility datasets in order to evaluate the efficiency of the developed model. STF-RNN is implemented using Theano (Team et al., 2016).

4.2 STF-RNN: Model Description

Figure 4.1 shows the graphical illustration of STF-RNN model. The model contains four layers: input, embedding, recurrent (hidden) and prediction layers as well as inner weight matrices.

The input layer consists of two vectors.The first one is $r_i \in \mathbb{R}^N$ which represents the identifier of the RoI where N is the number of RoIs. This vector is encoded

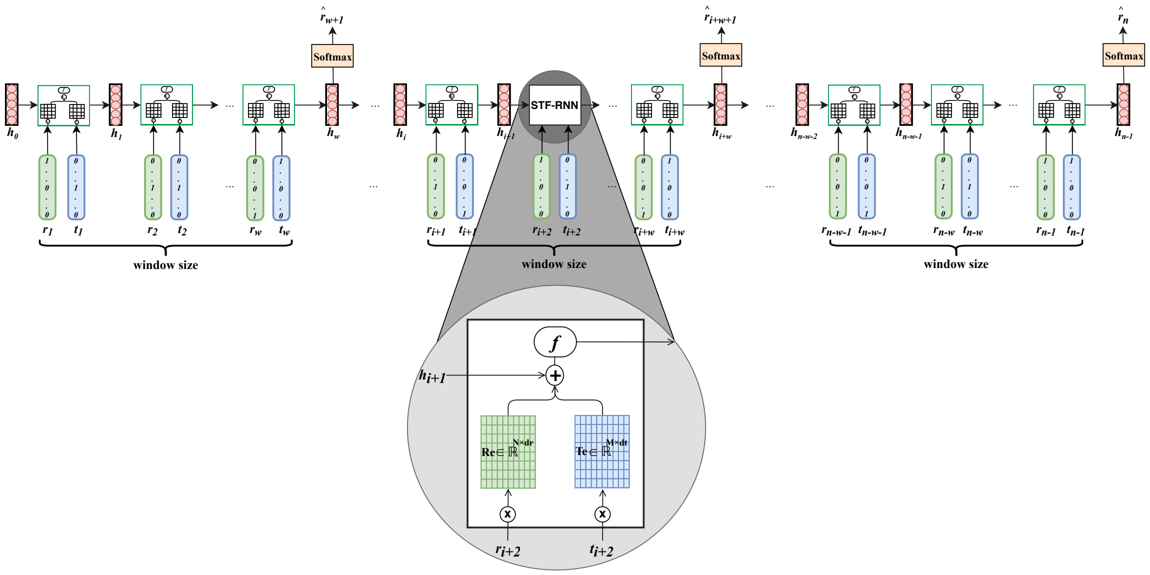


Figure 4.1: STF-RNN architecture.

using 1-of-N (or one-hot encoding) (Harris and Harris, 2012), which means for a given input data, only one out of the vector values will be 1, and all the others are 0. The second vector represents the leaving time (in hours) from the RoI. We denote this vector by $t_i \in \mathbb{R}^M$ and M is the number of different time intervals. It is encoded also using 1-of-M encoding technique. The time intervals represent the number of hours per day in which there are 24 time intervals (*i.e.* hours). In the one-hot vector representation, the RoIs (or the leaving times) are equidistant from each other and, hence, no relationship among them is preserved. To overcome this drawback, each input vector is passed through an embedding layer to produce a vector with d_r and d_t dimensions for RoIs and leaving times, respectively. The embedding layer maps the input vectors into real value vectors to learn a meaningful representation of the RoIs and leaving times input features. This representation enables the model to capture the embedded semantic information about user behaviour and as a consequence improving the prediction performance. Therefore, the trainable features will be used as input to further layers in the network rather than using one-hot vectors. More formally, let $\mathit{Re} \in \mathbb{R}^{N\times d_r}$ be the embedded matrix that represents a set of RoIs, where d_r is the dimensionality of the embedded vector of the RoI. The embedded vector

$re_i \in \mathbb{R}^{d_r}$ is given by multiplying the embedded matrix Re and the input vector r_i.

$$re_i = r_i \cdot Re \tag{4.1}$$

Similarly, the embedded vector $te_i \in \mathbb{R}^{d_t}$ is given by multiplying the embedded matrix $Te \in \mathbb{R}^{M \times d_t}$ and the input vector t_i. Here, Te represents a set of leaving time and d_t is the dimensionality of the embedded vector of the leaving time.

$$te_i = t_i \cdot Te \tag{4.2}$$

The recurrent layer $h_i \in \mathbb{R}^{d_h}$ is used to maintain the user movement history where d_h is the dimensionality of the recurrent layer vector. In STF-RNN, vanilla RNN is used due to the small size of input window size w.

The prediction layer $\hat{y}_i \in \mathbb{R}^N$ produces a probability distribution over the RoIs whose output length is the input vector r_i size. The values of the recurrent and the prediction layer are computed as below:

$$h_i = f\left(re_i \cdot W_r + te_i \cdot W_t + h_{i-1} \cdot W_{h_{i-1}} + b_h\right) \tag{4.3}$$

$$\hat{y} = g(h_i \cdot W_h + b_o) \tag{4.4}$$

In equation 4.3, $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_t \in \mathbb{R}^{d_t \times d_h}$ are the weight matrices between the input and recurrent layers, $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ is the recurrent connection propagating sequential signals and $b_h \in \mathbb{R}^{d_h}$ is the hidden layer bias. In equation 4.4, $W_h \in \mathbb{R}^{d_h \times N}$ represents the weight matrix between the recurrent and prediction layers and $b_o \in \mathbb{R}^N$ is the output layer bias.

Hyperbolic tangent (tanh) is used as the non-linear activation function for the recurrent layer due to its computational efficiency and effectiveness in literature (Collobert et al., 2011).

$$f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{4.5}$$

The prediction layer is a *Softmax* layer which is suitable for this case as its outputs can be interpreted as conditional probabilities (Bridle, 1990).

$$g(\hat{y}_i) = \frac{exp(\hat{y}_i)}{\sum_{j=1}^{N} exp(\hat{y}_j)} \tag{4.6}$$

The current input layer, as well as the previous state of the recurrent layer, is used to compute the next state of the recurrent layer. Thus, the next RoI prediction depends on not only the current input RoI, but also the sequential historical information. This property of RNN structure helps the model keep track of user movement history and discover meaningful dependencies and as consequence, enhance the model performance.

4.3 Learning Algorithm

In this section, the learning process of STF-RNN model with the Backpropagation Through Time (BPTT) algorithm (Rumelhart et al., 1986) is presented. As stated before, the next state of the recurrent layer is computed based on the current input layer as well as the previous state of the recurrent layer. The cost function used is the cross entropy which is defined as:

$$J = -\sum_{i=1}^{n} y_i \cdot log(\hat{y}_i) \tag{4.7}$$

where n is the number of training samples, y is the real user' next RoI, and $\hat{y}$ is the predicted next RoI probability. Because we have represented the RoI identifiers using one-hot vector representation, the cost function can be redefined as:

$$J = -log(\hat{y}_i) \tag{4.8}$$

ADADELTA update rule (Zeiler, 2012) is employed to estimate the model parameters. STF-RNN parameters are $\theta = [Re, Te, W_r, W_t, W_{h_{i-1}}, W_h, b_h, b_o, h_0]$, where h_0 is the initial vector for the recurrent layer. This process is repeated iteratively until reaching the convergence state.

4.4 Experiments and Results

In this section, large-scale experiments are conducted to validate STF-RNN model's effectiveness. The experimental results are reported in detail.

4.4.1 Experimental Settings

We use two publicly available real-world datasets in our experiments, *i.e.*, GeoLife and Gowalla.

The two datasets GeoLife and Gowalla have several general differences such as: the size or the number of records, input variables, the distribution of the points in the movement area, in addition to meta-data related to the collection process. The most important difference between them is that GeoLife should be processed in order to discover the RoIs. The RoIs in Gowalla dataset are given different IDs during collecting the data.

In order to use GeoLife dataset, the interest regions are detected as described in Section 3.2.2. The two DBSCAN's parameters were: maximum distance ε between any two points and the minimum number of points $minPts$ required to form a dense region. In this study, we set $\varepsilon = 100$ meters and $minPts = 3$.

Gowalla dataset is preprocessed by removing the users who had the same number of check-ins and RoIs. Moreover, users with less than 20 check-ins and less than 5 RoIs are removed from the dataset. Finally, the check-in records are organised as RoIs sequences.

Recall and Precision are employed as evaluation metrics in all experiments. The model of each user is trained using three-fold cross validation technique where mobility data is partitioned into three sub-data of equal size. The Recall and Precision scores of each case from each user is then calculated and the final results of all users are averaged.

To evaluate the model effectiveness, we compare STF-RNN with three outstanding proposals found in the literature [1]:

[1] We have implemented all baseline models based on our understanding of the original papers.

Table 4.1: Model parameters values.

Description	Values
Training epochs	100
Hidden layers size	40
RoI vector size	100
Time vector size	10
Window size	2

- **MC** (Ashbrook and Starner, 2003): It is the basic baseline method, that uses k-means for extracting RoIs in a user movement region and MC for location prediction .
- ***n*-MMC** (Gambs et al., 2012) is a classical sequential model which exploits the transition probabilities. n-MMC depends on the sequence of n previous visited locations.
- **HPHD**: It is proposed in (Gao et al., 2012) where spatial, hourly and daily information are used for location prediction.
- **AR**: It is proposed in (Daoui et al., 2013; Kedia, 2012) to predict next location of a moving object.
- **NN** (Parija et al., 2013c; Leca et al., 2015; L. Vintan and Ungerer, 2004) has been successfully applied in computer vision, speech recognition, etc.
- **RNN** (H. Kaaniche, 2010) is widely used for time series prediction.

The goal of these comparisons is to show how incorporating the recurrent structure with the space and time interval sequences in our model has improved prediction overall performance. For consistency, the common parameters of the models are given the same values, Table 4.1. For example, the second order MC is use which achieved better prediction accuracy as shown in Gambs et al. (2012) and Huang et al. (2013). The grid search method is used to evaluate various model parameters setting in order to select the optimum set of these parameters. The evaluated parameters include: the dimensionality of time-and-RoI embedded vector, the hidden layer size and the width of input window (number of visited RoIs taken as input to the model).

Table 4.2: Performance comparison on the datasets evaluated by Recall@N and Precision@N. Best scores are in bold.

Dataset	Model	@1		@2		@3	
		R	P	R	P	R	P
GeoLife	MC	58.46	58.46	66.91	35.78	86.8	27.64
	n-MMC	58.9	58.9	78.7	39.3	87	29
	HPHD	58.07	58.07	73.23	39.8	87.41	29.7
	AR	59.32	59.32	67.43	38.9	86.7	28.78
	NN	58.1	58.1	82.1	40.9	91.1	30.3
	RNN	67	67	86.5	42.4	92.7	30.8
	STF-RNN	**73.3**	**73.3**	**87.7**	**44**	**93.1**	**31.1**
Gowalla	MC	15.3	15.3	21.1	11.04	29.6	9.34
	n-MMC	17.1	17.1	23.64	11.82	30.05	10.02
	HPHD	22.37	22.37	36.99	19.71	39.82	12.48
	AR	21.23	21.23	36.29	18.13	38.65	11.6
	NN	34.13	34.13	48.44	24.12	55.33	18.56
	RNN	34.32	34.32	48.68	24.34	55.67	18.56
	STF-RNN	**39.68**	**39.68**	**52.34**	**25.17**	**60.21**	**19.4**

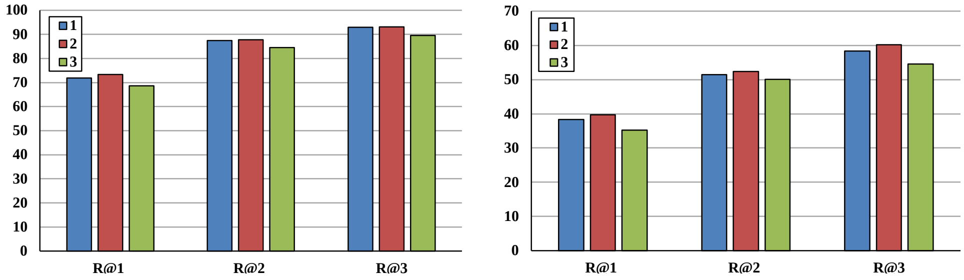
4.4.2 Results and Analysis

Table 4.2 illustrates the comparison between the prediction models on the two datasets in terms of R@N and P@N with $N = 1, 2$ and 3. Since the models' performances are consistent for different values of N, most of the representative results are shown when N=1. When N=1, Recall and Precision have the same values because the total number of real visited RoIs equal to the total number of predictions. We observe that the models perform better on GeoLife than Gowalla. The reason lies in that some user in Gowalla dataset has small mobility data size.

It is clear from the table that MC, *n*-MMC, HPHD, AR and NN models have achieved approximately similar results on GeoLife dataset when N=1. However, NN outperforms them when N=2 and 3 on both datasets under all evaluation metrics. Movement time information is not considered in these models, thus, they can not perform well.

RNN outperforms the compared ones on GeoLife dataset and slightly improves the results in comparison with NN on Gowalla dataset. This can be attributed to incorporating the recurrent structure of RNN which enables the model to get more accurate results by taking into account the historical dependencies.

In addition, it can be observed form the table that our proposal achieves the best



(a) GeoLife dataset.

(b) Gowalla dataset.

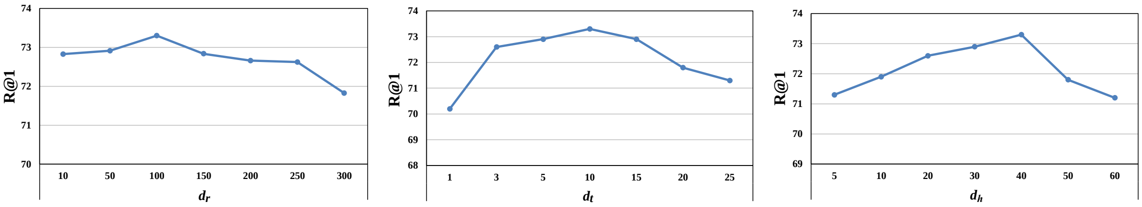
Figure 4.2: Performance of STF-RNN with varying window size w (1, 2 and 3).

results among all models under all settings of the evaluation metrics. For instance, on GeoLife, STF-RNN outperforms HPHD, NN and RNN by up to 26.23%, 26.16% and 9.4%, respectively, in term of P@1. The superiority of STF-RNN over the models can be attributed to incorporating the recurrent structure and using the time feature in the operations which impact positively the efficiency of the model. Moreover, the embedding learning technique of the input features enables the model to extract the hidden semantic information about the users' behaviour more efficiently. In all experiments, the highest performance is achieved under R@3 which indicate that the longer ranking list, the best accuracy can be obtained.

4.4.3 Effects of Parameters

In this section, we explore the effects of the parameters on the prediction model performance. Table 4.2 demonstrates how varying window size may affect the performance of STF-RNN on GeoLife dataset. The best performance is obtained with a window size value of 2 under all metrics on the two datasets. This is similar to the case of second order in MC which achieved the best performance as shown in Gambs et al. (2012); Huang et al. (2013).

In STF-RNN, the parameters d_r, d_t and d_h are responsible for determining the dimensionality of the embedded vectors of the RoI, time and hidden layer respectively so they play an important role in the efficiency of the model. To investigate the impact of these parameters and to select the best settings of them, we conduct several experiments to check the performance of STF-RNN with various dimensionalities as



(a) Impact of parameter d_r. (b) Impact of parameter d_t. (c) Impact of parameter d_h.

Figure 4.3: Parameters impact on STF-RNN performance.

shown in Figure 4.3. We only show R@1 on GeoLife dataset for ease of presentation. We start by varying the value of one parameter while fixing the others and then studying how the model efficiency is affected. The same procedure is then repeated for the rest of the parameters.

The impact of d_r parameter on R@1 of the model is shown in Figure 4.3a. We consider various space embedded vector dimensionality: 10, 50, 100, 150, 200, 250 and 300. In this figure, we exclude Gowalla dataset because of the low results in comparison with GeoLife dataset. We observed that the performance is improved with the increase of d_r and then it decreases slightly in terms of R@1. The best performance reaches its peak when $d_r = 100$. The smaller the values of d_r means that less RoI information is provided to the model limiting its efficiency in discovering dependencies and as consequence impairing its performance. When d_r value is large (*e.g.*, greater than 100), more noisy information has to be considered by the model which leads to poor performance.

The effect of d_t and d_h parameters on the model performance is depicted in Figure 4.3b and 4.3c respectively. We consider various time embedded vector dimensionality: 1, 3, 5, 10, 15, 20 and 25. In case of recurrent vector dimensionality, 5, 10, 20, 30, 40, 50 and 60 are considered. As shown in the figures, the best performance of STF-RNN is obtained with d_t value of 10 while it reaches its peak under the d_h parameter value of 40. Here, the fact that the model achieves its best performance with a small d_t value gives the impression that only a little time information is needed in representing the model dependencies in contrast with the RoI features where much more information is needed. The results confirm that the previous parameters play an important role in building an accurate RoI prediction model based on RNN.

4.5 Summary

In this chapter, STF-RNN is proposed to predict the future state of people movement. An embedding learning layer is used to effectively discover adequate internal representations of space and time input features enabling the model to capture the embedded semantic information about the users's behaviour more effectively. The recurrent structure is incorporated with space and time interval sequences in order to discover long-term dependencies which increases the efficiency of the proposed model. A performance evaluation is conducted on two large real life mobility datasets (GeoLife and Gowalla) showing that our model has improved the prediction effectiveness in comparison with the state-of-the-art models.

CHAPTER 5

The Effect of Different Architectural Configurations in Location Prediction Model

5.1 Introduction

Models in the literature have achieved satisfactory results, but they lack exploiting of timestamps-sensitive property while combining it with locations sequences. Whereas people's movement behaviours change according to the time (such as going to the workplace, restaurant or home), the effect of time interval becomes important for the prediction in such situations. Timestamp in mobility data contains different information: explicit (such as year, month, week and hour), implicit such as weekday type (weekday or weekend), and timeslots (morning, afternoon and night etc.) The main questions are: Which of these information should be considered when building the prediction model? Should all timestamp information be included to the model for the purpose of enhancing the model performance? To answer these questions, we adapt the time indexing scheme (Zhao et al., 2016) where timestamps are encoded into a particular time identifier.

Other prediction models ignore an important fact that values and representations of some variables can be much more relevant to the final location prediction than the rest of variables. Some studies considered only spatial context as the main factor to predict people's next location, missing other information of trajectories which is important to build accurate prediction models. Recently, several studies have been proposed to enhance the performance of next location prediction models by considering both spatial and temporal contexts (Al-Molegi et al., 2016; Liu et al., 2016). Studies showed that using temporal factors can significantly improve the performance of location prediction models. In Gatmir-Motahari et al. (2013), the authors demonstrated that time factor can significantly impact randomness, size, and probability distribution of people's movements.

As mentioned in the previous chapter, the models that use a variety of hand-crafted features lack the capability of capturing semantic information from people's mobility data. Therefore, embedding representation learning technique has been utilized for automatically learning the best internal representations of input data features. Moreover, in order to extract rich features, neural pooling functions are explored. Neural pooling functions (Collobert et al., 2011) are commonly used to extract meaningful features automatically from contexts according to each type of embeddings.

The purpose of this chapter is to study users' future locations' prediction performance by modifying our STF-RNN model described in Chapter 4. We report the results of a large number of experiments exploring different configurations of STF-RNN model. The main contributions of this chapter are as follows: First, we propose a model based on RNN called *Space-Time Pooling-based Architecture*

(ST-PA), where embedding representation technique and neural pooling functions are used to extract rich features of the context data. Second, we study the impact of using different data as inputs to the model (those include times of entering and leaving a RoI, day of the week and other timestamps information). We propose time encoding scheme to provide the model with more information related to the movement time. Third, we investigate the impact of using different input feature representation techniques on the prediction performance. Lastly, an extensive set of experiments is conducted on two large real life mobility datasets in order to evaluate the efficiency of the developed models.

5.2 Prediction Models Description

The STF-RNN architecture we considered requires specifying: the number and type of data inputs, input features representation and the dimensionality size of the hyper-parameters. In this chapter, we conduct a sensitivity analysis of STF-RNN to explore the effect of modifying the architectural components on model performance. We investigate the effect of using three factors: (i) pooling-based architecture (Section 5.2.1), (ii) different data inputs (Section 5.2.2) and (iii) different input features representation techniques (Section 5.2.3).

5.2.1 Pooling-based Architecture

In this section, we address the description of the ST-PA model for location prediction. To begin with, we describe the formulation of ST-PA model. Then, we demonstrate the training procedure.

The task needs to learn a probability function for a given user u to a next location at time t given his/her previously visited locations.

As mentioned in Chapter 3.2.2, a user trajectories are represented as a sequence of movements $\mathcal{M}^u = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, where $u \in \mathcal{U}$ and n is the length of the user's trajectories. Figure 5.1 illustrates the neural network architecture. We denote the vector of RoI identifiers by $r_i \in \mathbb{R}^N$ where N is the number of user's RoI, while

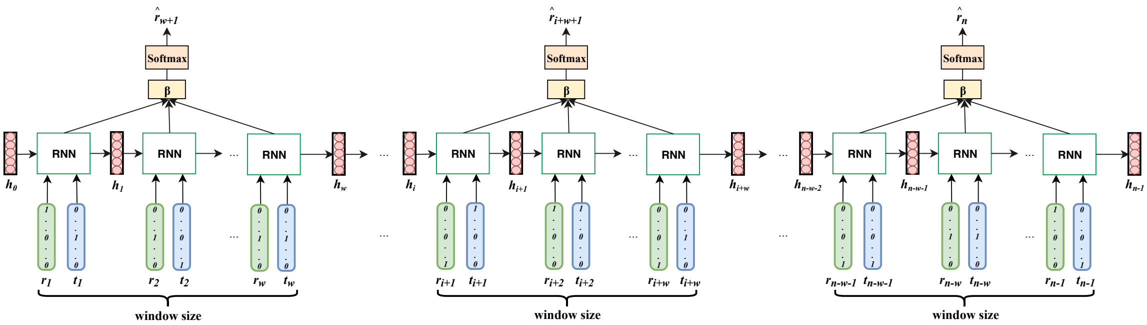


Figure 5.1: ST-PA architecture.

$t_i \in \mathbb{R}^M$ and M is the number of different time intervals.

The embedded layer between the input and the hidden layers is used to learn a meaningful representation of the RoI and the time identifier features. Let $Re \in \mathbb{R}^{N \times d_r}$ be the embedding matrix that represents a set of RoIs, where d_r is the dimensionality of the embedded vector of the RoI. The embedding matrix $Te \in \mathbb{R}^{M \times d_t}$ represents a set of time identifiers where d_t is the dimensionality of the embedded vector of the time.

At each time step i, RNN takes the input vectors $re_i \in \mathbb{R}^{d_r}$, $te_i \in \mathbb{R}^{d_t}$ and the previous hidden state $h_{i-1} \in \mathbb{R}^{d_h}$ and outputs the next hidden state $h_i \in \mathbb{R}^{d_h}$ by applying the following equation:

$$h_i = f\left(re_i \cdot W_r + te_i \cdot W_t + h_{i-1} \cdot W_{h_{i-1}} + b_h\right) \tag{5.1}$$

where $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_t \in \mathbb{R}^{d_t \times d_h}$, $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ and $b_h \in \mathbb{R}^{d_h}$ are the parameters of the RNN model and f is an element-wise non-linearity transformation function. The embedded vectors re_i and te_i come from the embedding matrices of the RoI Re and time identifiers Te, respectively. Hyperbolic tangent is used as the non-linear activation function for the hidden layer.

The value of the prediction layer is computed as below:

$$\hat{y} = g(\beta \cdot W_\beta + b_o) \tag{5.2}$$

where $W_\beta \in \mathbb{R}^{p * d_h \times N}$ represents the weight matrix between the hidden and output layers, $b_o \in \mathbb{R}^N$ is the output layer bias and $\beta \in \mathbb{R}^{p * d_h}$ is a fixed vector which contains a high level features summarized the historical movement where $p \in \{1, 2, 3\}$ is the number of pooling function. *Softmax* is used as the non-linear activation function.

To compute the vector β, we use a RNN following by a pooling functions. RNN has been proved to be very useful in representing such sequential data, while pooling functions have shown being highly effective in extracting high level features from dense real-valued vectors.

Given a sequence of historical movements $[\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_w]$, where w is the length of the sequence, the RNN model produces a corresponding sequence of hidden states $[h_1, h_2, ..., h_w]$. Then, we pass it through a pooling layer to obtain a fixed vector that contains high level features. The following equation illustrates this idea:

$$\beta = \bigcup_{pool \in P} pool_{i=1}^{d_h} h_i \tag{5.3}$$

where $\bigcup$ denotes the concatenation operation and $P \in \{sum, dot, max\}^3$ is the set of pooling functions. The pooling function is an element-wise function and it converts the sequence of vectors into a fixed length vector.

BPTT algorithm is used as a learning process of the model. ADADELTA update rule is employed to estimate the model parameters, $\theta = [Re, Te, W_r, W_t, W_{h_{i-1}}, W_\beta, b_h, b_o, r_0]$, where r_0 is the initial vector for the recurrent layer. This process is repeated iteratively until reaching the convergence state.

5.2.2 Different Data Inputs

In this section, we conduct a sensitivity analysis of STF-RNN to explore the effect of using different data inputs on model performance. Different data inputs are used such as the time of entering a RoI, weekday types. Moreover, time encoding scheme is used to provide the model with more information related to the activity time.

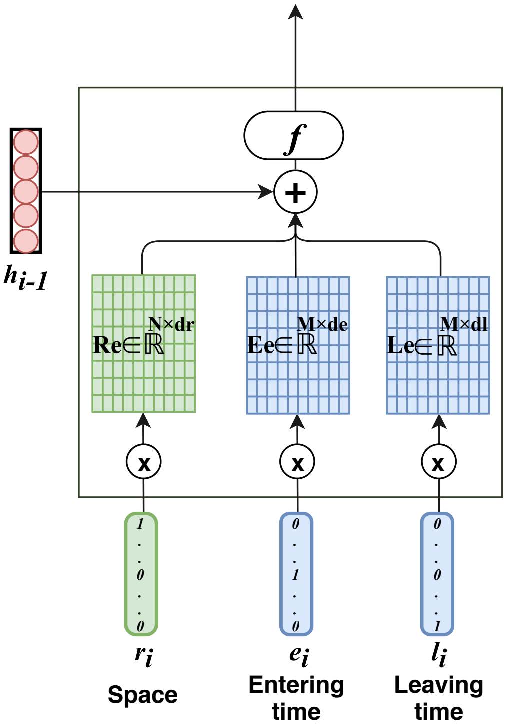


Figure 5.2: STE-RNN cell.

5.2.2.1 Time of Entering RoI Information

In this section, the time of entering a RoI is included as input to STF-RNN. The purpose is to discover if there is a relation (*i.e.* dependence) between the leaving time of the current RoI and the entering time of the next one. The movement trajectory $\mathcal{M}$ is represented as a sequence of tuples (r_i, e_i, l_i) where r is the RoI identifier, l is the leaving time from the RoI and e is the entering time to the RoI. r and l are the same data inputs of STF-RNN (we change the symbol t into l to indicate that this is the leaving time). This model is called Space-Time of Entering-RNN model (STE-RNN).

The cell unit, shown in Figure 5.2, is a slight variant of STF-RNN cell. The input layer consists of three vectors. The first and second ones are similar to the input vectors of STF-RNN. The third vector represents the time unit part in hours of entering a RoI at time i. We denote this vector by $e_i \in \mathbb{R}^M$ and M is the number of different time intervals. The values of the recurrent layer are computed as it follows:

$$h_i = f\Bigg(re_i \cdot W_r + ee_i \cdot W_e + le_i \cdot W_l + h_{i-1} \cdot W_{h_{i-1}} + b_h\Bigg) \tag{5.4}$$

where $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_e \in \mathbb{R}^{d_e \times d_h}$, $W_l \in \mathbb{R}^{d_l \times d_h}$, $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ and $b_h \in \mathbb{R}^{d_h}$ are the weight matrices. The embedded vector $ee_i \in \mathbb{R}^{d_e}$ is given by multiplying the embedded matrix $Ee \in \mathbb{R}^{M \times d_e}$ and the input vector e_i. Ee represents a set of entered times and d_e is the dimensionality of the embedded vector of the entered times. Thus, STF-RNN-II parameters are $\theta = [Re, Ee, Le, W_r, W_e, W_l, W_{h_{i-1}}, W_h, b_h, b_o, h_0]$.

5.2.2.2 Weekday Types Information

Here, we explore the effect of including weekday types information (*i.e.* weekday and weekend) on the model performance. The movement trajectory $\mathcal{M}$ is represented as a sequences of tuples (r_i, l_i, q_i) where r and l are the same input features of STF-RNN and q is the weekday information of the movement. This model is called Space-Time-Week-RNN model (STW-RNN).

The cell unit of STW-RNN is similar to STE-RNN cell which the input layer consists of three vectors. The input layer consists of three vectors. The first and second ones are similar to the input vectors of STF-RNN, while the third vector represents the weekday type information. We denote this vector by: $q_i \in \mathbb{R}^J$ where $J \in \{2, 7\}$, 2 is the two types of weekday (*i.e.*, weekday (WD) and weekend (WE)), and 7 is the seven days in the week (AllWD).

In STW-RNN, the values of the recurrent layer are computed as it follows:

$$h_i = f\Bigg(re_i \cdot W_r + le_i \cdot W_l + qe_i \cdot W_q + h_{i-1} \cdot W_{h_{i-1}} + b_h\Bigg) \tag{5.5}$$

where $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_l \in \mathbb{R}^{d_l \times d_h}$, $W_q \in \mathbb{R}^{d_q \times d_h}$ and $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ are the weight matrices and $b_h \in \mathbb{R}^{d_h}$ is the hidden layer bias. The embedded vector $qe_i \in \mathbb{R}^{d_q}$ is given by multiplying the input vector q_i and the embedded matrix $Qe \in \mathbb{R}^{J \times d_q}$. Qe represents a set of weekday type and d_q is the dimensionality

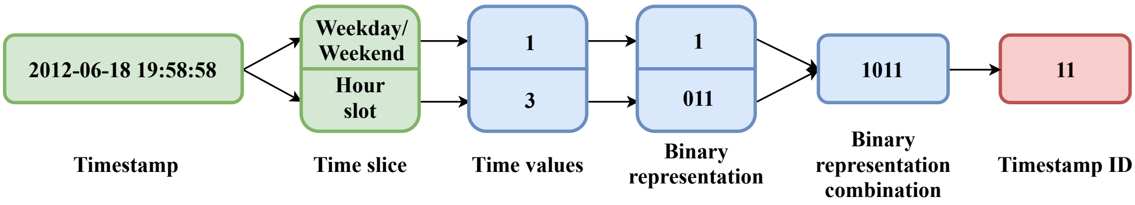


Figure 5.3: Time encoding scheme.

of the embedded vector of the weekday type. Thus, STF-RNN-III parameters are $\theta = [Re, Le, Qe, W_r, W_l, W_q, W_{h_{i-1}}, W_h, b_h, b_o, h_0]$.

5.2.2.3 Time Encoding Scheme Information

In order to capture the temporal movement behaviour characteristics, we adapt the time encoding scheme proposed by Zhao et al. (2016) where timestamps are encoded into a particular time identifier. First, a timestamp is divided into two bins in terms of weekday type (weekday and weekend) and day hour slot. Following the work introduced by Liu (2018), the day hours are discretized into the following five timeslots: ‘Morning’, ‘Noon’, ‘Afternoon’, ‘Evening’, and ‘Night’. ‘Morning’ is the time from 6:00 to 10:59, ‘Afternoon’ is from 11:00 to 13:59, ‘Evening’ is from 14:00 to 17:59, ‘Evening’ is from from 18:00 to 21:59 and ‘Night’ is from 22:00 to 5:59. In addition, 1 bit is used to denote weekday or weekend (0 value for weekend days and 1 otherwise). 3 bits encoding is used for the five different hour slots, 000 for ‘Morning’, 001 for ‘Noon’, 010 for ‘Afternoon’, 011 for ‘Evening’ and 100 for ‘Night’. Finally, the binary code is converted into a unique decimal digit as the time identifier, where the identifier is in the range of: 0 to 12. Figure 5.3 demonstrates the procedure of encoding an exemplary time stamp, “2012-06-18, 19:58:58”. In this scenario, the time identifiers numbers will be reduced to 13 instead of 95 in the original paper (Zhao et al., 2016) which means less number of parameters and fast model training. Different time encoding schemes are evaluated in Section 5.3.3.

The movement trajectory $\mathcal{M}$ is represented as a sequences of tuple (r_i, x_i) where r is the RoIs sequences and $x \in \mathbb{R}^{13}$ represents the time identifiers vector and 13 is the number of different time identifiers. This model is called Space-Time Encoding Scheme-RNN model (STES-RNN). The recurrent cell is similar to STF-RNN cell whose input layer consists of two vectors. The values of the recurrent layer are computed as follows:

$$h_i = f\left(re_i \cdot W_r + xe_i \cdot W_x + h_{i-1} \cdot W_{h_{i-1}} + b_h \right) \tag{5.6}$$

where the embedded vectors re_i and xe_i come from the embedding matrices of the RoIs Re and time encoding scheme identifiers Xe, respectively. $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_x \in \mathbb{R}^{13 \times d_h}$, $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ and $b_h \in \mathbb{R}^{d_h}$ are the model's parameters. Thus, STF-RNN-IV parameters are: $\theta = [Re, Xe, W_r, W_x, W_{h_{i-1}}, W_h, b_h, b_o, h_0]$.

5.2.3 Different Data Input Representation Techniques

An important property of location prediction model is to learn a meaningful representation of mobility data which enables the model to capture the embedded semantic information about people's behaviour. Learning input features representation offers further gains in performance. Therefore, we explore the impact of STF-RNN with respect to the input representation that is used. Specifically, we use one-hot encoding for some data input instead of using the embedding learning representation. For example, we keep fixed representations of space data input using one-hot vector and learn the embedded representation of time input and vice versa. Based on the representation learning of space and time inputs, we employ the following variations:

- Without Representation Learning (**Wo-RL**): all input features are randomly initialized and kept static.
- Space Representation Learning (**S-RL**): all input features are randomly initialized and then only space input feature is modified during training.
- Time Representation Learning (**T-RL**): all input features are randomly initialized and then only time input feature is modified during training.
- Space-Time Representation Learning (**ST-RL**) is our original model, STF-RNN, where all input features are randomly initialized and then modified during the training process.

Table 5.1: Performance comparison on the datasets evaluated by Recall@N and Precision@N. Best scores are in bold.

Dataset	Model		@1		@2		@3	
			R	P	R	P	R	P
GeoLife	STF-RNN		73.3	73.3	87.7	44	93.1	31.1
	ST-PA	sum	71.87	71.87	86.09	43.04	92.07	30.68
		dot	72.12	72.12	86.91	43.46	92.43	30.79
		max	73.17	73.17	87.47	43.84	92.45	30.8
		max + sum	72.51	72.51	87.12	43.55	92.56	30.84
		sum + dot	72.48	72.48	86.96	43.48	92.53	30.83
		max + dot	72.82	72.82	87.41	43.70	92.63	30.86
		max + sum + dot	**73.58**	**73.58**	**87.82**	**44.41**	**93.25**	**31.45**
Gowalla	STF-RNN		39.68	39.68	52.34	25.17	60.21	19.4
	ST-PA	sum	39.08	39.08	53.03	26.51	60.03	20.01
		dot	39.2	39.2	53.3	26.65	60.25	20.08
		max	**40.99**	**40.99**	**53.38**	**26.69**	**60.35**	**20.12**
		max + sum	39.14	39.14	53.17	26.59	60.08	20.03
		sum + dot	39.09	39.09	53.24	26.62	60.11	20.04
		max + dot	39.18	39.18	53.22	26.61	60.1	20.03
		max + sum + dot	38.02	38.02	53.24	26.62	60.31	20.1

5.3 Experiments and Results

In this section, large-scale experiments are conducted to explore the effect of modifying the architectural components on model performance. The settings of the conducted experiments including the datasets used for the evaluation are described. Finally, experimental results are reported.

5.3.1 Experimental Settings

We use two publicly available real-world datasets in our experiments: GeoLife (Zheng et al., 2010) and Gowalla (Cho et al., 2011). Recall and Precision are employed as our evaluation metrics in all experiments to assess the efficiency of the prediction models. The model of each user is trained on its own mobility data using three-fold cross validation technique. The Recall and Precision scores of each case from each user is then calculated and the final results of all users are averaged.

5.3.2 Effect of Pooling-based Architecture

Table 5.1 shows the R@N and P@N values on the two datasets with different values of N (*i.e.* 1, 2 and 3). In terms of the performance metrics, ST-PA model outperforms

STF-RNN prediction model on the two datasets. On the GeoLife dataset, the best performance is achieved by using a combined *sum*, *max* and *dot* pooling. Furthermore, results showed that using *max* pooling is particularly appropriate to the separation of features that are very sparse (*i.e.*, have a very low occurrence probability). This may explain why maximum pooling performs better than other features.

The advantage of ST-PA can be attributed to the fact that pooling functions achieve invariance to feature transformations, more compact representations and better robustness to noise and clutter. ST-PA outperforms STF-RNN by 0.38% in terms of R@1. Using the combination function (*sum*+*max*+*dot*), the model achieved better performance with 73.58%.

On the Gowalla dataset, using *max* pooling alone outperforms other features, with 40.99% in terms of R@1. ST-PA outperforms STF-RNN by 3.3% in terms of R@1. By using a combination of the pooling functions, the model shows approximately similar results.

5.3.3 Effect of Different Data Inputs

In this section, we begin with investigating the effect of including the time of entering a RoI. We also experiment the effect of adding an information that contains the activity weekday types (*i.e.* weekday and weekend). Finally, time encoding scheme is used to provide the model with more information related to the activity time. Different time interval types are evaluated as follows: Month Week DayHours (MWDH), Month Week 5TimeslotsDay (MW5TD), Week DayHours (WDH) and Week 5TimeslotsDay (W5TD).

The performance comparison on the two datasets in terms of R@N and P@N is illustrated in Table 5.2. It can be observed form the table that STF-RNN outperforms STE-RNN. Unlike STE-RNN, STF-RNN uses the features: users' leaving time information and RoIs sequences. Those two features were the reason for such improvement. In other words, the next RoI users will visit depends on the current RoI and the time at which they leave without the need to know the entry

Table 5.2: Performance comparison on the datasets evaluated by Recall@N and Precision@N. Best scores are in bold.

Dataset	Model	@1		@2		@3	
		R	P	R	P	R	P
GeoLife	STF-RNN	73.3	73.3	87.7	44	93.1	31.1
	STE-RNN	73.13	73.13	87.53	43.77	93	31.03
	STW-RNN + WD/WE	**73.9**	**73.9**	88.51	44.26	93.65	31.21
	STW-RNN + AllWD	73.57	73.57	88.43	44.22	93.55	31.17
	STES-RNN + MWDH	72.34	72.34	87.87	43.94	93.49	31.15
	STES-RNN + MW5TD	72.87	72.87	88.29	44.15	93.62	31.2
	STES-RNN + WDH	73.14	73.14	88.46	44.23	93.66	31.21
	STES-RNN + W5TD	73.55	73.55	**88.6**	**44.29**	**93.7**	**31.23**
Gowalla	STF-RNN	39.68	39.68	52.34	25.17	60.21	19.4
	STE-RNN	37.92	37.92	52.24	24.86	60.19	17.79
	STW-RNN + WD/WE	**40.58**	**40.58**	**53.79**	**26.89**	**60.47**	**20.16**
	STW-RNN + AllWD	39.99	39.99	53.61	26.81	60.33	20.11
	STES-RNN + MWDH	39.05	39.05	53.4	26.75	60.21	20.07
	STES-RNN + MW5TD	40.04	40.04	53.5	26.75	60.26	20.09
	STES-RNN + WDH	40.36	40.36	53.54	26.77	60.26	20.09
	STES-RNN + W5TD	40.56	40.56	53.7	26.85	60.45	20.15

time. In Gowalla dataset, there is a significant difference between the two models. We think that the reason for such difference is that the timestamp information in Gowalla dataset cannot be expressed in terms of the time of entering and leaving a RoI. Not only STE-RNN did not perform well, but was also less efficient as more features were included in comparison with STF-RNN. As we mentioned earlier, those experiments showed also that users' leaving time of the RoI is more relevant to the prediction process than their entry time.

Regarding to STW-RNN, in terms of R@1, STW-RNN+WD/WE outperforms STF-RNN in both evaluated datasets with 0.82% and 2.26%. Using the variable 'weekday types' is shown to be very useful and such variable is a significant predictor of users' next location. It shows that users follow similar patterns in the weekdays in comparison to the weekends.

Finally, the results showed that in time encoding scheme, including W5TD is significantly better than others. This is verifies the effectiveness of using time encoding scheme and as well as the validity of incorporating various temporal characteristics. This helps the model to effectively capture new temporal effects and, as consequence, impacts positively the efficiency of the model.

Results showed also that with time related features, using small time periods (*e.g.*, hours of the day) is more predictive features than larger time features such as the month of the year. One interesting observation from the reported results is that using the month in the encoding scheme does not show improvement in the performance. This can be attributed to the similarity of user's behaviours along the months which means month information is not discriminative feature. Users may follow similar behavioural patterns across several months (when excluding vacation months). However, splitting the day hour into five timeslots shows significant improvement. This is not surprising as the people's daily activities usually occur in slots such as morning, afternoon, etc. For example, users may go to a restaurant after leaving the workplace at noon, while they may be more likely to go to a gym when they leave office at night.

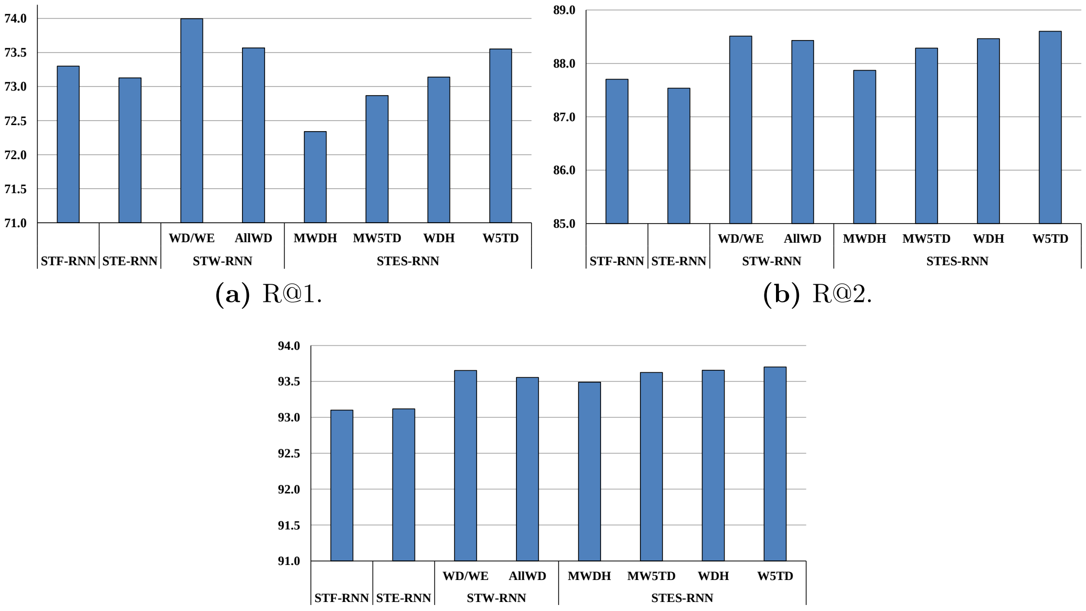
All in all, the selected features in each model is what significantly impact each model overall prediction performance, Figures 5.4 and 5.5. The best prediction results are obtained when STW-RNN (with WD/WE) is used on GeoLife and Gowalla datasets in terms of R@1. This validates our strategy of incorporating time information which exhibits explicit differences of users' behaviours.

While the addition of some features can improve model prediction performance, the addition of some other features can worsen some other models. This is the case with STES-RNN model with MWDH feature. W5TD and time encoding scheme features improve the performance in most models, best in STES-RNN (using R@2 and 3 performance metrics). STE-RNN achieves the worst performance on both datasets.

5.3.4 Effect of Different Data Input Representations

The goal of these comparisons is to show how representation learning of data inputs can affect the prediction performance. We report results achieved using different space and time representations in Table 5.3.

Results showed that SRL feature is a significant predictor. More specifically,

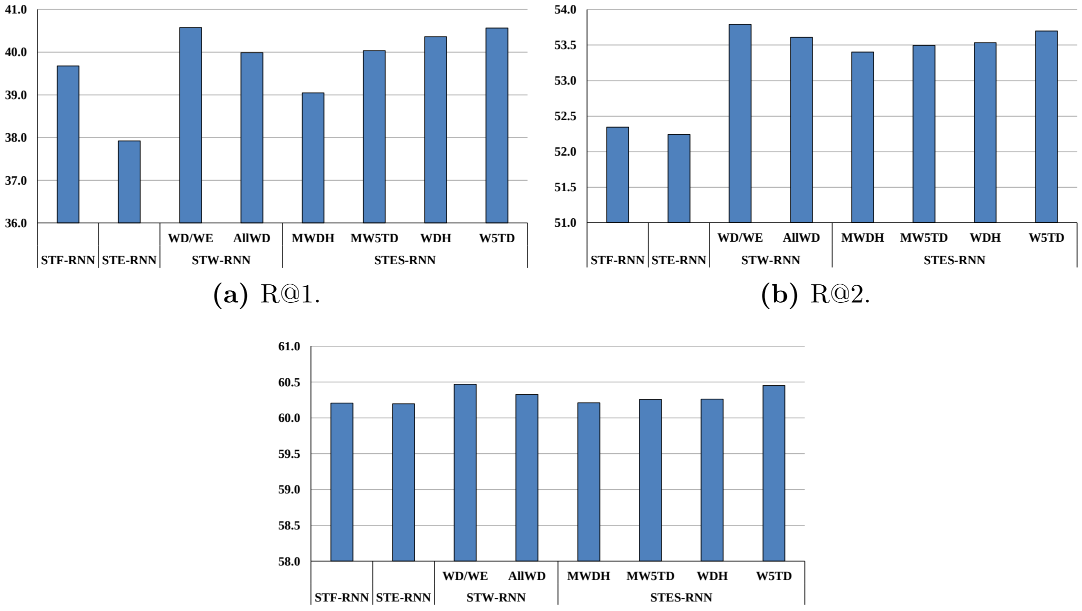


(a) R@1.

(b) R@2.

(c) R@3.

Figure 5.4: Different data input models comparison using GeoLife dataset.



(a) R@1.

(b) R@2.

(c) R@3.

Figure 5.5: Different data input models comparison using Gowalla dataset.

Table 5.3: Performance comparison of STF-RNN with different input representations. Best scores are in bold.

Dataset	Model	@1		@2		@3	
		R	P	R	P	R	P
GeoLife	STRL (STF-RNN)	73.3	73.3	87.7	44	93.1	31.1
	SRL	**73.61**	**73.61**	**88.23**	**44.11**	**93.42**	**31.13**
	TRL	72.57	72.57	87.46	43.73	92.86	30.94
	WoRL	72.55	72.55	86.96	43.48	92.62	30.86
Gowalla	STRL (STF-RNN)	39.68	39.68	52.34	25.17	60.21	19.4
	SRL	**41.31**	**41.31**	**55.39**	**25.94**	**63.4**	**19.95**
	TRL	39.16	39.16	52.11	24.79	60.93	18.63
	WoRL	39.09	39.09	52.13	24.81	60.76	18.57

in terms of R@1 on GeoLife dataset, SRL outperforms STRL, TRL and WoRL by up to 0.42%, 1.4% and 1.5% respectively. On Gowalla dataset, SRL outperforms STRL, TRL and WoRL by up to 4.1%, 5.5% and 5.8% respectively. This means that significant information needed for building the prediction model is included in space input data (*i.e.* RoI sequences) in contrast with time input data where only a little information is needed in representing the model dependencies. WoRL and TRL models have achieved approximately similar results in both datasets. The reason is that models learn from time data input (in case of TRL), without adequate data of space input. As a result, such features do not capture sufficient discriminative information for RoI prediction.

5.4 Summary

In this chapter, we have focused on evaluating models to predict the future location of smartphone users. We described a series of experiments to extend our previous prediction model (STF-RNN). We proposed time encoding scheme to encode timestamps into particular time identifiers. A set of neural pooling functions are explored in order to extract rich features. We evaluated the use of several input variables and their impact on accurately predicting users' next locations. Different input representation methods (*i.e.* embedding learning and one-hot vector), for those input variables are investigated. We showed when and where each method can show better results. We concluded here by summarising our main findings as follows:

- Multiple pooling functions offers rich sources of feature information, which leads to an improvement on the prediction performance.
- The number of input features can play an important role in the prediction performance (given the selection of proper and relevant features). However, increasing the number of input features will increase the training time of the model (*i.e.* overhead, efficiency, etc.). LBSs services are very sensitive to speed and efficiency and we cannot trade-off those to improve prediction. Ideally, we want models to achieve the best in both (*i.e.* prediction accuracy and speed or efficiency).
- Using the leaving time input variable/feature only performs well in comparison with using both entering and leaving RoIs (*i.e.* together).
- Using weekday types information is shown to be such a good input feature in improving the prediction performance (in all evaluated models).
- Time encoding scheme is useful to provide the prediction model with more information related to the movement behaviour characteristics. The results indicate that considering different timestamp information is always beneficial. However, as we mentioned before, we need to balance this with efficiency.
- Learning input feature representation can have a significant positive impact on performance, and should be investigated. The best results are obtained when learning the representation of space data input only.
- Space embedded vector size has a relatively little effect on the model performance. For the model with space learning, it may be worth using a large space embedded vector size where more location information is provided to the model.

CHAPTER 6

An Attention-based Neural Model for People's Movement Prediction

6.1 Introduction

Several proposals on predicting people's next location can be found in the literature, which are based on well-known techniques, *e.g.*, MC, Association Rules, NN and RNN. Notwithstanding, the appropriate combination between sequences of locations (*i.e.* a specific area that can be geolocalized) and movement times (*i.e.* morning, afternoon, from 4 to 6 PM, etc.) is still relatively unexplored. A location represents those spatial regions where the user has stayed for more than a pre-specified time threshold (*e.g.*, 30 minutes), providing that the diameter of the region does not exceed a predefined threshold (*e.g.*, 20 meters).

In this work, we aim to improve the prediction of a person's next location through effectively capturing the temporal effects by learning an attention model over his/her location history. In other words, we seek to answer the question: Which time interval do people attend to, while moving from one location to another? Typically, proposals squash all the input data into a single fixed-length vector. Our hypothesis is that the movement predictions learned by our model enable to effectively combine the spatial and temporal information.

The spatial and temporal information have different degrees of importance when generating the information for location prediction. Specifically, the temporal information can contribute to add more weights to spatial information. Our proposal, based on the promising attention technique, learns the importance of temporal information that should be more involved when computing the prediction outputs.

The *attention technique* has been applied on RNN to find the relevant part of the information that helps generate the outputs. The combination of RNN and attention techniques improves the performance of many challenging tasks such as machine translation (Luong et al., 2015), generation of image captions (Xu et al., 2015), video description (Hori et al., 2017) and speech recognition (Chorowski et al., 2014).

In the area of predicting people's next location, various models have been proposed, including LZ algorithm, Markov Model, Bayesian Networks and Association rules. Another commonly used models for predicting users' next movement based on the past or historical mobility is NN (Leca et al., 2015; L. Vintan and Ungerer, 2004) and RNN (H. Kaaniche, 2010). In addition, a new model has been proposed in (Lee et al., 2016). The authors introduced the notion of Spatio-Temporal-Periodic (STP) patterns to represent user's past visits and then used the proposed model to extract these patterns for predicting next places. Liu et al. (2016) proposed a RNN model called ST-RNN, which considered the periodical context of location and/or time successions. Instead of squashing all the information into a single fixed-length vector, the model proposed in this chapter is able to weight the location information by the movement time.

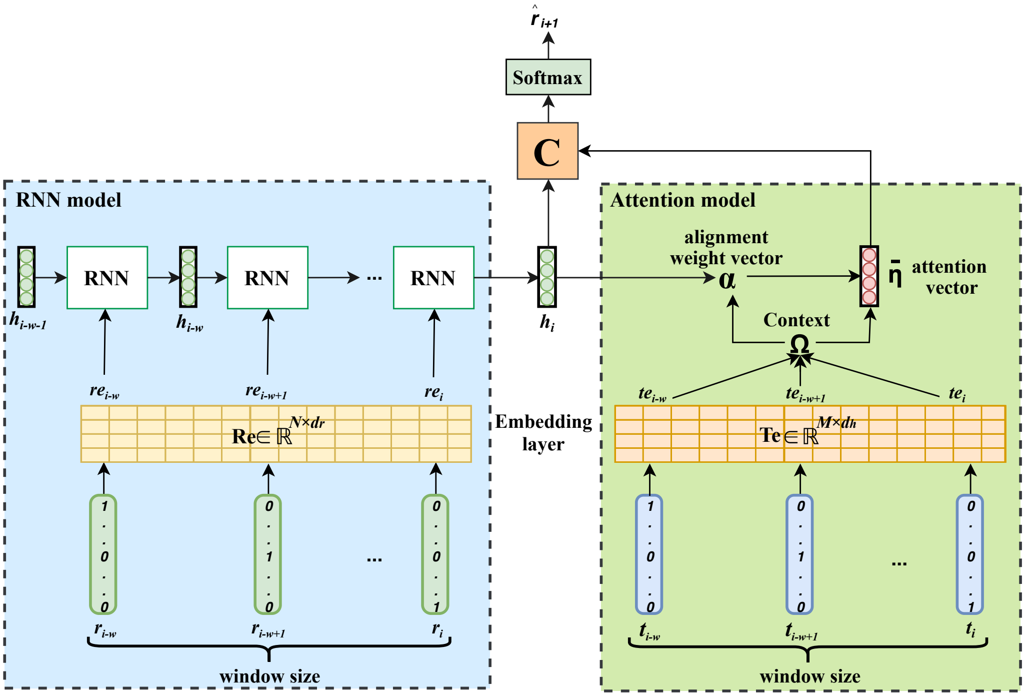


Figure 6.1: MAP architecture. The target hidden state h_i and the context vectors Ω are generated based on the spatial and temporal data, respectively. The model infers an alignment weight vector α based on the current target hidden state h_i and the context vectors Ω. The attention vector $\bar{\eta}$ is then computed using the alignment weight vector α and the context vectors Ω. To predict the next location, the target hidden state h_i is combined with the attention vector $\bar{\eta}$.

In this chapter, we propose MAP, a RNN architecture that improves location prediction by using an attention technique that provides the learning model with more information, related to the movement time. We conduct detailed experiments using two large real-life mobility datasets to evaluate the efficiency of the proposed model.

6.2 MAP: Model Description

In this section, we describe the MAP model, which accepts both RoIs and time intervals as inputs and predicts user's next location.

As stated previously, a user's trajectories are represented as a sequence of movements $\mathcal{M}^u = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, where $u \in \mathcal{U}$ and n is the length of the user's trajectories.

MAP consists of three major parts (see Algorithm 1): RNN (lines 3 to 8), the Attention Model (lines 9 to 16) and Softmax Classifier (lines 17 to 20). The RNN part reads through the input spatial sequences and generates a high level representation vector h that describes the historical movement summary, while the Attention Model reads the temporal sequences and generates an attention vector $\bar{\eta}$ to add more relevant information to the model outputs. Then, the attention and RNN vectors are used to generate the probability distributions for the next RoIs (*i.e.* the prediction itself).

$$h_i = RNN(r_i, \ldots, r_1) \tag{6.1}$$

$$\bar{\eta} = attention(h_i, [t_i, \ldots, t_1]) \tag{6.2}$$

$$P(\hat{r}_{i+1} = r_j | \mathcal{M}_i, \ldots, \mathcal{M}_1) = softmax(h_i, \bar{\eta}) \tag{6.3}$$

The overall structure of the MAP model is shown in Figure 9.1. We provide more details of these three parts in the following subsections.

6.2.1 RNN

The input of RNN part is a sequence of RoIs $[r_{i-w}, \ldots, r_i]$ which represents the spatial historical movements where w is the number of visited RoIs taken as input to the model. Each RoI is represented as a vector $r_i \in \mathbb{R}^N$ and N is the number of RoIs in $\mathcal{R}^u$. This vector is encoded using one-hot encoding $(1 - \text{of} - N)$. Each input vector r_i is passed through an embedding layer to produce a real-valued vector with d_r dimensionality in order to learn a meaningful representation of the RoI input feature.

Specifically, the RNN model takes the corresponding input vector r_i, produces a fixed-length embedded vector re_i which used as as input to RNN cell as follows:

Algorithm 1: MAP Model.

```
Input  : $Traj$: user's trajectories represented as a sequence of movements,
         each movement contains $(r, t)$, $r \in \mathcal{R}$ and $t \in \mathcal{T}$, $w$: number of
         visited RoIs.
Output: $S$: set of predicted RoIs.
$S \leftarrow []$
for $x \leftarrow 1$ to $Traj.length - w + 1$ do
    //RNN model
    initialize $h$ by zero.    //target hidden state
    for $i \leftarrow x$ to $x + w$ do
        $re \leftarrow$ embedding($Traj[i].r$, $Re$)
        $h \leftarrow$ RNN($re$, $h$)
    end
    //Attention model
    $\Omega \leftarrow []$           //context vectors
    for $j \leftarrow x$ to $x + w$ do
        $te \leftarrow$ embedding($Traj[j].t$, $Te$)
        $\Omega$.append($te$)
    end
    $\alpha = \text{softmax}(\Omega \cdot h)$   //alignment weight vector
    $\bar{\eta} = \alpha \cdot \Omega$           //attention vector
    //Classifier
    $\hat{y} = \text{softmax}(\alpha(h, \bar{\eta}))$ //alignment function
    $k \leftarrow \text{argmax}(\hat{y})$
    $S$.append($\mathcal{R}[k]$)
end
```

$$re_i = r_i \cdot Re \tag{6.4}$$

$$h_i = \tanh\Bigg(re_i \cdot W_r + h_{i-1} \cdot W_{h_{i-1}} + b_h \Bigg) \tag{6.5}$$

where $Re \in \mathbb{R}^{N \times d_r}$ is the embedded matrix that represents a set of RoIs and d_r is the RoI embedded vector dimensionality. $h_i \in \mathbb{R}^{d_h}$ is the hidden state where d_h is the hidden layer vector dimensionality. $W_r \in \mathbb{R}^{d_r \times d_h}$, $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ and $b_h \in \mathbb{R}^{d_h}$ are the parameters of the RNN model. Hyperbolic tangent (tanh) is used as the non-linear activation function.

In this model, the next RoI prediction depends on the current input RoI and on the sequential historical information as well. This property helps the model keep track of user movement history and discover meaningful dependencies to enhance the model performance.

6.2.2 Attention Model

The Attention Model part inputs are the RNN output h_i and a set of vectors $[t_{i-w}, \ldots, t_i]$ that represent the hour part of the leaving time from the RoI. In this case, the time intervals are the number of hours per day (*i.e.* 24). Each time vector is denoted by $t \in \mathbb{R}^M$ and encoded using $1-\text{of}-M$ encoding technique, where M is the number of different time intervals in $\mathcal{T}^u$. Each time vector t_i is passed through an embedding layer to produce a fixed-length embedded vector $te_i \in \mathbb{R}^{d_h}$ as follows:

$$te_i = t_i \cdot Te \tag{6.6}$$

where $Te \in \mathbb{R}^{M \times d_h}$ is the embedded matrix and d_h is the leaving time embedded vector dimensionality. Therefore, after reading w temporal inputs, we extract a set of vectors $[te_{i-w}, \ldots, te_i]$ which are referred to the context vectors Ω.

Given the target hidden state h_i and the context vectors Ω, we employ a neural network to learn jointly the relationship between the RoI and time information as follows. The model first infers an alignment weight vector α by multiplying the target hidden state h_i and the context vectors Ω. This inner product finds the similarities between them. Then, the output is normalized using a Softmax function.

$$\alpha = softmax(\Omega \cdot h_i) \tag{6.7}$$

To generate the attention vector $\bar{\eta}$, the alignment weight vector α is multiplied by the context matrix Ω as follows:

$$\bar{\eta} = \alpha \cdot \Omega \tag{6.8}$$

6.2.3 Softmax Classifier

The hidden state h_i and the attention vector $\bar{\eta}$ are used to produce a probability distribution over the RoIs $\hat{y}$.

$$\hat{y} = g(C(h_i, \bar{\eta}) \cdot W_C + b_o) \tag{6.9}$$

where $W_C \in \mathbb{R}^{d_h \times N}$ and $b_o \in \mathbb{R}^N$ are the model parameters. Here, C is referred as a content-based function for which we consider two different combination alternatives:

$$C(h_i, \bar{\eta}) = \begin{cases} h_i + \bar{\eta} & \text{sum.} \\ h_i | \bar{\eta} & \text{concat.} \end{cases} \tag{6.10}$$

The vector $\hat{y}$ is further normalized using Softmax to obtain the probability distribution over the RoIs.

The computation path goes from $h_i \rightarrow \alpha \rightarrow \bar{\eta} \rightarrow C$ then a prediction is made as detailed in Equations 6.5, 6.7, 6.8 and 6.9. The feature vectors of spatial and temporal data are different, as they capture different aspects of the trajectories. All spatial and temporal vectors are trained jointly. Such a joint training is capable of capturing the dependencies between the different trajectory's parts, and results in effectively predictions.

6.2.4 Learning Algorithm

Optimization is performed using ADADELTA update rule and BPTT algorithm. The RNN and attention models are trained jointly. The cost J to be minimized by optimizing the model parameters is the cross entropy. MAP parameters are $\theta = \{Re, Te, h_0, W_r, W_{h_{i-1}}, W_C, b_h, b_o\}$ where h_0 is the initial recurrent layer vector.

6.3 Experiments and Results

In this section, we address the performance of MAP by conducting tests using two real-life datasets.

Table 6.1: Models feature demonstration

Model	Recurrent	Input features		Repres. learning		Attention
		Space	Time	Space	Time	
NN	no	yes	no	no	no	no
RNN	yes	yes	no	no	no	no
noAtten	yes	yes	no	yes	no	no
STF-RNN	yes	yes	yes	yes	yes	no
MAP	yes	yes	yes	yes	yes	yes

6.3.1 Experimental Settings

We use two publicly available datasets that contain data on people's locations: GeoLife (Zheng et al., 2010) and Gowalla (Cho et al., 2011).

We use Precision and Recall to assess the efficiency of the prediction models.

Three-fold cross validation technique is used to train the model of each user in which the mobility data are partitioned into three sub-data of equal size. The Precision and Recall scores of each case from each user are then calculated and the final results of all users are averaged.

We examine two alignment functions (sum, concat) as described in Section 6.2.2. To evaluate the model effectiveness, we compare MAP with four outstanding proposals found in the literature: NN (Leca et al., 2015; L. Vintan and Ungerer, 2004), RNN (H. Kaaniche, 2010), and STF-RNN (Al-Molegi et al., 2016). In addition, we implemented a RNN that does not consider the attention technique (noAtten). The goal of these comparisons is to show how including the attention technique with recurrent structure has improved prediction overall performance. Models feature demonstration is shown in Table 6.1.

The common parameters of the models are given the same values. For example, hidden layers in NN, RNN and STF-RNN are set to 24. The number of training epochs is set to 100. The number of visited RoIs taken as input to the model is $w = 2$. The dimensionality of the embedded vector of the RoI d_r and the hidden layer d_h are 160 and 24, respectively. All these parameter values are obtained using the Grid Search.

Table 6.2: Performance comparison on the datasets evaluated by Precision@N and Recall@N. Best scores are in bold.

Dataset	Model	@1		@2		@3	
		R	P	R	P	R	P
GeoLife	NN	58.1	58.1	82.1	40.9	91.1	30.3
	RNN	67	67	86.5	42.4	92.7	30.8
	STF-RNN	73.3	73.3	87.7	44	93.1	31.1
	no-atten	71.95	71.95	87.37	44.19	92.69	31.28
	MAP-sum	73.73	73.73	**88.6**	**44.3**	**93.89**	**31.28**
	MAP-concat	**73.8**	**73.8**	88.03	44.02	93.54	31.17
Gowalla	NN	34.13	34.13	48.44	24.12	55.33	18.56
	RNN	34.32	34.32	48.68	24.34	55.67	18.56
	STF-RNN	39.68	39.68	52.34	25.17	60.21	19.4
	no-atten	39.08	39.08	54.73	21.91	61.96	13.39
	MAP-sum	**41.11**	**41.11**	**57.96**	**24.26**	**65.13**	**15.41**
	MAP-concat	40.77	40.77	56.27	23.41	63.43	14.84

6.3.2 Results and Analysis

Table 6.2 compares the results in terms of P@N and R@N with $N = 1, 2$ and 3. It is shown that the models that used a RNN structure perform better than NN. This indicates that RNN is effective in modeling trajectory sequences, which enables the models to get more accurate results by taking into account historical dependencies. RNN improves the results compared to NN, but does not model well the movement sequences without taking into consideration the movement time. noAtten greatly improves the performance comparing with RNN due to the fact that the embedding representation learning of the space input feature enable the model to extract the embedded semantic information about the users' behaviour more efficiently.

Another great improvement is achieved by STF-RNN though using the time feature in the model operations. Concat-MAP and sum-MAP consistently perform better than STF-RNN, and this demonstrates that the proposed integration of space and time using the attention technique performs better than using space and time as features to be input to the model. For instance, MAP-concat outperforms STF-RNN and noAtten by 0.68% and 2.57%, in terms of R@1 on GeoLife dataset. On Gowalla dataset, MAP-sum outperforms STF-RNN and noAtten by 3.6% and 5.2%.

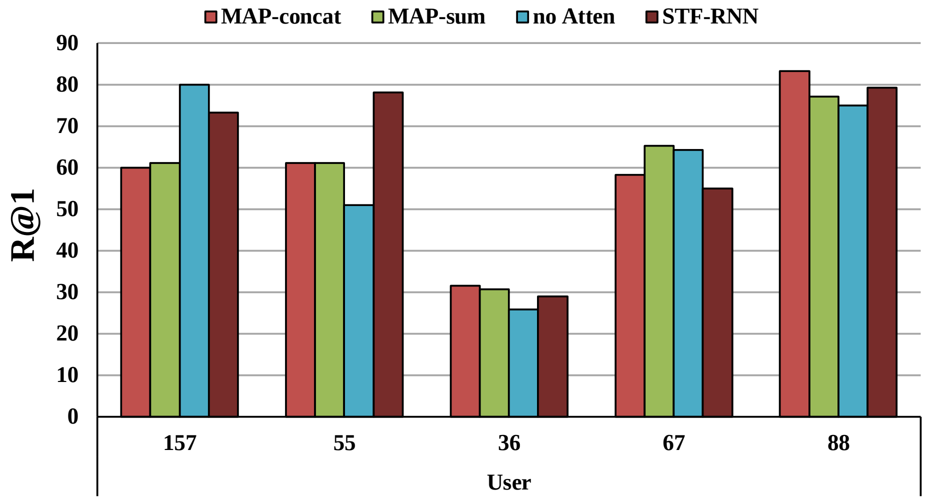


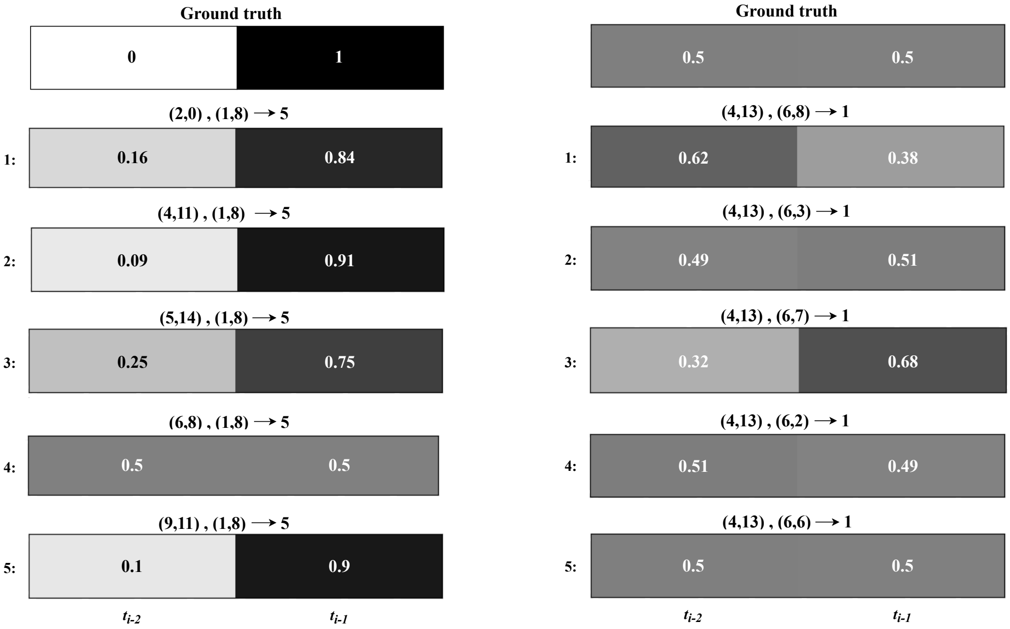
Figure 6.2: Models performance at users level.

All in all, MAP outperforms the other models, which implies that MAP can effectively learn to align RoIs and the movement time intervals. Our analysis shows that MAP attention-based model is superior to non-attentional ones in finding the relation between the two main parts of the mobility dataset (RoI sequences and movement times). Using the alignment function (MAP-concat), the model achieved better accuracy with 73.8%.

In spite of the slight differences between the four prediction models architecture (*i.e.* MAP-concat, MAP-sum, noAtten and STF-RNN), different prediction accuracies are exhibited when looking at user's individual result. The performance of each model varies greatly, as shown in Figure 6.2 for five selected users. We explain these variations by the fact that the data input types and the way of exploiting them have different impact on the prediction performance. Besides that, the different types of people's behaviour and few mobility data of some users are the major causes of such a difference in prediction performance.

6.3.3 Qualitative Analysis

To analyse the learned attention model, we extract the attention weight vector computed in Equation 6.7 and visualize the attention weights accordingly. Figure 6.3 shows the representation of how the attention model focuses on the temporal input that influences a given RoI. The gray level indicates the importance degree of



(a) Case 1.

(b) Case 2.

Figure 6.3: Attention visualization. Higher gray level means higher contribution.

the weight in attention vector, where high gray level means high contribution.

To predict the next RoI r_j, the model inputs are the previous and current RoIs and times where $w = 2$: $(r_{i-2}, t_{i-2}), (r_{i-1}, t_{i-1}) \rightarrow r_j$. We study two cases. The first case is where the prediction depends on the current RoI and time. This case can be interpreted as the following rule: *if* $r = r_{i-1}$ *and* $t = t_{i-1}$ *then next RoI is* r_j. The second case is where the only important information for the prediction is the current RoI. This case can be interpreted as the following rule: *if* $r = r_{i-1}$ *then next RoI is* r_j.

Our hypothesis is that the attention model is able to learn the importance of the time information, this importance is given by the alignment vector in Equation 6.7. For the first case, the model must give the importance to the input time t_{i-1} (*i.e.* its weight should be 1) and discard the input time t_{i-2} (*i.e.* its weight should be 0), see the ground truth in Figure 6.3a. Regarding the second case, there is no importance for the times information; thus, the model must give them the same weight (*i.e.* 0.5) for both time inputs as shown in the ground truth in Figure 6.3b.

To examine whether the model satisfies this hypothesis, we analyse the

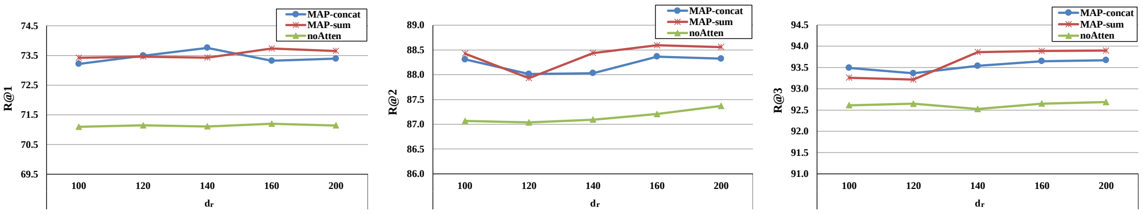


Figure 6.4: Impact of RoI embedded vector dimensionality parameter d_r.

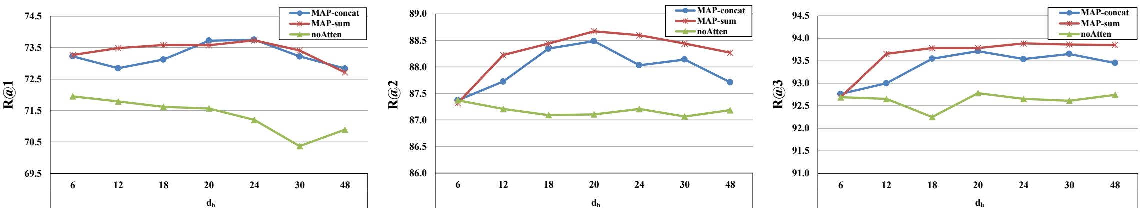


Figure 6.5: Impact of hidden layer embedded vector dimensionality parameter d_h.

transaction matrix of one user and randomly select some examples that satisfy the two cases mentioned above. Figure 6.3a shows the examples of case 1, the output of the attention model and their visualization. It is clearly shown that the attention model gives the highest weight to the time input t_{i-1} which has high impact on the next RoI decision. In the fourth example, the model gives same weight for both t_{i-2} and t_{i-1} and this is reasonable as they are identical (*i.e.* both of them equal 8). Figure 6.3b shows the visualization of the attention weights for some examples that satisfy the second case. The model gives approximately same weight for both t_{i-2} and t_{i-1} as they have the same impact (*i.e.* there is no importance for the time information). Overall, the model successfully learns to highlight the relevant part of time inputs which helps to improve the prediction performance.

6.3.4 Effects of Parameters

In addition, we study the effects of the dimensionality of both RoI and hidden layer embedded vectors.

To assess the effect of the dimensionality of RoI embedded vector d_r, R@N results are shown in Figure 6.4. We observe that the models performance gives the best results as long as we enlarge the value and then it decreases slightly in terms of R@1. Small d_r (*e.g.*, less than 140) means that less RoI information has to be

Table 6.3: Running time in seconds.

Model	Training	Predicting	Epoch	All op.
STF-RNN	0.235±0.0029	0.00029±0.000009	0.262±0.003	153.5±1.43
MAP-concat	0.197±0.0027	0.00025±0.000009	0.22±0.0028	122.14±1.29
MAP-sum	0.183±0.0026	0.00025±0.000008	0.206±0.0027	117.97±0.361
noAtten	0.161±0.0021	0.00022±0.000007	0.181±0.0021	105.21±0.529

considered by the model which limits its efficiency in discovering dependencies and, hence, impairing its performance.

Regarding the effect of hidden layer embedded vector dimensionality d_h on the model performance, the performance improves with the increase of d_h (see Figure 6.5). The best performance reaches its peak with $d_h = 24$ which indicates that the model learns to divide the movement time into 24 intervals. Small (large) d_h means that few (many) time intervals are provided, which fails to capture the relationships between the RoIs and times.

6.3.5 Running Time

Finally, we measure the exact running time of MAP and STF-RNN models. We simply run the models on trajectories of a randomly selected user and measure the amount of time for four cases, namely training, predicting, one epoch and all the operations. The experiment is repeated 10 times for each model. The average and standard deviation are then calculated. All experiments were conducted on iMac PC with 3.06 GHz Intel Core 2 Due CPU and 4 GB memory. Table 8.9 shows the models' exact running time in seconds. It can be noted that MAP models running times are less than STF-RNN's. This is due to the small number of MAP parameters compared to the number of STF-RNN parameters.

6.4 Summary

In this chapter, we have proposed MAP, an attention-based neural network for the problem of predicting people's next location. Our proposed model tends to align time intervals in people's trajectories that are relevant to a specific location. The proposed model learns to weight location inputs according to the movement time intervals associated in the trajectories. Experiments on GeoLife and Gowalla datasets show that MAP has improved the prediction effectiveness compared to state-of-the-art models.

CHAPTER 7

Convolutional Neural Network for Predicting People's Next Location

7.1 Introduction

The convolution technique has shown to be quite effective in exploiting the correlation of various types of data which is considered as the key to the success of CNN for a variety of tasks (LeCun et al., 2015). A CNN assumes that there is a very specific structure in the data, where inputs that are close to each other are related, whereas inputs that are further away are less related. In images, this makes sense since we normally have patches of similar texture, lighting and colour. In speech, words that occur close to each other in a sentence or a paragraph are more likely to share some semantic meaning. In sound, there are similar patterns where

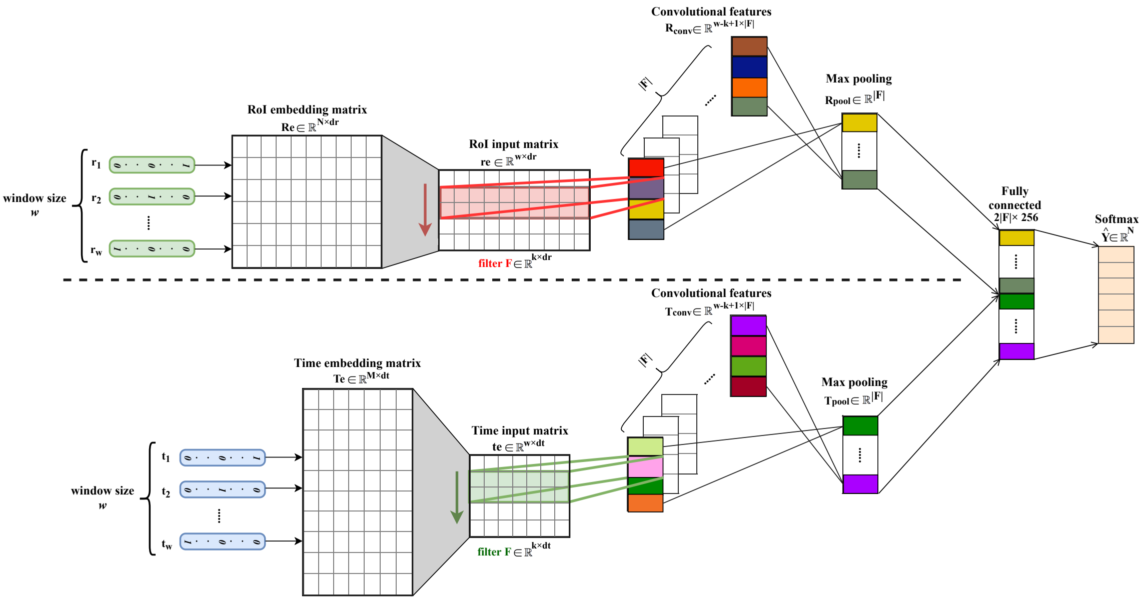

Convolutional features
$R_{conv} \in \mathbb{R}^{w-k+1 \times |F|}$
|F|
Max pooling
$R_{pool} \in \mathbb{R}^{|F|}$
RoI embedding matrix
$Re \in \mathbb{R}^{N \times dr}$
RoI input matrix
$re \in \mathbb{R}^{w \times dr}$
window size
w
r_1
r_2
r_w
filter $F \in \mathbb{R}^{k \times dr}$
Fully connected
2|F|× 256
Softmax
$\hat{Y} \in \mathbb{R}^N$
Convolutional features
$T_{conv} \in \mathbb{R}^{w-k+1 \times |F|}$
|F|
Time embedding matrix
$Te \in \mathbb{R}^{M \times dt}$
Max pooling
$T_{pool} \in \mathbb{R}^{|F|}$
Time input matrix
$te \in \mathbb{R}^{w \times dt}$
window size
w
t_1
t_2
t_w
filter $F \in \mathbb{R}^{k \times dt}$

Figure 7.1: ST-CNN model architecture.

the sound spectra at time steps close to each other tend to be similar and particular transitions from one phoneme to another are more common than others in a given language. In people's mobility modeling, the movement flow pattern of individuals from one location to another is required to extract. One promising approach is to adopt the fast, scalable, and end-to-end trainable CNN.

In this chapter, we use CNN architectures in order to build location prediction model called ST-CNN. We conduct the experiments using two large real-life mobility datasets to evaluate the efficiency of the proposed model. The use of large-scale datasets is one of the extremely important issues when using CNN, which have massive parameter numbers to be updated.

7.2 ST-CNN: Model Description

The overall structure of ST-CNN model is shown in Figure 7.1. The ST-CNN model (Algorithm 2) is composed of an embedding layer for both RoIs and time (lines 3-6) followed by a convolutional and max pooling layers (lines 7-18), fully connected network (lines 19-21) and a softmax classification layer (lines 22-26).

The input layer consists of two vectors. The first vector $r_i \in \mathbb{R}^N$ is the RoI ID

Algorithm 2: ST-CNN Model.

```
Input  : $Traj$: user's trajectories represented as a sequence of movements,
         each movement contains $(r, t)$, $r$ is RoI $\in \mathcal{R}$ and $t$ is movement time
         $\in \mathcal{T}$, $w$: number of visited RoIs, kernels, $|F|$: number of filters.
Output: $S$: set of predicted RoIs.
$S \leftarrow []$
for $x \leftarrow 1$ to $Traj.length - w + 1$ do
    for $i \leftarrow x$ to $x + w$ do
        $re$.append(embedding($Traj[i].r$))
        $te$.append(embedding($Traj[i].t$)
    end
    // RoIs conv
    for $k \leftarrow 1$ to kernels do
        $R_{conv} \leftarrow$ Conv1D($|F|, k, re$)
        $R_{conv} \leftarrow$ relu($R_{conv}$)
        $R_{pool} \leftarrow$ MaxPool1D($w - k + 1, R_{conv}$)
    end
    // Time conv
    for $k \leftarrow 1$ to kernels do
        $T_{conv} \leftarrow$ Conv1D($|F|, k, te$)
        $T_{conv} \leftarrow$ relu($T_{conv}$)
        $T_{pool} \leftarrow$ MaxPool1D($w - k + 1, T_{conv}$)
    end
    $RT_{concat} \leftarrow$ Concat($R_{pool}, T_{pool}$)
    $fc \leftarrow$ Dense($256, RT_{concat}$)
    $fc \leftarrow$ relu($fc$)
    //Classifier
    $fc \leftarrow$ Dense($|\mathcal{R}|, fc$)
    $\hat{y} \leftarrow$ softmax($fc$)
    $Indx \leftarrow$ argmax($\hat{y}$)
    $S$.append($\mathcal{R}[Indx]$)
end
return $S$
```

where N is the number of RoIs. The second vector $t_i \in \mathbb{R}^M$ represents the leaving time (in hours) from the RoI (Al-Molegi et al., 2018) and M is the number of hours per day (*i.e.* 24 hours). These two vectors are encoded using *one-hot encoding* (Harris and Harris, 2012).

The two input vectors are passed through an embedding layer in order to learn a meaningful representation of the RoIs and the leaving times input features. This representation enables the model to capture the embedded semantic information about user behaviour and as a consequence improving the prediction performance. The embedded matrix $Re \in \mathbb{R}^{N \times d_r}$ represents a set of RoIs where N is the number of RoIs and d_r is the dimensionality of the embedded vector. Similarly, $Te \in \mathbb{R}^{M \times d_t}$ is the embedded matrix that represents a set of times and d_t is the dimensionality of the embedded vector. If w is the number of movements $\mathcal{M}$ taken as inputs to the model, then $re \in \mathbb{R}^{w \times d_r}$ and $te \in \mathbb{R}^{w \times d_t}$ are the RoI and time inputs respectively.

To learn to capture and compose features of movement sequences, the neural network applies a series of transformations to the RoI and time input matrices using convolution, nonlinearity, pooling and concatenation operations.

A convolution operation involves a *filter* $F_i \in \mathbb{R}^{k \times d_r}$, which is applied to each window size w comes from Re matrix $\{re_{1:k}, re_{2:k+1}, \ldots re_{w-k+1:w}\}$ where k is the filter height. This operation results in a vector $R_{conv_i} \in \mathbb{R}^{w-k+1}$ which is computed as follows:

$$R_{conv_i} = f(F_i \otimes re_{i:i+w-1} + b) \tag{7.1}$$

where $\otimes$ is the element-wise multiplication, and f is a non-linear function such as a Rectified Linear Unit (ReLU) and b is a bias vector.

The previous operation is applied using a single filter. For a richer feature representation of the input data, we apply a set of filters that work in parallel generating multiple feature maps. If the number of filters are $|F|$, then a feature map matrix $R_{conv} \in \mathbb{R}^{w-k+1 \times |F|}$ is obtained.

$$R_{conv} = [R_{conv_1}, R_{conv_2}, \ldots R_{conv_{|F|}}] \tag{7.2}$$

The same procedures are applied to the time input matrix and finally, feature map matrix $T_{conv} \in \mathbb{R}^{w-k+1\times|F|}$ is obtained.

After passing the convolutional layer outputs (R_{conv} and T_{conv}) through the activation function, it is passed to the pooling layer in order to aggregate the information and reduce the representation. We apply a max pooling operation which simply returns the maximum value to capture the most important feature R_{pool} and T_{pool}.

The output of the pooling layer RT_{concat} is passed to a fully connected softmax layer to obtain the probability distribution over the RoIs.

$$\begin{aligned} P(\hat{r}_{i+1} = r_j|\mathcal{M}_i, \ldots, \mathcal{M}_{i-w}) &= softmax(RT_{concat} \cdot W + b_o) \\ &= \frac{e^{RT_{concat}\cdot W_i + b_{s_i}}}{\sum_{j=1}^{N} e^{RT_{concat}\cdot W_j + b_{s_j}}} \end{aligned} \tag{7.3}$$

where W and b_o are the model parameters to be trained.

Adam optimization algorithm (Kingma and Ba, 2014) is used to train the network, while BPTT algorithm is used to compute the gradients. The model parameters are $\theta = [Re, Te, F, W, b, b_o]$ where Re and Te are the embedding matrices of the RoIs and time, respectively. F is the set of filters and b is the convolutional bias. W and b_o are the weight and bias of the softmax layers, respectively. The cost function used is the cross entropy.

7.3 Experimental Evaluation

In this section, we aim at demonstrating the performance of the ST-CNN model, by conducting tests using real-word GPS dataset of individuals.

7.3.1 Experimental Settings

We use two publicly available datasets: GeoLife (Zheng et al., 2010) and Gowalla (Cho et al., 2011).

Table 7.1: Performance comparison on the datasets evaluated by Recall@N and Precision@N. Best scores are in bold.

Dataset	Model	@1		@2		@3	
		R	P	R	P	R	P
GeoLife	*n*-MMC	58.9	58.9	78.7	39.3	87	29
	NN	58.1	58.1	82.1	40.9	91.1	30.3
	RNN	67	67	86.5	42.4	92.7	30.8
	STF-RNN	73.3	73.3	87.7	44	93.1	31.1
	GTR	73.6	73.6	88.62	44.31	93.73	31.2
	ST-CNN	**74.07**	**74.07**	**88.62**	**44.32**	**94.2**	**31.2**
Gowalla	*n*-MMC	17.1	17.1	23.64	11.82	30.05	10.02
	NN	34.13	34.13	48.44	24.12	55.33	18.56
	RNN	34.32	34.32	48.68	24.34	55.67	18.56
	STF-RNN	39.68	39.68	52.34	25.17	60.21	19.4
	GTR	40.59	40.59	52.69	26.47	60	19.91
	ST-CNN	**41.28**	**41.28**	**53.9**	**27**	**60.73**	**20.20**

We compare the prediction model performance with four outstanding proposals found in the literature: ***n*-MMC** (Gambs et al., 2012), **NN** (Parija et al., 2013c; Leca et al., 2015; L. Vintan and Ungerer, 2004), **RNN** (H. Kaaniche, 2010), **STF-RNN** (Al-Molegi et al., 2016) and **GTR** (Al-Molegi et al., 2017) which used embedding representation and neural pooling function of input data.

The parameters of our model are as follows: the window size w is set to 2. The dimensionality of the embedded vector of the RoI (d_r) and time (d_t) are 160, 6 respectively. The height of the convolution filters k is set to 2 and the number of convolutional feature maps is 175. We use ReLU activation function and a simple max-pooling function.

Precision and Recall are employed as our evaluation metrics in all experiments in order to assess the efficiency of the prediction models. The model of each user is trained on its own mobility data using three-fold cross validation technique. The Precision and Recall scores of each case from each user is then calculated and the final results of all users are averaged.

7.3.2 Results and Analysis

The experiment results on GeoLife and Gowalla datasets in terms of P@N and R@N are shown in Table 7.1. The worst results are obtained by n-MMC and NN on GeoLife due to only considering the movement sequences without the time. On Gowalla dataset, it is clearly shown that NN results in a better prediction accuracy than n-MMC. This is due to the large training data size which results in a good model performance. Furthermore, the results demonstrate the models ability that used a RNN structure to successfully analyse movement sequences by taking into account historical dependencies which enables the models to get more accurate results. A great improvement is achieved by STF-RNN though using the internal representations learning of the space and time features. In addition, it can be observed form the table that GTR slightly improves the results comparing with STF-RNN due to using the time encoding scheme in the model operations which helps the model to effectively capture the temporal effects. We further notice that our model outperforms STF-RNN, GTR and the other baseline models on both datasets. For instance, ST-CNN outperforms STF-RNN and GTR by 1.1% and 0.64%, respectively, in terms of P@1 on GeoLife. This is mainly because that CNN explicitly employs the interactive information, while most other methods only rely on the global information.

7.3.3 Summary

In this chapter, we have proposed a model for the problem of predicting the future location of people movement. An embedding learning layer is used to effectively discover adequate internal representations of space and time input features enabling the model to capture the embedded semantic information about the users's behaviour more effectively. The CNN is used to discover long-term dependencies which increases the efficiency of the proposed model. Experiments on two datasets (GeoLife and Gowalla) show that the model has improved the prediction effectiveness compared to state-of-the-art models.

Part III

Contributions to Specific Applications

CHAPTER 8

RoI Discovering and Predicting in Smartphone Environments

8.1 Introduction

Discovering RoIs (historical locations where smartphone users frequently visit) and predicting people's next RoI during their daily life are a key component in the success of modeling people's mobility.

Most of the existing studies for discovering RoIs depend on extracting contiguous GPS points that satisfy threshold conditions (*e.g.*, stay time, distance, etc.). If contiguous GPS points satisfy the threshold conditions, it can be concluded that a RoI has been discovered. Moreover, some other studies are based on clustering algorithms such as DJ (Zhou et al., 2004) and *k*-means. In general, few studies focused on discovering RoIs to predict people's mobility.

Smartphone devices are used to collect the traces of outdoor environments, thus, most of the recorded GPS points will be located in a regions that connect significant locations. This information helps to segment the trajectory into paths and significant locations. In this chapter, a method to discover RoIs is proposed which can help in building an accurate location prediction model. The proposed method detects the paths and associates them with the nearest significant locations.

It is worth mentioning that the accuracy of location prediction models depends on the completeness or comprehensiveness of the collected data. Integrating space and time context is not necessarily to improve the prediction performance due to the different degrees of importance of various contexts when generating the information for location prediction. Moreover, the appropriate integration between space and time is still relatively unexplored. The time context must be formulated in a way to add extra information to the space context. To this end, including time context in a specific way could help in detecting human movements changes and as consequence, enhance the prediction model performance.

Any successful location prediction model should target three major goals: location prediction accuracy, high throughput or fast response and efficiency in terms of utilizing smartphone resources. Markov model becomes a better choice to predict people' next locations due to the high prediction accuracy and low complexity compared to other models such as NN, LZ, Prediction by Partial Match (PPM) and Sampled Pattern Matching (SPM) (Song et al., 2006).

This chapter claims three contributions:

- Firstly, a new method to discover RoIs located in people's trajectories is proposed, which helps to build an accurate users' next location prediction model. The method starts by discovering Candidate RoIs (CRoIs) based on three types of thresholds: distance, time, and region density. Then, the DBSCAN algorithm (Ester et al., 1996) is used to cluster the CRoIs into different RoI groups. Once the RoIs are discovered, the previously collected GPS points are converted into a sequence of RoIs that represent a series of locations visited by a user. Soundness and completeness are used as metrics to evaluate the performance of the proposed method to discover RoIs.

- Secondly, a prediction model is proposed to predict people's next location; a **s**pace-**t**ime-based Generalized Markov Model (st-GMM). In st-GMM, MC model is improved to include the time context factor where different space-time integration methods are employed. To improve st-GMM performance, the model also applies explicit feedback during the testing phase to automatically update its transitions content. Due to the high prediction performance and low complexity of the models, they could be applied to an environment with limited resources such as services and applications installed in smartphones.
- Thirdly, we use two real-life datasets to evaluate the proposed approach. The experimental results demonstrate that our approach performs well in terms of the evaluation metrics, as well as time complexity.

8.2 The Proposed Approach

Figure 8.1 shows our proposed next interest region discovering and prediction approach which includes four main steps:

- In the first step, the dataset is preprocessed to detect and remove possible noise.
- In the second step, RoIs located in a user movement region are discovered.
- The third step involves several sub-steps: discretization, reduction, quantization and transformation of the training dataset, and building a prediction model.
- The last step involves evaluating the prediction model using a testing dataset.

Next sections cover with details the different steps except the first one (data preprocessing) which is described in Section 3.2.1.

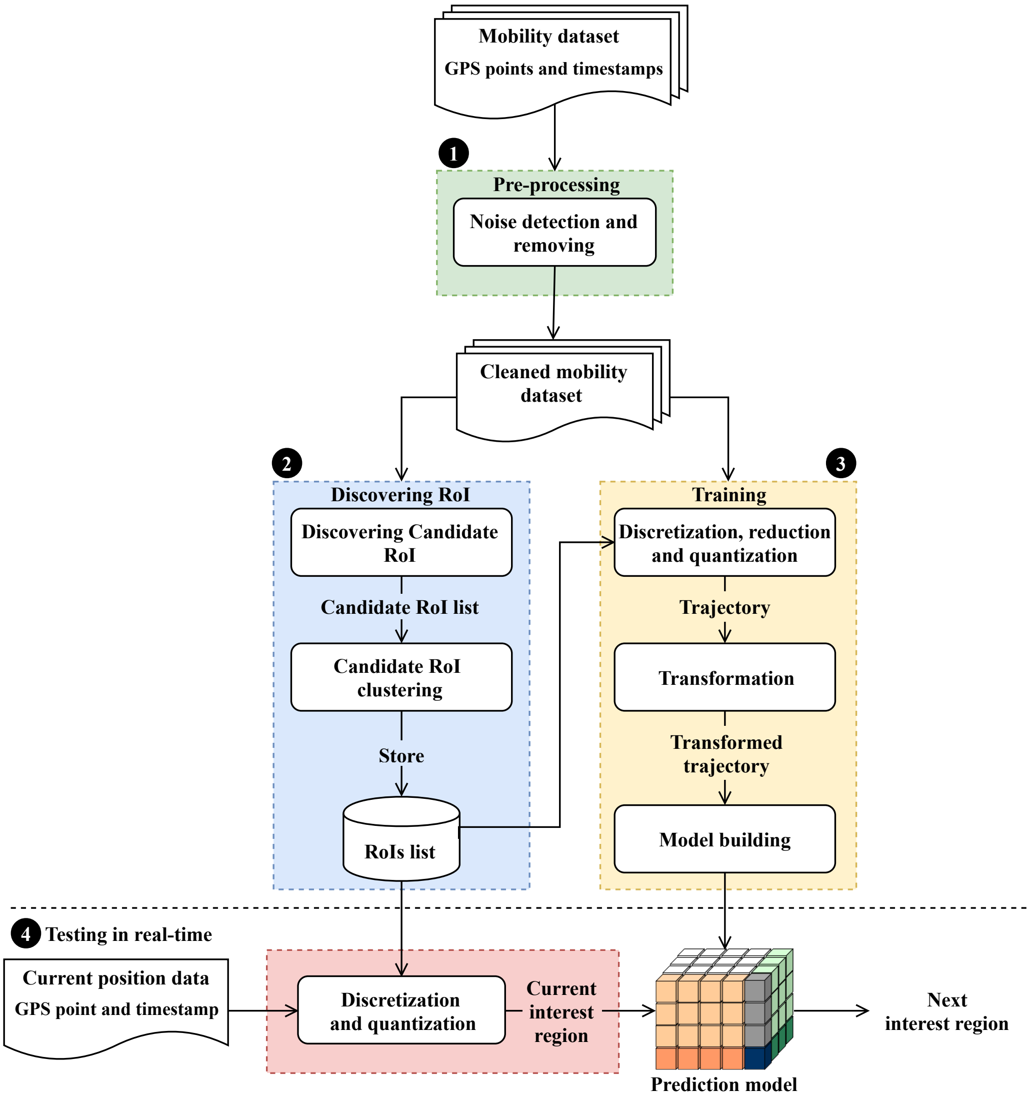


Figure 8.1: Our proposed next interest region discovering and prediction approach.

8.2.1 Discovering RoIs

RoIs discovery method has to satisfy the soundness and completeness properties which are used as metrics to evaluate the effectiveness of the method. More details are provided in Section 8.3.2.1.

We propose Discovering RoI (DRoI) (see Algorithm 3); a new method that takes a sequence of GPS points from user's mobility data in order to produce a set of RoIs $\mathcal{R}^u = \{r_1, r_2, \dots, r_m\}$, where m is the number of the obtained RoIs.

DRoI is accomplished in two levels: the first level is called Discovering Candidate RoI (DCRoI) (lines 1 to 33) and the second one is called Candidate RoI Clustering

Algorithm 3: Discovering RoI.

```
Input  : $\mathbb{G}$: a sequence of GPS-Point and time, $\delta$: distance threshold, $\tau$: time
         threshold, $rd$: region density threshold.
Output: A set of RoI.
$CRoI$.add($\mathbb{G}$[1].GPS-Point)
$CRoIindex \leftarrow 1$
$RegionDensity$.add(1)
for $i \leftarrow 2$ to $\mathbb{G}.length$ do
    $isBelongToCRoI \leftarrow$ False
    $mini \leftarrow \infty$
    for $j \leftarrow 1$ to $CRoI.length$ do
        $dist \leftarrow$ HaversineDistance ($\mathbb{G}[i]$.GPS-Point,$CRoI[j]$) // computing the
         distance
        if $dist < \delta$ then
            $isBelongToCRoI \leftarrow$ True
            $RegionDensity[j] \leftarrow RegionDensity[j] + 1$
            break
        else if $dist < mini$ then
            $mini \leftarrow dist$
            $x \leftarrow$ j //$x$ is the index of the nearest CRoI
        end
    end
    if not $isBelongToCRoI$ then
        $\Delta$T $\leftarrow \mathbb{G}[i].time$ - $\mathbb{G}[CRoIindex].time$ // computing the overall spent
         time
        if $\Delta T > \tau$ then
            if $RegionDensity[x] > rd$ then
                // region density
                $CRoI$.add($\mathbb{G}[i]$.GPS-Point) // new CRoI
                $RegionDensity$.add(1)
                $CRoIindex \leftarrow$ i
            else
                $RegionDensity[x] \leftarrow RegionDensity[x] + 1$
            end
        else
            $RegionDensity[x] \leftarrow RegionDensity[x] + 1$
        end
    end
end
RoI $\leftarrow$ DBSCAN($CRoI$)
return $RoI$
```

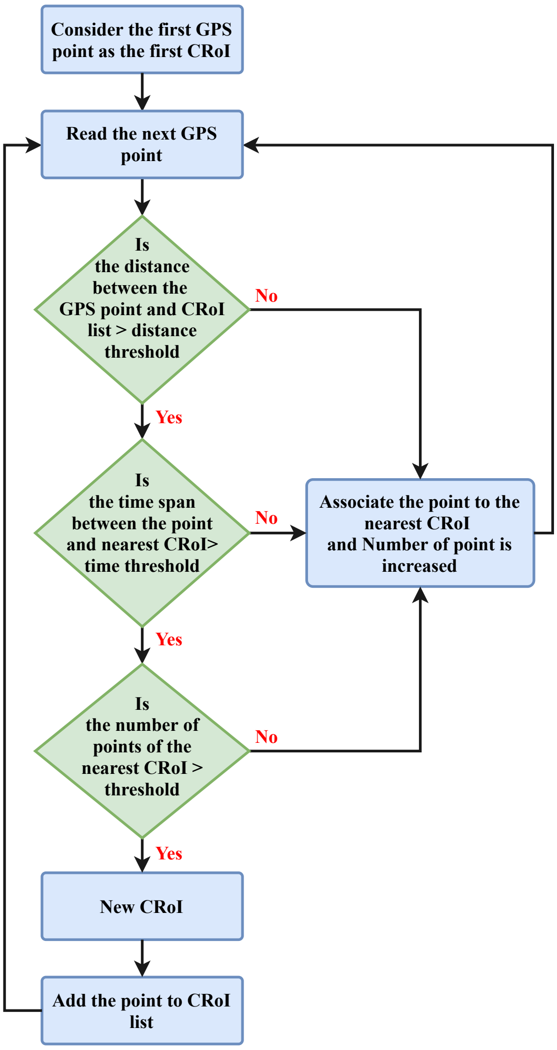

Consider the first GPS point as the first CRoI
Read the next GPS point
Is the distance between the GPS point and CRoI list > distance threshold
No
Yes
Is the time span between the point and nearest CRoI> time threshold
No
Associate the point to the nearest CRoI and Number of point is increased
Yes
Is the number of points of the nearest CRoI > threshold
No
Yes
New CRoI
Add the point to CRoI list

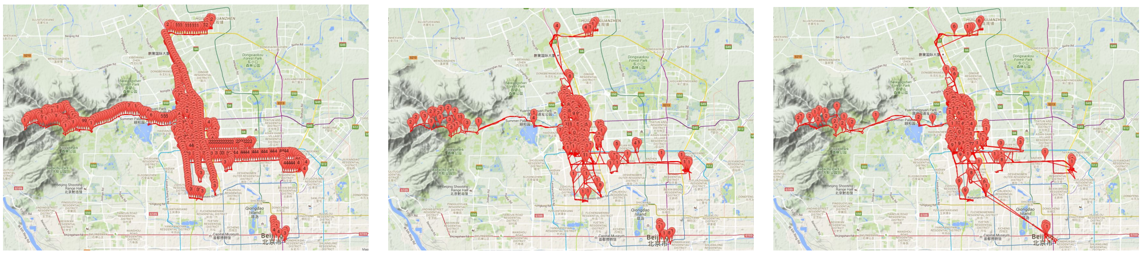
Figure 8.2: DCRoI flowchart.

(CRoIC) (line 34).

- DCRoI: In the first level, mobility data is grouped based on three types of thresholds: distance (δ), stay time (τ) and region density (rd), where δ, τ and rd are three tuning parameters.
- CRoIC: Based on DBSCAN algorithm, the second level performs clustering on the CRoIs to obtain the RoIs.

8.2.1.1 DCRoI

Figure 8.2 shows a flowchart of DCRoI method. The first GPS point is added to the CRoI list (line 1). It is assumed that participants have started collecting GPS



(a) After applying distance threshold.

(b) After applying distance and time thresholds.

(c) After applying distance, time and region density thresholds.

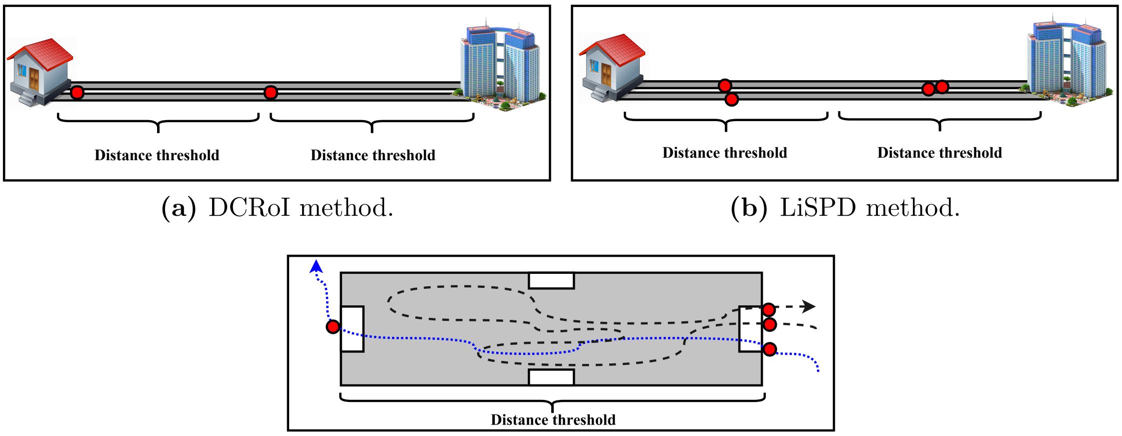
Figure 8.3: The first level: Discovering CRoI.

points from RoIs. The GPS points of each discovered RoI are counted. Then, all GPS points are compared with the CRoI list, which starts with only one at the beginning. We begin by examining the first threshold, which is the distance δ (lines 7-17) computed using Haversine Distance, Equation 8.1:

$$distance = 2r \arcsin\left(\sqrt{\sin^2(\tfrac{\phi_2-\phi_1}{2}) + \cos(\phi_1)\cos(\phi_2)\sin^2(\tfrac{\lambda_2-\lambda_1}{2})}\right) \tag{8.1}$$

where ϕ is latitude, λ is longitude and r is the radius of the sphere. If the distance is less than a given threshold, the current GPS point is associated with the nearest CRoI and its density is increased by one. Otherwise, the second threshold (time spent in the current CRoI τ) is computed (line 19). If the spent time is less than a given threshold, the current GPS point is also associated with the nearest CRoI and one point is counted. Otherwise, the last threshold (the density of the nearest CRoI to the current point rd) is compared to a given threshold (line 21). The current GPS point is also associated with the nearest CRoI in case the region density is less than a given threshold. In other words, if the GPS points satisfy the three threshold conditions, they are considered as new CRoIs. Otherwise, they will be associated with the nearest CRoI. The threshold conditions are as the followings:

- The distance between any two CRoIs must be larger than a given threshold, Figure 8.3a.
- The time spent in a CRoI must be longer than a given threshold, Figure 8.3b.
- The density of the nearest CRoI to the current point must be greater than a



(a) DCRoI method.

(b) LiSPD method.

(c) Indoor senario.

Figure 8.4: Discovering CRoI in mobility data.

given threshold, Figure 8.3c.

According to Klepeis et al. (2001), it is assumed that people spend 87% of their time indoors, 8% outdoors and 5% in a vehicle. During outdoors, GPS points are recorded when users move from one RoI to another. Thus, the paths that connect the RoIs must be given a higher attention than the RoIs themselves. The GPS points that don't satisfy the threshold conditions, will be then associated with the nearest CRoI, instead of ignoring them, as in LiSPD. Therefore, the RoI will contain the interest region itself and the relevant paths that lead to a another RoI.

Figure 8.4 illustrates the behaviour of the our proposed method compared to LiSPD according to some scenarios. To move from one RoI to another, the state of the user changes from indoor to outdoor and vice versa. Let us consider the user's commuting movement. Suppose that the distance between home and workplace locations is two times greater than the threshold distance. In our method, only two CRoIs are considered, Figure 8.4a. The first one is the first recorded GPS point when the user leaves home which represents the identifier of the first region. The second CRoI is considered at the beginning of the second region after the threshold conditions are satisfied. In LiSPD method, four CRoIs are considered (Figure 8.4b). After time and distance thresholds, the first and second points are discovered in the first and second regions, respectively. For each region, the mean values of all recorded points are calculated. The same scheme is repeated when the user returns home. It can be noticed that the two CRoIs that were discovered by our proposed method are more accurate than those obtained by the LiSPD method. The four points discovered by LiSPD are located in insignificant locations such as road. In addition, discovering many CRoIs in the first level negatively affects the clustering process of the second level and RoI discovering process.

For more clarification, consider the user moves to a new indoor RoI such as the workplace (Figure 8.4c). Only one RoI should be discovered for that building. The last recorded GPS point before the user enters the workplace building is considered as CRoI, assuming that the threshold conditions are satisfied at this point. Inside the building, a few GPS points might be recorded. When the user leaves the building, new GPS points are recorded. Two cases are presented in this situation based on region density threshold. In the first case where the region density threshold is not considered, the new recorded GPS point is considered as new CRoI if the distance between the last CRoI and the new recorded GPS point is larger than δ and the time span is greater than τ. When region density threshold is considered in the second case, the region density is not greater than rd. Thus, the new recorded GPS point is not considered as a new CRoI but referred to the nearest CRoI.

CRoIs list is used to avoid considering more than one CRoI in the same region. Region density threshold condition is used to address some of the noise cases that are more likely to occur during GPS points recording process, which could lead to considering a wrong RoI. If GPS points are recorded every T times and the threshold time that a user must stay in any region is τ, then any RoI region must contain at least $rd = \tau/T$ points. If GPS signal is lost for any reason, few GPS points may be recorded in such regions. Thus, these points will be assigned to the nearest CRoI even if the time and distance thresholds are satisfied. In a normal case, during time threshold and within a distance threshold, rd points are recorded and CRoIs are detected.

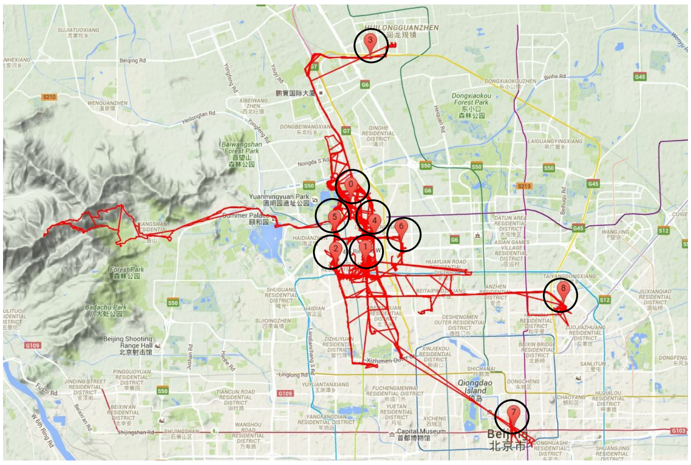


Figure 8.5: The second level: CRoI clustering.

8.2.1.2 CRoI Clustering

It can be observed that, in general, the density of GPS points in RoI is higher than in other regions because people tend either to move slowly or not to move at all. To obtain a set of RoIs in the movement area, DBSCAN (one of the most common clustering algorithms) is utilized to cluster CRoIs (Figure 8.5). Different clustering algorithms can be used in this level. We use DBSCAN because it does not require specifying the number of clusters a priori as in the case of k-means. It also groups points together with many nearby neighbours and marks outliers that are too far from the nearest neighbours.

Time complexity For a dataset with n GPS points and m CRoIs, the DRoI method has the following time complexity features:

- For every iteration, distances, elapsed time and region density between GPS points and the list of all CRoIs are calculated $(m \cdot n)$, where $m < n$.
- Time complexity of DBSCAN algorithm is $\log m$ (Ester et al., 1996).
- Therefore, the time complexity is $m \cdot n + \log m$ instead of $n^2 + \log m$ in LiSPD.

8.2.2 Next Location Prediction Model Construction

This section presents the process of building the prediction model which considers both RoIs and time intervals as inputs and predicts user's next RoI. Various sub-steps are involved before building the prediction model: discretization, reduction, quantization and transformation. The first three sub-steps are described in Section 3.2.3. The transformation is described in the next section.

8.2.2.1 Generalized Markov Model: GMM

MC is one of the popular models that are used in people's movement prediction. A state in MC (state model) corresponds to the RoI while state transition corresponds to the movement from one RoI to the next. The procedures of building a first order MC model are different from second, third, and n-order MC models. In this study, a general transformation function is used to transform n-order MC model into first order. Thus, all n-order MC models will be treated as a first order which provides more abstraction for n-order MC. Here, order indicates the number of visited RoIs taken as input to the prediction model.

Given a set of observable objects, $\mathbb{O} = \{o_1, \ldots o_m\}$ and a sequence $\mathbb{S} = \{o_i, o_j, \ldots o_r, \ldots o_k, o_p, \ldots, o_l\}$ where $o_i, o_j, o_r, o_k, o_p, o_l \in \mathbb{O}$, then the n-transformed sequence, $\mathbb{S}^{(n)}$, is defined as the following:

$$\mathbb{S}^{(n)} = \left\{ \left(\underbrace{o_i, o_j, \ldots}_{n}; o_r \right), \ldots, \left(\underbrace{o_k, o_p, \ldots}_{n}; o_l \right) \right\} \tag{8.2}$$

Based on that, the n-order MC model can be defined as the following:

$$P(o_{n+1}|o_n, \ldots, o_1) = \frac{N(o_1, \ldots, o_n \rightarrow o_{n+1})}{\sum_{\acute{o} \in \mathbb{O}} N(o_1, \ldots, o_n \rightarrow \acute{o})} \tag{8.3}$$

where $N(o_1, \ldots, o_n \rightarrow o_{n+1})$ is the number of times $o_1, \ldots, o_n \rightarrow o_{n+1}$ occurs in $\mathbb{S}^{(n)}$. In the case of next location prediction, the interest regions represent the set of observable objects, while trajectories represent the sequence. The probability is represented using the five most different variables as it follows: $P(Nxt_r = r_j | Cur_r = r_i, Cur_t = t_i, Prv_r = r_{i-1} : r_{i-order}, Prv_t = t_{i-1} : t_{i-order})$ which means: What is the probability that the next RoI is $Nxt_r = r_j$ given that the current RoI is $Cur_r = r_i$, the current time is $Cur_t = t_i$, the previous RoIs are: $Prv_r = r_{i-1} : r_{i-order}$ and the

Table 8.1: s-GMM probability matrix.

	r_1	r_2	r_3	r_4	r_5
r_1	-	0.14	-	0.77	0.09
r_2	0.1	-	0.27	0.63	-
r_3	-	0.73	-	-	0.27
r_4	0.36	0.48	-	-	0.15
r_5	0.72	-	0.11	0.17	-

Table 8.2: t-GMM probability matrix.

	r_1	r_2	r_3	r_4	r_5
t_0	-	-	-	1	-
t_2	-	-	0.81	0.19	-
t_3	-	-	-	0.33	0.66
t_8	-	0.37	0.44	-	0.19
t_{13}	0.71	-	-	-	0.29
$\vdots$			$\vdots$		

previous times are: $Prv_t = t_{i-1} : t_{i-order}$. When order equals to 1, the previous RoIs and times are removed.

Definition 5 ***Trajectory Transformation.*** *The interest region of the trajectory is transformed into a pair of values* $((r_i, r_{i+1}, ..., r_{i+order-1}), r_{i+order})$, $i = 1 \ldots n$, *where the first value represents the current interest region preceded by* $order - 1$ *visited RoIs and the second one is the next RoI.*

Example 1 *Given* $\mathbb{O} = \{1, 2, 3, 4, 5\}$ *and* $\mathbb{S} = \{1, 4, 2, 3, 2, 3, 2, 1, 5, 1\}$.

The transformation of the trajectory to first, second and third orders based on equation (8.2) *will be as the following:*

$\mathbb{S}^{(1)} = \{$*(1→ 4), (4→ 2), (2→ 3), (3→ 2), (2→ 3), (3→ 2), (2→ 1), (1→ 5), (5→ 1)* $\}$.

$\mathbb{S}^{(2)} = \{$*(1, 4→ 2), (4, 2→ 3), (2, 3→ 2), (3, 2→ 3), (2, 3→ 2), (3, 2→ 1), (2, 1→ 5), (1, 5→ 1)* $\}$.

$\mathbb{S}^{(3)} = \{$*(1, 4, 2→ 3), (4, 2, 3→ 2), (2, 3, 2→ 3), (3, 2, 3→ 2), (2, 3, 2→ 1), (3, 2, 1→ 5), (2, 1, 5→ 1)* $\}$.

Based on the different types of trajectory information, different prediction models can be built. For the trajectory that involves only the space, a space-based model (s-GMM) is built, whereas a space-time based model (st-GMM) is built for the trajectory that has both space and time. Additionally, only the time information can be used without considering any space information, thus, a time-based prediction model (t-GMM) is built. Those were our different experimental models that we developed and set as the benchmark.

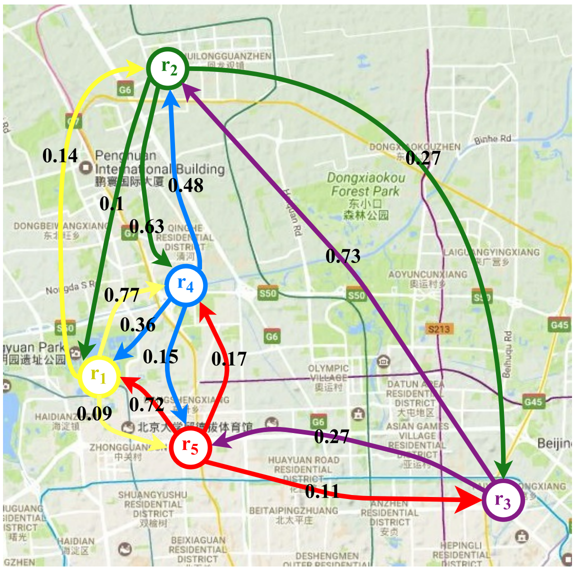


Figure 8.6: s-GMM transitions probability graph.

s-GMM. This model is similar to classical MC and it only considers RoIs (*i.e.* sequence of locations visited by a user). Thus, the observable objects-set of this model is $\mathbb{O} = \{r_1, r_2, \ldots, r_m\}$, where m is the number of discovered RoIs. To build n-order s-GMM, the RoI transitions matrix is calculated. First, we construct different prefixes n RoIs based on the trajectories. Then, we compute the frequency of each distinct RoI that appears after every prefix. The frequency value is set to 0 if there is no movement from any prefix to any RoI. The frequencies of the RoI transitions matrix are then normalized to get the probability matrix using equation 8.4 (see Table 8.1). The rows of the matrix represent the from-RoI(s) (the last n-visited interest region), while the columns represent the to-RoI (*i.e.* the next interest region). Each row of the matrix represents the predicted scores for RoIs of a certain user. s-GMM probability matrix can be represented as a graph where nodes and arrows represent the RoIs and the transitions between them, respectively (see Figure 8.6).

$$P(Nxt_r = r_j | Cur_r = r_i) = \frac{N(r_i \rightarrow r_j)}{\sum_{r_k \in \mathcal{R}} N(r_i \rightarrow r_k)} \tag{8.4}$$

The main issue in this model is that, the next RoI of the user in a specific region will be always the same region that has the high probability value. For example, based on the probability matrix in Table 8.1, the next RoI for r_1 is always r_4.

Table 8.3: The RoI transition matrix.

	r_1	r_2	r_3	r_4	r_5
r_1	-	6	-	33	4
r_2	5	-	13	30	-
r_3	-	11	-	-	4
r_4	24	32	-	-	10
r_5	13	-	2	3	-

Table 8.4: The time matrix.

	r_1	r_2	r_3	r_4	r_5
r_1	-	8,7	-	0,23,2	3,4,2
r_2	17,18	-	8,10	17,15,18	-
r_3	-	7,8	-	-	9,11,13
r_4	18,19,21	8,9	-	-	8,14
r_5	13,12,11	-	4	1,3,11	-

t-GMM. Typically, human mobility behaviour is strongly influenced by the time interval information such as movement time, time of the day, day of the week etc. In other words, humans have trends in what they do in the different times of the day, week, etc. This model considers the time intervals sequence to predict the next RoI. This means that the time intervals are used to produce the candidate RoIs based on their relationships with the current RoI. To build n-order t-GMM, the time transitions matrix is calculated. We First construct different prefixes n time intervals based on the trajectories. Then, we compute the frequency of each distinct RoI that appears after every prefix. The frequencies of the time transitions matrix are then normalized to get the time probability matrix using equation 8.5 (see Table 8.2). The rows of the matrix represent the from-time(s) (*i.e.* the current time when n equals to 1), while the columns represent the to-RoI (*i.e.* the next interest region).

$$P(Nxt_r = r_j | Cur_t = t_i) = \frac{N(t_i, r_j)}{\sum_{r_k \in \mathcal{R}} N(t_i, r_k)} \tag{8.5}$$

st-GMM. This model is similar to previous s-GMM except that it considers time intervals obtained from the GPS timestamp. The transformation of the trajectory will be modified to include the time interval t as it follows: $((r_i, r_{i+1},$ $\ldots,$ $(r_{i+order-1}, t_{i+order-1})), r_{i+order})$, where the first part contains the current RoI and time preceded by: $order-1$, visited RoIs and the second one is the next RoI.

To model space-time information, we consider three cases. The first case where the contribution of time information is considered as independent from the space. The model is referred to as: ind-st-GMM. The second case where the contribution of time and space information are dependent on each other, and the model is referred to as: dep-st-GMM. The final case where the contribution of space and time are

Table 8.5: Tran-Time matrix.

	r_1	r_2	r_3	r_4	r_5
r_1,7	-	2	-	-	-
r_1,8	-	4	-	-	-
r_1,0	-	-	-	19	-
r_1,23	-	-	-	13	-
r_1,2	-	-	-	1	2
r_1,3	-	-	-	-	1
r_1,4	-	-	-	-	1
r_2,17	3	-	-	8	-
r_2,18	2	-	-	13	-
r_2,8	-	-	-	5	-
r_2,10	-	-	-	8	-
r_2,15	-	-	-	9	-
r_3,8	-	6	-	-	-
r_3,7	-	5	-	-	-
r_3,9	-	1	-	-	-
r_3,11	-	2	-	-	-
r_3,13	-	1	-	-	-
r_4,18	8	-	-	-	-
r_4,19	5	-	-	-	-
r_4,21	11	-	-	-	-
r_4,8	-	21	-	-	8
r_4,9	-	11	-	-	-
r_4,14	-	-	-	-	2
r_5,12	4	-	-	-	-
r_5,13	3	-	-	-	-
r_5,11	6	-	-	1	-
r_5,4	-	-	2	-	-
r_5,1	-	-	-	1	-
r_5,3	-	-	-	1	-

Table 8.6: dep-st-GMM probability matrix.

	r_1	r_2	r_3	r_4	r_5
r_1,7	-	1	-	-	-
r_1,8	-	1	-	-	-
r_1,0	-	-	-	1	-
r_1,23	-	-	-	1	-
r_1,2	-	-	-	0.33	0.67
r_1,3	-	-	-	-	1
r_1,4	-	-	-	-	1
r_2,17	0.27	-	-	0.73	-
r_2,18	0.13	-	-	0.87	-
r_2,8	-	-	-	1	-
r_2,10	-	-	-	1	-
r_2,15	-	-	-	1	-
r_3,8	-	1	-	-	-
r_3,7	-	1	-	-	-
r_3,9	-	1	-	-	-
r_3,11	-	1	-	-	-
r_3,13	-	1	-	-	-
r_4,18	1	-	-	-	-
r_4,19	1	-	-	-	-
r_4,21	1	-	-	-	-
r_4,8	-	0.72	-	-	0.28
r_4,9	-	1	-	-	-
r_4,14	-	-	-	-	1
r_5,12	1	-	-	-	-
r_5,13	1	-	-	-	-
r_5,11	0.86	-	-	0.14	-
r_5,4	-	-	1	-	-
r_5,1	-	-	-	1	-
r_5,3	-	-	-	1	-

adjusted using weight values. This model is refereed to as: w-st-GMM.

In ind-st-GMM model, the space and time matrices are built separately as explained in the previous models (*i.e.*, s-GMM and t-GMM). Both space and time information contribute equally in making the decision of next RoI prediction. In this case, we first find the probability of each type occurring separately, and then multiply the probabilities. The probability matrix of this model is calculated based on the equation 8.6.

$$P(Nxt_r = r_j | Cur_r = r_i, Cur_t = t_i) = P(r_j | r_i) \cdot P(r_j | t_i) \tag{8.6}$$

Building w-st-GMM model is similar to ind-st-GMM except that the probability of each type is multiplied by weight value to make more balanced decisions on next

RoI prediction. Then, the final weighted probability values are summed as shown in equation 8.7.

$$P(Nxt_r = r_j|Cur_r = r_i, Cur_t = t_i) = W_r \cdot P(r_j|r_i) + W_t \cdot P(r_j|t_i) \tag{8.7}$$

where W_r and W_t are the weight variables in which their values belong to the interval [0,1] and $W_r + W_t$ is equal to 1. When one weight value is equal to zero, it means that the contribution of specific information is excluded from the prediction decision.

In dep-st-GMM, the space-time information is used by integrating both into one matrix. dep-st-GMM can be tailored as it follows. Everyday trajectory is first divided into 24 time intervals, each of which lasts one hour long. Then, for each RoI, a list of time intervals are extracted. The transitions and time matrices among RoIs are calculated. Time matrix represents different time intervals among regions (see Table 8.4 for a portion of user mobility data). Finally, transitions and time matrices are integrated into one matrix called Tran-Time matrix (see Table 8.5). For example, the number of movements from r_1 to r_2, r_4 and r_5 are 6, 33 and 4, respectively. From user's trajectories, we observed that the six times of movements from r_1 to r_2 were two main times at 7 o'clock and four times at 8 o'clock. The probability matrix of this model is calculated based on equation 8.8 as shown in Table 8.6.

$$P(Nxt_r = r_j|Cur_r = r_i, Cur_t = t_i) = \frac{N(r_i \rightarrow r_j|t_i)}{\sum_{r_k \in \mathcal{R}} N(r_i \rightarrow r_k|t_i)} + [\![N(r_i \rightarrow r_j|t_i) = 0]\!] * P(r_j|r_i) \tag{8.8}$$

where

$$[\![\text{ x }]\!] \rightarrow \begin{cases} 1, & \text{if x is ture.} \\ 0, & \text{otherwise.} \end{cases}$$

If the $N(r_i, r_j|t)$ is 0, the model will work according to only the transitions probability matrix $P(r_j|r_i)$ computed in equation 8.4. In this case, dep-st-GMM applies explicit feedback to automatically update the Tran-Time matrix by adding the new time to the region's time movements list.

8.2.3 Next Location Prediction

Before predicting the next location, user's current position (in real-time) or trajectories previously collected (in testing) are preprocessed. The GPS coordinates are converted into their RoI identifiers, while the timestamps are quantized into specified time intervals. The trajectory will be represented as a sequence of regions in case of s-GMM, and a sequence of time intervals in case of t-GMM. In case of st-GMM, the trajectory will be represented as a sequence of tuples (r_i, t_i) where r_i is the RoI and t_i is the time interval.

s-GMM and t-GMM are utilized to predict next RoI based on s-GMM and t-GMM probability matrix, respectively. Determining the next RoI depends only on the number of transitions among the RoIs. Specifically, the most frequently visited RoI is assigned the highest probability.

To make prediction using s-GMM, a set of probability values is obtained based on the current RoI. The RoI with the highest probability value is predicted to be the most likely next RoI of a certain user as shown in equation 8.9. Similarly, in t-GMM probability matrix, the RoI of the column with the highest value for the row that represents the n previous time interval(s) is retrieved (see equation 8.10).

$$s - GMM(r_i) = \underset{\hat{r} \in \mathcal{R}}{\arg\max}\, P(\hat{r}|r_i) \tag{8.9}$$

$$t - GMM(r_i) = \underset{\hat{r} \in \mathcal{R}}{\arg\max}\, P(\hat{r}|t_i) \tag{8.10}$$

where $\hat{r}$ is the predicted RoI and $\mathcal{R}$ is the set of discovered RoIs in a user movement region.

In all st-GMM models, time context is included to improve the prediction model. Determining the next RoI depends not only on the number of transition but also on the movement time. The RoI with the highest value is retrieved as the RoI the user will visit next as shown in equation 8.11.

$$st - GMM(r_i, t_i) = \underset{\hat{r} \in \mathcal{R}}{\arg\max}\, P(\hat{r}|r_i, t_i) \tag{8.11}$$

8.3 Experiments and Results

Various experiments are conducted to evaluate the performance of the proposed approach. The settings of the experiment, the datasets used and experimental results are explained in this section.

8.3.1 Datasets

The proposed models are evaluated using two publicly available real-world datasets, *i.e.*, GeoLife and Gowalla. In our experiments, GeoLife is preprocessed in order to discover the RoIs. Different RoIs' discovery methods are used in the experiments. The RoIs in Gowalla dataset are given different identifiers during collecting the data.

8.3.2 Experimental Settings

8.3.2.1 DRoI Method Specifications

To select the parameters specifications, a grid search method is implemented on one user trajectories. For DCRoI, three parameters are considered: the distance (δ), stay time (τ) and region density (rd). In this study those are: $\delta = 200$ meters, $\tau = 20$ minutes and $rd = 400$ points. For CRoI clustering using DBSCAN, two parameters are required: maximum distance ε between any two points and the minimum number of points $minPts$ required to form a dense region. In this study, those are: $\varepsilon = 300$ meters and $minPts = 3$. k-means algorithm is used with $k = 8$.

In order to evaluate the performance of various methods used to discover RoIs, trajectory dataset must be annotated with meaningful RoIs. The GeoLife dataset used in this study is not originally annotated. We asked 10 participants to manually annotate the RoIs into a set of semantically meaningful labels based on the density of trajectories and the real locations on the maps. Each participant annotated the RoIs for 20 randomly selected users' trajectory. These manually annotated outlying RoIs serve as ground truth in the experiments. Soundness and completeness properties are used as metrics to evaluate the effectiveness of the methods used to discover RoIs.

8.3.2.2 Prediction Models Specifications

In this chapter, first, second, third, fourth and fifth order GMM models are implemented. The trajectories' datasets of each user are split into 80% training and 20% testing set. The partition is repeated 10 times with different randomly selected trajectory files. The performance score of each case from each user is then calculated and the final results of all users are averaged. The weight values W_r and W_t are set to 0.8 and 0.2, respectively. To assess the efficiency of the prediction model, Recall score is used as a metric in all the experiments, as described in Section 2.5.1.

The proposed models are compared with baseline models in the task of predicting user's next location [1]: MC (Ashbrook and Starner, 2003), n-MMC (DJ Cluster is used for discovering RoI), HPHD (Gao et al., 2012), AR (Daoui et al., 2013; Kedia, 2012) and NN (Parija et al., 2013c; Leca et al., 2015; L. Vintan and Ungerer, 2004) . We selected those models in particular as based on using the same datasets we have used or based on their usage of MC prediction models.

Additionally, we compare our method of discovering RoIs with LiSPD (Li et al., 2008; Yuan et al., 2013) and k-means to show the different RoIs discovered by these methods. The same trajectories and parameters specifications are considered for these methods. For HPHD, AR and NN, k-means is used as a basic algorithm for extracting RoIs.

8.3.3 Comparison Results: DRoI Methods

Figure 8.7 illustrates a comparison of RoIs discovery methods in terms of soundness and completeness. High completeness value means that most of the significant locations are discovered, while the high soundness value means that most of the insignificant locations are ignored. The best soundness and completeness values are achieved by DCRoI+DBSCAN and LiSPD+k-means by 81% and 82%, respectively. Furthermore, it can be noted that the variance of completeness values is much smaller than variance of soundness values. Thus, soundness is more appropriate metric to

[1] We have implemented all baseline models based on our understanding of the original papers.

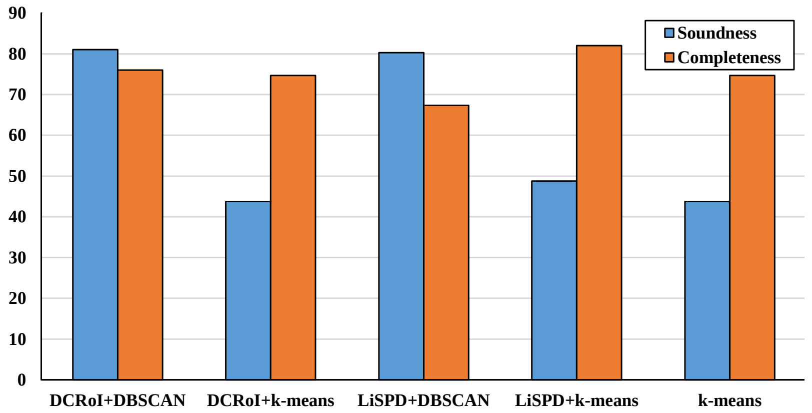


Figure 8.7: Soundness and completeness of RoI discovery methods.

evaluate the proposed methods. The highest soundness values are 81% and 80% for DCRoI+DBSCAN and LiSPD+DBSCAN, respectively. DCRoI is better than LiSPD when DBSCAN is used as a second level algorithm. Discovering many CRoIs in the first level using LiSPD negatively affects the clustering process of the second level. Additionally, it can be observed that using k-means results in high completeness value because of the large defined number of clusters ($k = 8$) which might be greater than the number of the real RoIs of some users. DBSCAN results in better soundness than k-means. Grouping in DBSCAN depends on the density of points with many nearby neighbours rather than the mean of all points that belong to the same cluster.

8.3.4 Comparison Results: Location Prediction

Table 8.7 illustrates the performance comparison on the two datasets in terms of Recall@N with N = 1 and 2. We can find that in the GeoLife dataset, the models where k-means is used to find the RoIs (*i.e.* MC, HPHD, AR and NN) do not perform well. The reason is that k-means clustering algorithm is not tailored for geolocated data where the grouping process depends on the mean of all points that belong to the same cluster. MC, HPHD and AR models have achieved approximately similar results under Recall@1. NN slightly improves the results comparing with MC, HPHD and AR, but still performs poor in terms of predicting the future RoIs. n-MMC improves the performance greatly comparing with the other baseline models. Extracting the RoIs using DJ cluster is better than k-means, where the number of

Table 8.7: Performance comparison on the datasets evaluated by Recall@N. Best scores are in bold.

Model	GeoLife		Gowalla	
	Recall@1	Recall@2	Recall@1	Recall@2
MC	58.46	66.91	15.3	21.1
n-MMC	68.6	-	-	-
HPHD	58.07	73.23	22.37	36.99
AR	59.32	67.43	21.23	36.29
NN	63.01	82.05	**34.13**	**48.44**
s-GMM	68.92	84.21	21.38	36.84
t-GMM	47.32	64.53	19.11	32.76
ind-st-GMM	66.93	78	20.42	37.48
dep-st-GMM	**70.52**	**88.4**	22.23	39.49
w-st-GMM	69.73	88.25	22.3	41.58

discovered RoIs depends on the people's mobility traces (Zhou et al., 2004).

The worst results are obtained by t-GMM when only time sequences are used in building the prediction model. In contrast, a great improvement that is brought by s-GMM which is considered as the best model among the compared ones. Using the space information results in higher prediction performance than time information. Additionally, using only the RoIs sequences (or only time sequences) are not enough inputs in location prediction as the models can not well capture the users' mobility behaviours. Including the time factor in ind-st-GMM did not cause any performance improvement. Using the probability of time transition matrix can change the final prediction decision when combining with the probability of space transition matrix. In other words, using time context as a basic component either individually as, in t-GMM or with space context, as in both ind-st-GMM and HPHD is not recommended. This is due to the different degrees of importance of space and time context when generating the information for location prediction (*i.e.*, space information is the most important one). Thus, the time context must be formulated in a way to add more information to the space context as in w-st-GMM. ind-st-GMM and HPHD models are similar in terms of using time information as a basic components in building the prediction model. However, they are different in terms of probability calculation where HPHD model depends on the

Gaussian distribution and therefore there is no probability that equals to zero as in ind-st-GMM model. w-st-GMM performs better than ind-st-GMM and HPHD. This is due to the importance of space information where it is given higher weight than time information. Moreover, we can observe that, dep-st-GMM outperforms the compared models on the GeoLife dataset measured by all the metrics. The superiority of dep-st-GMM can be attributed to using the method of RoIs discovery and the addition of time context information.

On Gowalla dataset, it is clearly shown that NN results in a better prediction performance than other models. While NN has the ability to learn non-linear and complex relationships, probabilistic models in general depend on distributional assumptions. On the other hand, NN is computationally intensive due to the large number of parameters that should to be updated numerous times. As a result, using smartphone devices with limited resources to train NN and make a prediction can be computationally expensive and time-consuming.

MC and n-MMC models are excluded from the comparison. They are based on the classical MC but with different clustering algorithms. Gowalla dataset is used without discovering the interest regions.

8.3.4.1 Effect of Different Orders

In order to explore the impact of different orders, we perform experiments with first, second, third, fourth and fifth order. Results are shown in Table 8.8. From the table, we can see that the prediction accuracies of dep-st-GMM and w-st-GMM are higher than others. On Gowalla dataset, HPHD achieved the best results when the order equals to 1. However, the prediction performance often decreases when the order is greater than 2 where more constraints must be applied. The best result on both datasets is achieved by w-st-GMM. It shows that the impact of RoIs sequences and time interval information are more significant on modeling user's behaviour than using only one of them. When n-order equals to 2, w-st-GMM achieved better performance with 73.86% and 24.17% on GeoLife and Gowalla datasets, respectively. In all experiments, second order results in a higher prediction performance than other

Table 8.8: Performance comparison with different order evaluated by Recall@1. Best scores are in bold.

Dataset	Model			Order		
		First	Second	Third	Fourth	Fifth
GeoLife	MC	58.46	64.65	63.89	63.72	63.67
	n-MMC	68.6	69.1	63.2	-	-
	HPHD	58.07	64.15	65.19	65.01	64.32
	s-GMM	68.92	72.53	70.96	69.41	66.45
	t-GMM	47.32	50.39	52.01	52.35	52.22
	ind-st-GMM	66.93	70.99	70.69	68.5	66.25
	dep-st-GMM	**70.52**	73.44	71.16	69.56	66.58
	w-st-GMM	69.73	**73.86**	**71.19**	**69.92**	**67.23**
Gowalla	MC	-	-	-	-	-
	n-MMC	-	-	-	-	-
	HPHD	**22.37**	22.4	20.36	17.7	14.89
	s-GMM	21.38	22.92	20.48	17.80	14.95
	t-GMM	19.11	19.66	14.47	11.21	9.73
	ind-st-GMM	20.42	22.47	20.19	18.13	17.53
	dep-st-GMM	22.23	24.06	22.41	20.66	17.64
	w-st-GMM	22.3	**24.17**	**22.64**	**20.74**	**17.65**

n-order.

8.3.4.2 Effect of RoI Discovering Methods

Discovering all locations that represent a real RoI to a certain user enables to build a real trajectory. Then, it helps the model to extract the user's mobility pattern which means more accurate prediction models. Therefore, the effect of RoIs discovering methods is explored.

As stated in section 8.2.1, discovering RoIs takes place in two levels of clustering. For the first level, DCRoI and LiSPD methods are used, whereas DBSCAN and k-*means* are used for the second level. For testing, different methods of discovering RoIs are considered: DCRoI+DBSCAN, LiSPD+DBSCAN, DCRoI+k-means and LiSPD+k-means. k-means is used also as one clustering level. The results are reported in Figure 8.8.

It is shown that when using the method we proposed for discovering RoIs (DCRoI+DBSCAN), the models perform better. This indicates that DRoI is effective in discovering the significant RoIs, which enables the models to extract

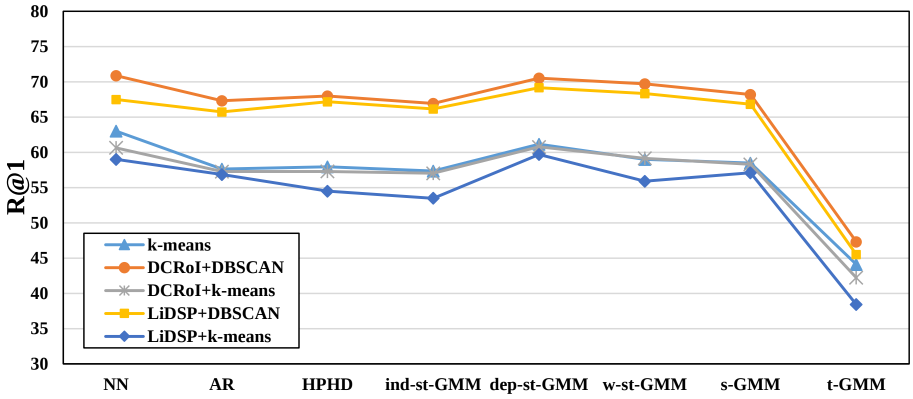


Figure 8.8: Prediction models with different RoIs discovery methods.

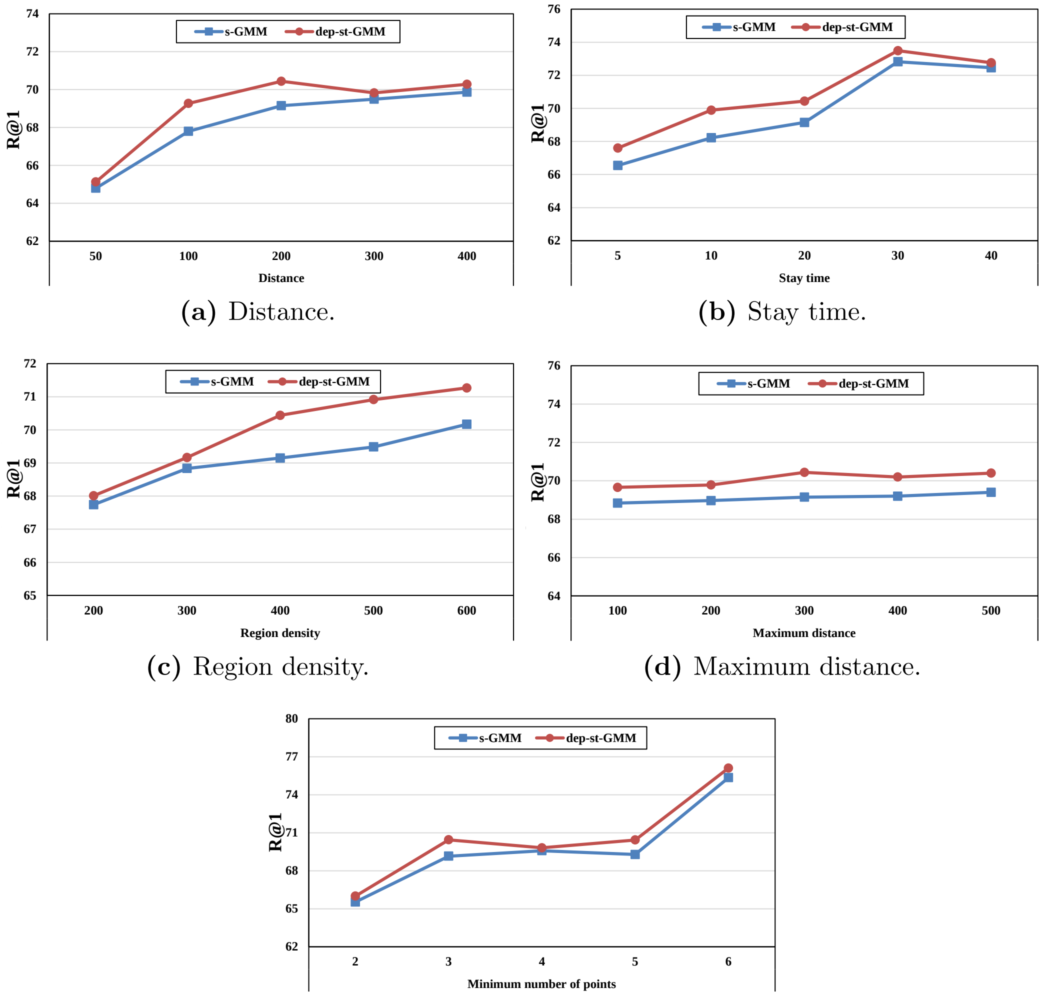
a user' mobility behaviour and as a consequence, it can be used to build more accurate models. DCRoI+DBSCAN method produces highest performance rate of the prediction model and then LiSPD+DBSCAN. The three other methods (*i.e.* DCRoI+k-means, LiSPD+k-means and k-means) have almost the same prediction performance rate. Using DCRoI in the first level improves the performance of the prediction models, better than LiSPD. Regarding second level, DBSCAN results in better performance than k-means.

Adding time factor to a real discovered RoI positively affects the prediction performance. When k-means is used, some significant RoIs could be missed. Also, some regions that do not carry semantic meaning (like 'roads') might be detected as a RoI. Thus, including the time to RoIs that represent insignificant locations will not provide a good prediction performance.

8.3.4.3 Effect of DRoI Parameters

We also study the impact of the five DRoIs parameters on the performance of the models: δ, τ, rd, ε and $minPts$. We conduct several experiments with various values as shown in Figure 8.9. We start by varying the value of one parameter while fixing the others and then studying how the prediction performance is affected. The same procedure is then repeated for the rest of the parameters.

In general, we observe that the models performance gives the best results as long as we increase the parameters values. Small values mean that many RoIs will be discovered which limits its efficiency in discovering significant regions and, hence,



(a) Distance.

(b) Stay time.

(c) Region density.

(d) Maximum distance.

(e) Minimum number of points.

Figure 8.9: The effects of DRoIs parameters.

impairing the prediction performance. On the contrary, large values mean that few RoIs with large areas will be discovered. Thus, it will be easy for the models to extract the movement pattern and then predict the next location.

8.3.4.4 Running Time

Finally, we measure the exact running time of the RoIs discovery methods and the prediction models. For the RoI discovery methods, we simply run the methods on trajectories of different users with different number of GPS points, and then measure the amount of time. For the prediction models, we run the models on trajectories of a randomly selected user and measure the amount of time for building the model

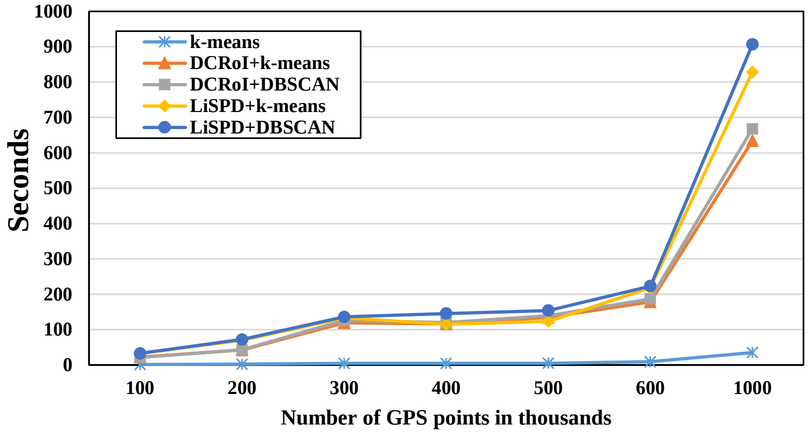


Figure 8.10: Running time of different RoI discovery methods.

Table 8.9: Running time in seconds.

Model	Model building	Predicting
AR	0.59±0.11	0.37±0.07
NN	1464.3±23.7	3.07±0.66
HPHD	24.74 ± 2.11	1.63 ± 0.24
s-GMM	0.19±0.03	0.36±0.03
t-GMM	0.17±0.02	0.43±0.12
ind-st-GMM	0.2±0.02	0.42±0.06
dep-st-GMM	0.24±0.01	0.42±0.1
w-st-GMM	24.80±1.90	0.48±0.01

and the predicting. For each model, the experiment is repeated 10 times, and the average and standard deviation are then calculated. All experiments were conducted on iMac, equipped with a 3.06 GHz Intel Core 2 Due CPU and 4GB RAM.

Figure 8.10 shows the methods' running time in seconds. As expected, running times grow gradually with the dataset size. We observe that our method is faster that LiSPD due to the less number of iterations. Of course, k-means is still significantly faster, but it is not tailored for geolocated data.

Table 8.9 shows the prediction models' exact running time in seconds. It can be noted that, during model building, our proposed models are less than others. During the prediction, all st-GMM models' running times are more than AR's due to including an extra factor (the time context). t-GMM is faster that s-GMM during model building but slower during the prediction phase due to the large number of time intervals compared to the number of RoIs for some users. However, NN model has the highest running time. This is not surprising as the NN model has a large number of parameters that need to be updated numerous times. It is worth mentioning that our models are faster than others for any dataset size.

8.3.4.5 Discussion

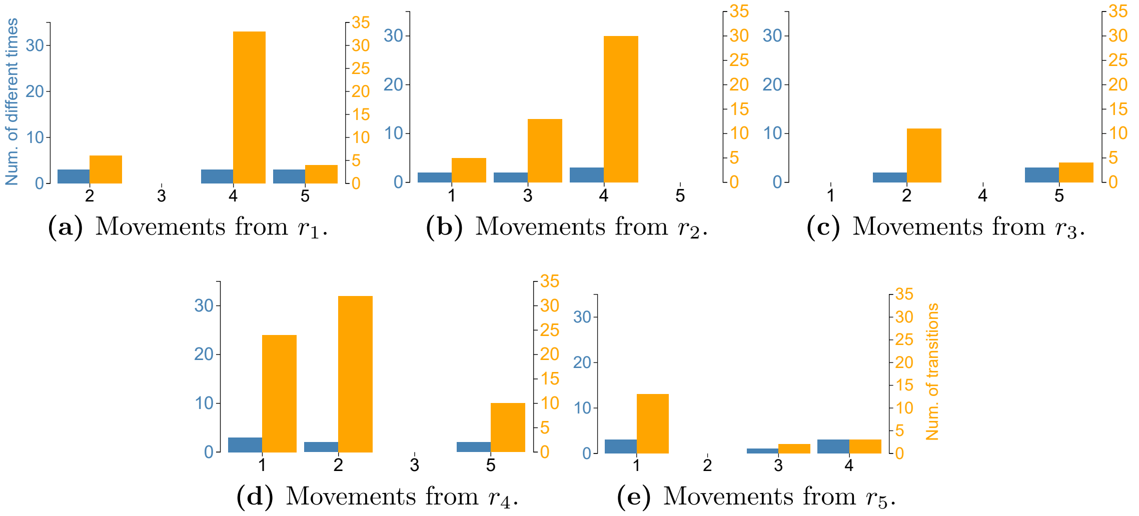
Overall results show that w-st-GMM outperforms most of the state-of-the-art models despite the fact that the model is simple. As the training set size increases (for example, 90% training and 10% testing), the performance is improved since the model can learn more (w-st-GMM achieved 74.7%).

Regarding the different evaluated orders, the prediction performance improves as the n order increases but then decreases gradually when $n > 2$ due to having more constraints that must be applied to get the next location. For example, the inputs of a fifth order model are the last five RoIs the user visited which are rarely visited or occurred in the same RoIs order. In first order Model, only the current RoI is considered. Thus, more constraints means less performance.

In addition to that, the impact of the time factor becomes little important when the order is high. For example, with first order, there is an obvious difference between the prediction performance of space-based and space-time-based models but the gap between them is narrowing with the increase of the order.

In general, it is possible for a user to change his/her mobility patterns over time. All the prediction models are unable to deal with new context of movements. In other words, the models fail to make a prediction for new locations in unknown territories (*i.e.* the unseen locations) that did not appear in the user's history. Applying explicit feedback to automatically update the transition matrix can be used as a simple solution to improve the prediction performance.

As indicated in the description of the prediction phase, the model retrieves the RoI with the highest probability value given the current RoI. In this regard, it is possible to have the same probability value for more than one RoI. In our model, we choose to retrieve the first one added to transition matrix.



(a) Movements from r_1.

(b) Movements from r_2.

(c) Movements from r_3.

(d) Movements from r_4.

(e) Movements from r_5.

Figure 8.11: User movement regularity.

Movement Regularity As shown in Figure 8.11, the movement regularity of a user can be identified based on the transition matrix (Table 8.3) and time matrix (Tables 8.4). For example, the number of transitions from r_5 to r_4 is equal to the number of movement times between them. This means that a user moves to r_4 in different times. Thus, there is no regularity in the movement between these two RoIs. Additionally, the RoIs that are visited by the user at a specific time can be extracted. For example, r_2 can be considered as a workplace because the number of transitions from r_4 to r_2 is much greater than the number of the different values of movement times.

8.4 Summary

This chapter presents an approach to discover RoIs in the users movement area and then predicts their future locations, which play a key role in the success of advanced location-based services. The RoIs discovery method presented in this chapter includes two levels. The first level is to group GPS points based on three threshold conditions: distance, time and region density, which identifies a set of CRoIs. The second level is to perform clustering using DBSCAN algorithm on the CRoIs to obtain the RoIs. Soundness and completeness are used as metrics to evaluate the proposed RoIs discovery method. We found that our method is effective compared to other relevant methods in the literature. While the method is able to find most of the RoIs, the method's overall execution time is less than other methods. Based on MC, a model for predicting a user's next location is proposed. Moreover, a general transformation function for the trajectory is used to include the space and time context. Any order MC will be processed as first order, which helps to make more abstraction on n-order. The latter performs better than space-based and time-based models, but the gap between the models' prediction performance is narrowing with the increase of the order. We evaluated the proposed approach with real-world datasets: GeoLife and Gowalla. Overall, the approach where DRoI and second order MC were used, achieved better performance.

CHAPTER 9

Monitoring Seniors with Mild Cognitive Impairments using Deep Learning and Location Prediction

9.1 Introduction

The aging of the population is one of the most important challenges for public healthcare sector in the 21st century. MCI is one of the most prevalent diseases suffered by the seniors. People suffering from MCI (*i.e.* patients) may forget their destination while moving from one area to another. As a result, strange trajectory patterns are obtained and the so-called *wandering* episodes occur. Fortunately, technology allows their movements to be continuously tracked and hence, appropriate real-time assisting services to address their difficulties in navigational tasks can be provided. Otherwise they may get lost, which can cause serious injuries or even their death.

GPS and navigation applications are key enablers to many services offered by mobile devices, from state-of-art products to low-cost smartphones. As a result, tracking and monitoring systems have emerged as a good solution to assist elderly people during their outdoor mobility issues. In such systems, the patient makes use of a wearable or a smartphone that is capable of obtaining its location (using GPS technology). In addition, the system is able to warn patients or to send alarms to caregivers. Monitoring systems contribute to self-care, and reduce stress on patients' relatives and caregivers as well. Moreover, cognitive environments and smart cities pave the way to the deployment of advanced assistance services for seniors in the smart health and cognitive health paradigms (Solanas et al., 2014a).

Some of the existing tracking and monitoring systems require the patients to interact with smartphone application in a variety of ways (*e.g.*, checking in to their destination (Hossain et al., 2011), pushing a panic button (Rodríguez et al., 2012) or selecting the destination area (Wan et al., 2011)). However, monitoring systems should take into account the inability of some patients to interact with their devices. Hence, it is essential that monitoring systems nothing is required from the patient except to carry the smartphone.

It is commonly assumed that elderly people follow regular mobility routines, *i.e.* they visit the same locations and walk through the same routes from one area to another. This fact makes it possible to detect the abnormal movements. Notwithstanding, most of the proposals are unable to work well unless enough mobility data are available. For example, to detect the pacing pattern in the wanderer movement, he/she must move back and forth between two points or more. However, the abnormal situation should be discovered in its first appearance to prevent the person from getting lost in advance.

At initial stages of dementia and other cognitive deficits due to age-related memory loss, the system can be used to provide information related to early disease diagnosis by assessing their movement behaviour in case of abnormal movements.

We propose a monitoring system that can assist elderly people during their outdoor movements. The system contains a model, called ST-CNN, which runs CNN in a server utilizing the senior's historical movement data. The model is responsible for predicting the locations that the elder will visit. Based on that, the route and the expected time spent to reach that locations are obtained. In our system, all information related to the movement should have a key role in order to identify erroneous behaviours. During a movement, the caregivers are warned in case the patient spends a long time to reach one of the predicted locations or moves to unpredicted locations by changing the routes or still moves in the same area. Furthermore, we demonstrate how the abnormal movements can be detected and how the system assists the elderly people to be safe in real-time movement scenarios. It is based on a model called ABD that takes advantage of RNN to analyse time and space related variables. In order to evaluate the system, three different datasets are used, each one of them with its own descriptors. First, outdoor trajectories from Catalan patients diagnosed with MCI are used. Second, two additional online datasets, which contain trajectories from individuals (not necessarily suffering from mild cognitive impaired) are also used; GeoLife and OpenStreetMap datasets.

The system is autonomous and hence, no explicit input from the user is required. The system is able to learn about a user's movement behaviour. With the aim of minimising patient interventions or interactions with the monitoring system, our proposal generates the predicted destinations and detects the abnormal behaviour based on deep learning models. In our system, the monitoring is performed in real-time and the abnormal behaviour is immediately detected.

9.2 *SafeMove*: System Description

In this section, first we overview our proposed system and then we describe with details all system components.

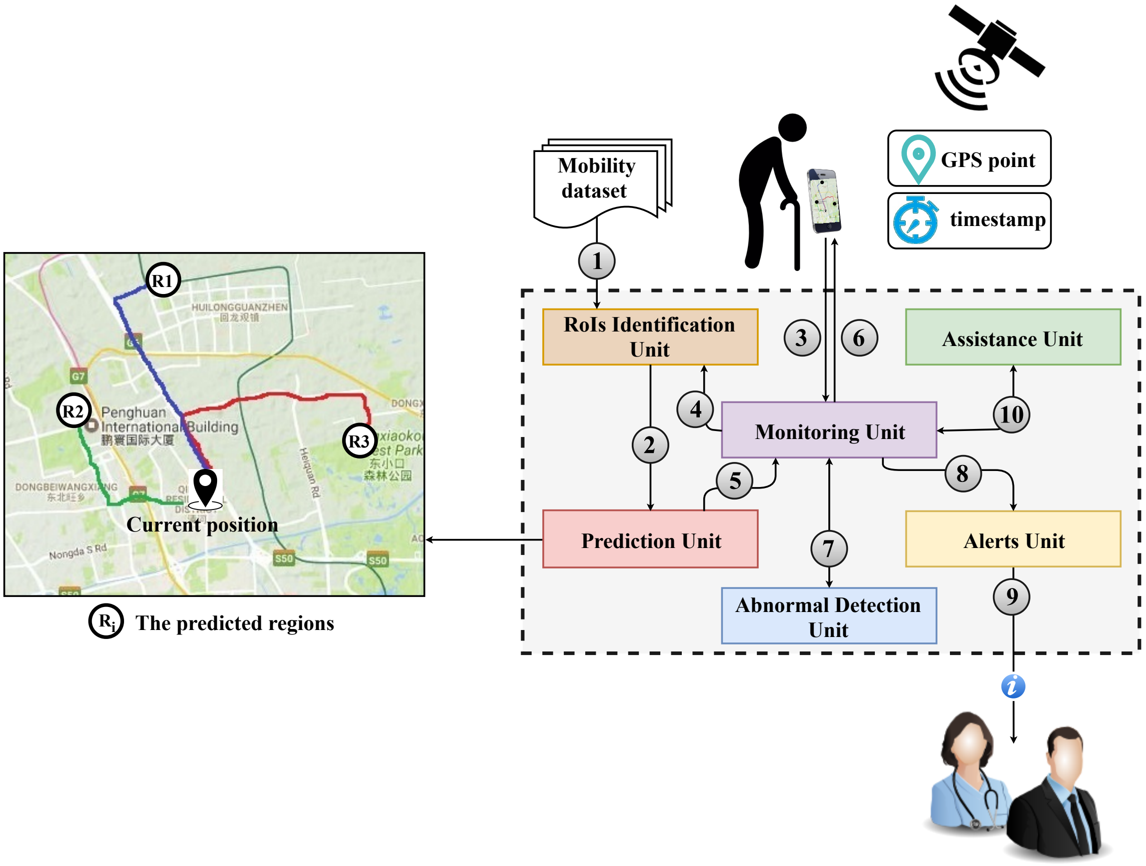


Figure 9.1: *SafeMove* system architecture.

9.2.1 Overview of *SafeMove*

The overall architecture of the system is shown in Figure 9.1. The system contains seven parts:

1. **Patient's smartphone application**: is responsible for sending the data related to the patient's positions to a remote server and displaying the outputs of the system.
2. **Monitoring Unit**: is the main part in *SafeMove* which is responsible for receiving and distributing the data from and to the different system parts. Moreover, each patient movement is analysed in this unit to determine whether or not he/she is located in a safe area.
3. **Prediction Unit**: is the part that is responsible for predicting people's mobility.
4. **RoIs Identification Unit**: detects the significant RoIs in the patient

movement area and provides mainly the RoI identifier that represents those regions.

5. **Abnormal Detection Unit**: is the part that is responsible for detecting the abnormal movement of the patient.
6. **Alert Unit**: is responsible for sending notifications to the relatives or caregivers.
7. **Assistance Unit**: is responsible for helping elderly people until reaching the desired and safe destination.

The overall functioning of the system is as follows:

1. Mobility data previously collected is converted from GPS coordinates into discrete values associated to specific RoIs, in the RoIs Identification Unit.
2. The output is then sent to the Prediction Unit, where each patient has a different prediction model.
3. The Patient's Smartphone Application provides the Monitoring Unit with the current GPS coordinates and the timestamp through the available Internet connection.
4. After receiving patient current location (GPS coordinates and the timestamp), the Monitoring Unit sends these information to the RoIs Identification Unit to provide the RoI identifier, and then to the Prediction Unit to predict the next RoIs.
5. Next, 'N' RoIs are sent to the Monitoring Unit which in turn finds the routes and computes the expected time spent to reach that RoIs .
6. The Monitoring Unit forwards back all the system output information to the Patient's Smartphone Application for displaying through the user interface (will be available for reference for the patient).
7. Every time threshold (5 seconds, for instance), the Monitoring Unit continuously receives the patient current location from Patient's Smartphone Application. Then, each patient movement is analysed to detect the abnormal behaviour using the Abnormal Detection Unit.
8. In case of an abnormal movement behaviour, warning notifications are sent to

the Alerts Unit.

9. The Alerts Unit will notify the relatives or caregivers by sending alerts.
10. Finally, the Assistance Unit is activated to help the elderly patient to reach a safe RoI.

Next sections cover with details the individual components. We specially focus on the Monitoring and Abnormal Behaviour Detection units.

9.2.2 Patient's Smartphone Application

Due to the widespread usage of smartphones, they have been used as our client hardware. The system is designed to run on Android-enabled devices with GPS in mind. Collecting data from the device's sensors such as GPS location and timestamps can run as a background service. The system uses the built-in location technologies of the smartphone for location purposes without any mobile user interaction.

9.2.3 RoIs Identification Unit

This is the part of the system that is responsible for mapping each GPS point into the matching region, using RoI identifiers. We convert the sequence of GPS points of each user mobility data into a sequence of RoIs by detecting the *significant RoIs*. To detect the significant RoIs, we use the algorithm proposed by Li et al. (2008) described in Section 3.2.2.

9.2.4 Prediction Unit

This the part is responsible for analysing the elder's mobility data previously collected in order to predict the most popular RoIs his/her might visit in the next time from the current RoI. We use ST-CNN prediction model described in Chapter 7.

9.2.5 Monitoring Unit

Monitoring unit is the core component that is used to track and monitor movements, in addition to distributing the data from and to the different system parts. Due to the necessity to take action as soon as possible, the system must work in real-time. Therefore, every time the system receives a new GPS location, the monitoring unit is executed to ensure that the patient behaves normally.

There are four system statuses: *no moving*, *predicting*, *moving* and *stop*. The system status is set to *no moving* to indicate that the patient is indoor such as at night or inside a building. The patient movement can be detected using the GPS and accelerometer sensor (Xia et al., 2014b). Therefore, when the patient is moving, the patient's smartphone application will send the current position to the system. Then, the monitoring unit will operate and forward the current position to the prediction model. Thus, the system status is set to *predicting*. After user progress to one of the predicted RoI, the system status is set to *moving* to indicate that there is no need to predict and every patient movement will be monitored. After reaching to one of the predicted RoI, the system status is set to *no moving*. The status property is set to *stop* where the system will stop working in case the patient in a holiday with his/her relatives, for instance.

During the movement from a region to another, the state of a user generally changes from indoor to outdoor and vice versa. The abnormal behaviour could happen in the outdoor environment. In this paper, we use the term "sub-trajectory" to refer to the movement between two regions. Each sub-trajectory starts from the first GPS point in the first region until the first GPS point belongs to the next region. Given a trajectory, we divide it into sub-trajectory and each one is labeled as normal or abnormal.

Martino-Saltzman et al. (1991) investigated travel patterns of nursing home residents with dementia. Four different travel patterns were identified: direct, random, pacing and lapping. In addition to that, we add two different patterns related to the predicted RoI: indirect and stopping patterns, Figure 9.2.

The movement behaviour of a person is considered as abnormal if one of the

Algorithm 4: *SafeMove* system.

```
Input  : Traj: a sequence of GPS coordinate and Time, $Dis\_threshold$,
         $Length\_threshold$.
Output: Normal or abnormal behaviour.
$First_{GPS} \leftarrow Traj.GPS[1]$ // The first GPS coordinate
$First_{Time} \leftarrow Traj.Time[1]$ // The first movement time
$First_{loc} \leftarrow$ RoIs_identification($First_{GPS}$) // To get the identifier of the
  current RoI
$PL_1, PL_2, PL_3 \leftarrow$ ST-CNN($First_{loc}$, $First_{Time}$)
$Rot_1, Rot_2, Rot_3 \leftarrow$ getRoute($First_{GPS}, PL_1, PL_2, PL_3$)
$Dis_1, Dis_2, Dis_3 \leftarrow$ getDistance($First_{GPS}, PL_1, PL_2, PL_3$)
$ET_1, ET_2, ET_3 \leftarrow$ getExpectedTime($Dis_1, Dis_2, Dis_3, 3.1$)
for $i \leftarrow$ 2 to Traj.length do
    $directions \leftarrow$ getBearingValue($Traj.GPS[i-1], Traj.GPS[i]$)
    $SpentDis \leftarrow$ getSpentDis($First_{GPS}, Traj.GPS[i]$)
    $\bar{Dis}_1, \bar{Dis}_2, \bar{Dis}_3 \leftarrow$ getDistance($Traj.GPS[i], PL_1, PL_2, PL_3$) // Updated
      distances
    $SpentTime \leftarrow$ getSpentTime($First_{Time}, Traj.Time[i]$)
    if $direction.length = Length\_threshold$ then
        if ABD_DetectionModel($directions, Dis\_threshold$) == True then
            return abnormal
        end
    end
    if $SpentDis > Dis_1, Dis_2, Dis_3$ then
        return abnormal
    end
    if $\bar{Dis}_1 > Dis_1$ and $\bar{Dis}_2 > Dis_2$ and $\bar{Dis}_3 > Dis_3$ then
        return abnormal
    end
    if $SpentTime > ET_1, ET_2, ET_3$ then
        return abnormal
    end
end
```

following constraints are satisfied, Figure 9.2:

1. Time spent to reach the predicted RoIs is more than the expected time.
2. Taking the opposite direction for all predicted RoIs.
3. The trajectory is random, pacing and lapping.

Note that the patient tends to forget the location and as a result, the patient may spend a long time to reach one of the predicted RoIs. The patient may also change the direction to unpredicted RoI or may wander in the same area.

The main steps of the *SafeMove* system are described in Algorithm 4 that takes a sequence of GPS points either from patient's mobility data previously collected or in real-time. The RoI identifier of the current location is obtained by calling RoIs identification unit (line 3). Based on the obtained RoI identifier at a certain time, the ST-CNN model will predict 'N' RoIs (line 4), three RoIs are specified in the algorithm. Based on the predicted RoIs, the routes, the distances and the expected times to reach that RoIs are obtained (line 5 and 7).

After user progress on a route, each user movement is analysed. Three different values are evaluated during the movement: direction, distance and time. The direction is computed between the current GPS point and the previous one (line 9). The spent distance is computed between the current and the first GPS points (line 10), while the updated distances are computed between the current GPS and the predicted RoIs (line 11). Finlay, the spent time is computed between the current time and the first GPS point time (line 12). The distance is computed using Haversine Distance, Equation 8.1.

After that, the obtained spent distance is compared with the distances to the predicted RoIs (lines 18-20) which can be used to detect the change in the direction. If the spent distance value is increased, it means that there is a change in the direction. The updated distances are compared with the distances to the predicted RoIs (lines 21-23) to know exactly the RoI the patient is very close to. However, if the updated distance still almost similar, it is an evidence of stopping pattern.

The obtained spent time is also compared between the expected time to reach the predicted RoIs and the time spent from the first movement until the current

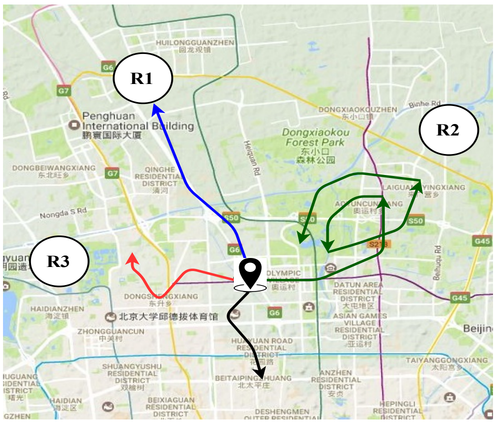
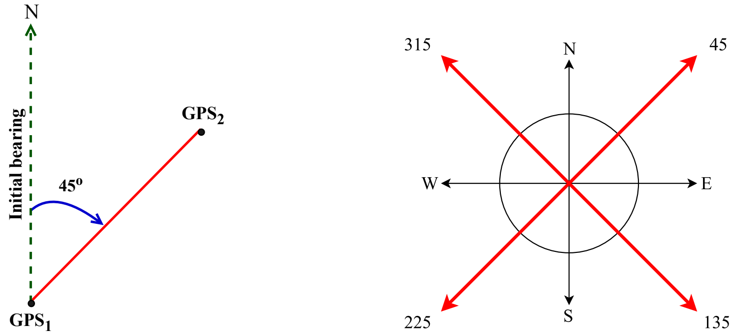

R1
R2
R3
Penghuan International Building
Dongxiaokou Forest Park
OLYMPIC
DATUN AREA RESIDENTIAL DISTRICT
ASIAN GAMES VILLAGE RESIDENTIAL DISTRICT
HAIDIANZHEN
ZHONGGUANCUN
TAIYANGGONGXIANG
Beijin

Figure 9.2: Movement behaviour types.



(a) Bearing.

(b) Movement directions.

Figure 9.3: Bearing and directions

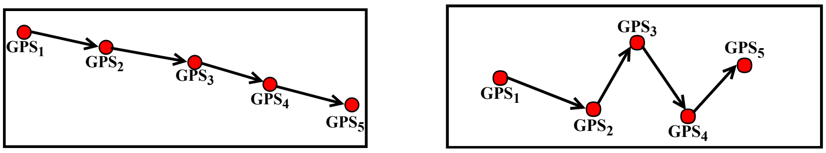
moment (lines 24-26).

Regarding lines 13-17, they are related to the abnormal detection unit which is described in the next subsection.

Meanwhile, the model updates the next RoI and the route by matching the current route with all other predicted routes taking into account that the patient can take different routes to the predicted RoIs. Also, the new position is compared against the predicted RoIs in order to know exactly the RoI the patient is very close to. During the movement, the patient is represented in a geographical map, thus, relatives or caregivers can keep track of the patient's progress continuously.

9.2.6 Abnormal Detection Unit

This part is responsible for detecting the abnormal behaviour in the elder movements that can be occurred due to the random, pacing and lapping pattern. For the sake



(a) Direct.

(b) Random.

Figure 9.4: Changes in directions

of brevity, such variety of travel pattern will be referred to as random.

In order to detect the random pattern, we use the Bearing Angle formula that used to find the direction between two consecutive GPS locations. Bearing can be defined as an angle or direction between the north-south line of earth (initial bearing) and the line connecting two GPS points, see Figure 9.3a. Given two GPS locations: $G_1(\phi_1, \lambda_1)$ and $G_2(\phi_2, \lambda_2)$, the bearing is computed using Equation 9.1:

$$Bearing = atan2\left(\sin\Delta\lambda \cdot \cos\phi_2, \cos\phi_1 \cdot \sin\phi_2 - \sin\phi_1 \cdot \cos\phi_2 \cdot \cos\Delta\lambda\right) \quad (9.1)$$

where ϕ is latitude, λ is longitude and $\Delta\lambda$ is the difference in longitude. The output of this formula is a value between 0 and 360.

As shown in Figure 9.3b, we have four intervals that describe the four directions a user could move towards: (45-135), (135-225), (225-315) and (315-360 and 0-45). These intervals are normalized in order to obtain the similar corresponding directions. We used four values: 1 (45-135), 2 (135-225), 3 (225-315) and 0 (315-360 and 0-45). If the normalized bearing values belong to the same interval, it means that the points are located in the same line Figure 9.4a, otherwise, the points are in different lines Figure 9.4b.

From the above analysis, we can see that the change in the bearing value intervals can serve as an indicator for random or abnormal behaviour. However, human behaviour mobility is essentially contains change in direction in the daily life. In order to distinguish an abnormal from normal behaviour, we add two threshold variables. The first one refers to the number of GPS readings that the system should wait before the final decision (*i.e.* normal or abnormal behaviour). The second one is length of the trajectory being evaluated (*i.e.* the distance between the first and

Table 9.1: Examples of pattern evaluation.

Direction sequence	State
1,1,1,2,2,2,1,1,3,3,3,3,2,2,2,0,0,0,0,0	Normal
0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0	Normal
1,0,0,0,0,0,3,3,3,3,3,3,2,2,2,0,0,0,0,0	Normal
1,1,2,2,1,1,2,1,3,0,3,0,3,0,0,2,2,0,0,2	Abnormal
1,2,3,0,1,2,3,0,1,2,3,0,1,2,3,0,1,2,3,0	Abnormal
1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,2,1,2,1	Abnormal

last GPS points of that trajectory should be greater than a distance threshold). In this paper, we consider a sequence of 20 GPS points within 100 meters as abnormal behaviour. Those values can be adjusted to meet individual needs.

If the monitoring system read the patient position every 5 seconds, 20 GPS points will be read within almost 1.6 minutes. This is means that if we have four different direction values, then number of different possible sequences are $4^{20} = 1,099,511,627,776$. Therefore, in order to train the abnormal detection model, we build a dataset which contains that number of records. Then, we manually label a portion of that records with normal or abnormal behaviour, see an examples in Table 9.1. Then, we used the labeled dataset to train RNN model in order to label the remaining unlabeled records. Finally, we used all dataset records to train ABD model using RNN as follows.

The user directions are represented as a sequence of values $\mathcal{S} = \{s_1, \ldots, s_n\}$, where $s_i \in \{0, 1, 2, 3\}$ and n is the length of the user's directions. Given a user u with a sequence of directions, the model classifies user's behaviour as normal or abnormal.

Figure 9.5 shows the graphical illustration of the model. The model contains an input, embedding, recurrent and classification layers as well as inner weight matrices.

The input layer consists of one vector $s_i \in \mathbb{R}^N$ which represents the direction value where N is the number of different direction values. This vector is encoded using one-hot encoding then passed through an embedding layer to produce a vector with d dimension. If the number of direction values is N and the dimensionality of the embedded vector is d, then the dimensionality of the embedded matrix $\mathcal{S}e$ is $N \times d$ where $\mathcal{S}e$ represents a set of direction values. The embedded vector $se_i \in \mathbb{R}^d$

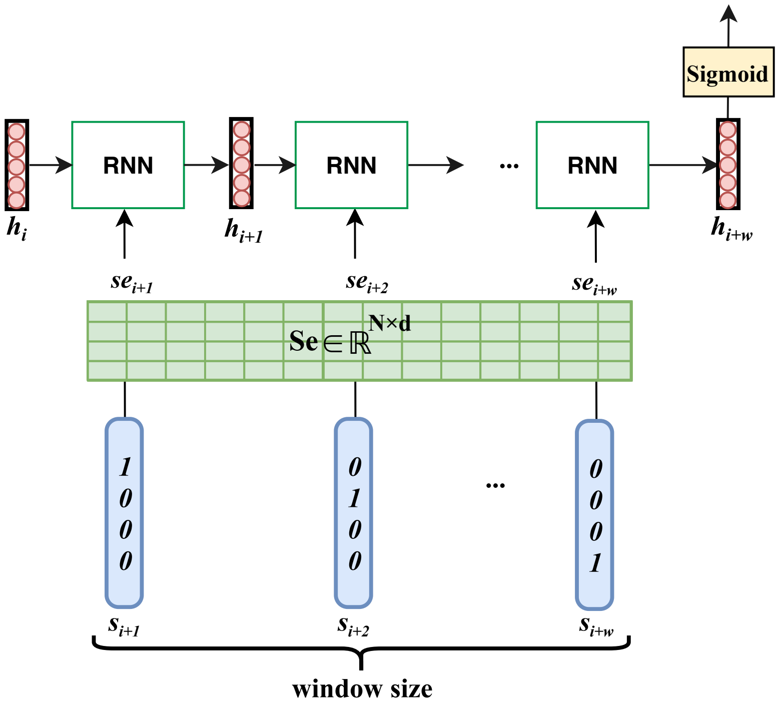


Figure 9.5: Abnormal behaviour detection model.

is given by multiplying the embedded matrix $\mathcal{S}e$ and the input vector s_i.

$$se_i = s_i \cdot \mathcal{S}e \tag{9.2}$$

The values of the recurrent layer $h_i \in \mathbb{R}^{d_h}$ are computed as below where d_h is the dimensionality of the recurrent layer vector:

$$h_i = f\left(se_i \cdot W_s + h_{i-1} \cdot W_{h_{i-1}} + b_h\right) \tag{9.3}$$

where $W_s \in \mathbb{R}^{d \times d_h}$ and $W_{h_{i-1}} \in \mathbb{R}^{d_h \times d_h}$ are the weight matrices and $b_h \in \mathbb{R}^{d_h}$ is the hidden layer bias. Hyperbolic tangent is used as the non-linear activation function for the recurrent layer.

The classification layer $\hat{y}_i \in \mathbb{R}$ produces a scalar value ranges from 0 to 1. The value more than 0.5 means normal class while less than 0.5 is abnormal class. Its values are computed as:

$$\hat{y} = g(h_i \cdot W_h + b_o) \tag{9.4}$$

where $W_h \in \mathbb{R}^{d_h \times N}$ represents the weight matrix between the hidden and output layers and $b_o \in \mathbb{R}^N$ is the output layer bias. The classification layer is a *Sigmoid* layer which is suitable for this case.

$$g(\hat{y}) = \frac{1}{1 + e^{-\hat{y}}} \tag{9.5}$$

Optimization is performed using Adam update rule and BPTT. The model parameters are $\theta = [Se, W_s, W_{h_{i-1}}, W_h, b_h, b_o, h_0]$, where h_0 is the initial vector for the recurrent layer. The cost function used is the cross entropy which is defined as:

$$J = -\sum_{i=1}^{n} y_i \cdot log(\hat{y}_i) \tag{9.6}$$

where n is the number of training samples, y is the real user behaviour, and $\hat{y}$ is the classified one.

9.2.7 Alert Unit

When the system detects patient's abnormal movement behaviour, it will notify the relatives and caregivers by sending an alerts accompanied by the patient current location (GPS coordinates).

9.2.8 Assistance Unit

This unit is responsible of providing the way that keep disoriented patient safe. It can help by displaying a map on their phone and creating a routes towards the nearest predicted RoIs or any RoI stored in the patient's previous history including the starting RoI of the movement. Then, the patient follow a series of navigation instructions sounds. The monitoring system will keep track the patient until reaching the desired destination.

Whilst different assistance means can be used, the most important issue is the ability of the patient to understand that assistance type, otherwise, it will be useless. Thus, the assistance type must be added to the system based on what the patient wants. This is one aspect that support the idea of customizable system.

9.3 Experiments and Results

In this section, we aim at demonstrating the performance of the deep learning core of our *SafeMove* system, by conducting tests using three real-world datasets.

9.3.1 Datasets

There are many online trajectory datasets, but finding datasets containing trajectories for elderly people with/without abnormal movement behaviour is not a straightforward task (Martínez-Ballesté et al., 2018). We use three real-world datasets in our experiments, *i.e.*, SIMPATIC, GeoLife and OpenStreetMap.

Due to the huge number of trajectories that each dataset contains, we evaluate a small subset of trajectories for each dataset. For SIMPATIC dataset, we select normal and abnormal trajectory and label them manually. We evaluate 496 trajectories, 417 of them presenting some kind of abnormal behaviour. Additionally, we can observe that SIMPATIC dataset contains shorter trajectories (elderly does not use to walk large distances), while GeoLife dataset is longer and with more GPS locations due to its high sampling rate. For OpenStreetMap datasets, we chose 16 individuals' GPS traces as our test datasets, while all individuals' GPS traces of GeoLife dataset are used. Since the datasets are not of elderly people, they do not contain abnormal movement patterns. Thus, in order to test the performance of the monitoring system in detecting abnormal patterns in trajectories, we added several trajectories with abnormal patterns manually.

9.3.2 Experimental Settings

We compare our detection model performance with an outstanding proposal in the literature. θ_WD (Lin et al., 2012) is method to determine wandering patterns by searching sharp changes of directions along their GPS traces.

The most important step is to determine the most suitable parameters of that model in that dataset. Since detecting abnormal behaviour depends on the dataset, the threshold depends on it too. The parameters of our model are as follows: the window size w is set to 20. The dimensionality of the embedded vector of the direction d and the hidden layer d_r are 50 and 20, respectively. Different distance values are used $\delta \in \{10, 50, 100, 150, 200\}$.

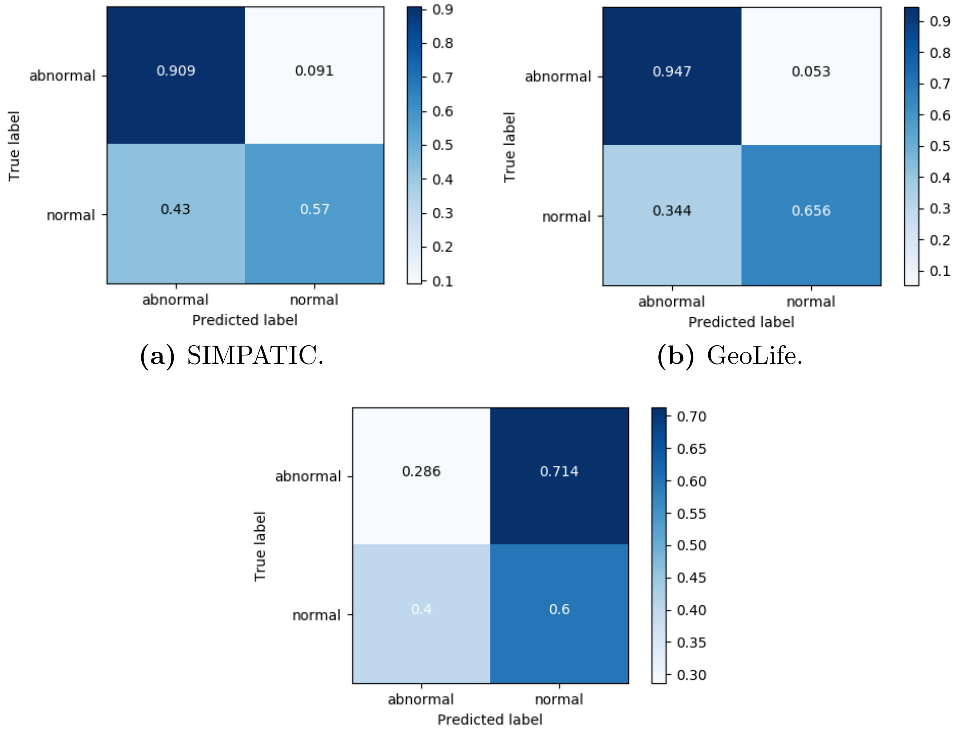
We validate whether the classification is correct, by comparing the prediction of the system monitoring and the trajectory's label. For this reason, we apply a binary classification. Moreover, we consider some statistical measures that can be derived from the confusion matrix after classifying each trajectory such as: Recall, Precision, Specificity, F1-score and Accuracy.

9.3.3 Results and Analysis

Figure 9.6 shows the confusion matrix obtained from our model classification procedure on the three datasets where main diagonal values represent the correct classifications, whereas off diagonal values are incorrect classifications. It is clear from the figure that our model is able to detect the abnormal trajectories more than the normal ones. For instance, the confusion matrix in Figure 9.6a shows that more than 90% from the abnormal trajectories were classified correctly, while the model mostly misclassifies normal trajectories as abnormal. This could be because of the shorter trajectories with low sampling rate.

For GeoLife dataset (9.6b), the biggest confusion happened when the trajectories are normal but classified as abnormal by the model. Since the GeoLife trajectories contain a huge number of GPS points, they have been classified as abnormal when detecting a change in the direction with a small distance value. On the contrary with the trajectories of OpenStreetMap dataset, our detection model needs more direction changes within a certain distance value to ensure abnormal behaviour.

The confusion matrix in Figure 9.6c shows that our model does not work correctly for OpenStreetMap dataset. This is due to that the sampling rate for location acquisition is not fixed and the distances between the consecutive GPS points are



(a) SIMPATIC.

(b) GeoLife.

(c) OpenStreetMap.

Figure 9.6: Confusion matrix of evaluating ABD model on the datasets. Each entry in column c and row r represents the percentage of action r that was classified to be action c.

Table 9.2: Performance comparison. Best scores are in bold.

Dataset	Model	Recall	Precision	Specificity	F1-score	Accuracy
SIMPATIC	ABD	**0.907**	**0.918**	**0.57**	**0.912**	**0.853**
	θ_WD	0.605	0.828	0.356	0.699	0.564
GeoLife	ABD	**0.947**	0.734	0.656	**0.827**	0.802
	θ_WD	0.784	**0.867**	**0.879**	0.823	**0.831**
OpenStreetMap	ABD	**0.286**	**0.25**	0.6	**0.267**	0.5
	θ_WD	0.143	0.2	**0.733**	0.167	**0.545**

almost high. When a low location acquisition rate is used together with long distances values between the consecutive GPS points, a huge number of abnormal trajectories will be missed. In addition to that, since this dataset is imbalanced, most of the correct classification are normal behaviour trajectories.

These results could be improved by modifying the values of the model parameters (*i.e.*, number of GPS readings and distance between the GPS points). Additionally, training the model with more data would allow for better generalize on its classifications and then improve the performance. In spite of this, the proposed detection model achieved good performance.

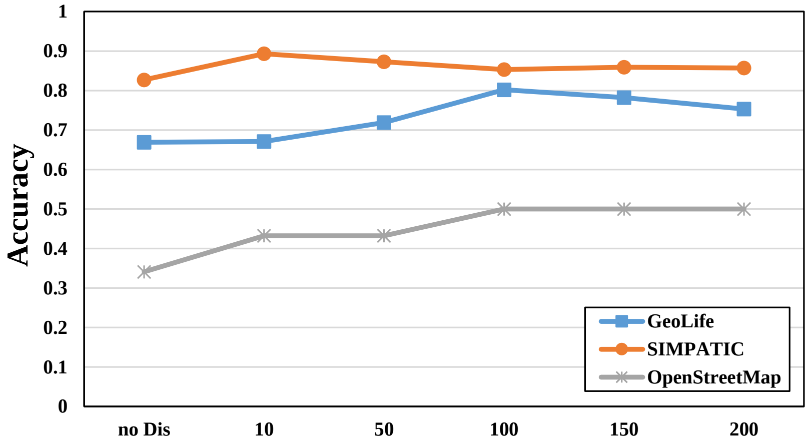


Figure 9.7: Distance effect on the datasets.

Table 9.2 illustrates the classification results of our model and θ_WD in terms of Recall, Precision, Specificity, F1-score and Accuracy. Focusing on the recall attribute, we see that our model detects up to 90% of the abnormal trajectories on SIMPATIC and GeoLife datasets. However, when the model detects abnormal, it is true with 73% and 91% of the cases on SIMPATIC and GeoLife datasets, respectively. Regarding OpenStreetMap, we can clearly see that our model is not appropriate for this dataset where recall is 28.6% and 25% from all abnormal predictions are detected. As mentioned before, any individual can upload a personal trace to the OpenStreetMap public repository. This means that the traces can contain trajectories using transportation means and sport activities which cannot be used for abnormal behaviour detection.

Regarding θ_WD, it is hard to detect abnormal trajectories but, when it happens, the trajectory is often abnormal with 82.8% and 86.7% on SIMPATIC and GeoLife datasets, respectively. We observe that the poor results are on OpenStreetMap dataset, since it has a low recall value (20%). Moreover, we conclude that this method performs well when GeoLife is used. Consequently, we could infer that this method works better when analysing trajectories with high sampling rate.

To investigate the impact of the distance parameter, we conduct several experiments to check the detection model performance with various distance values as shown in Figure 9.7. The best performance is obtained when the distance parameter value is 10 for SIMPATIC. However, the length is increased up to 100 and 150 in GeoLife and OpenStreetMap, respectively. This is due to the short trajectories of SIMPATIC compared to the trajectories of other datasets. Since GeoLife and OpenStreetMap trajectories are more dense (low sampling rate) than SIMPATIC, they are more likely to contain cycles.

9.4 Summary

We have presented a system called *SafeMove* which utilizes deep learning techniques and prediction model to provide cognitive assistance to elderly people. It relies on the historical mobility data as a basis for predicting likely locations and detecting abnormal behaviour. The system in the server runs a CNN utilizing an elder historical movement information in order to learn his/her movements. It is responsible for predicting the locations that he/she will visit, the route and the expected time spent to reach that locations. Moreover, three different variables are evaluated during the movement: distance, direction and time. We then developed a model called ABD that take advantage of RNN to detect the different abnormal behaviours scenario in real-time.

The success of the monitoring system depends mainly on the accuracy and availability of mobile user's location information. In addition to that, the quality of the Internet communication between the user's smartphone and the server is essential for the continued operation of the monitoring system.

Regarding the amount of data transferred between patient's smartphone and the server, it can be noticed that a considerable connections are to be used, which means more power consumption.

A major drawback of this system is that the used dataset for building the prediction model must be collected in the period when the patient had no problem with the movement at all. On the other hand, the question that might be asked is: How long should movement histories of a patient be stored and used to improve prediction accuracy? Another drawback can occur when the historical movement data for the patient is not available. A simple solution to overcome this drawback is to manually determine the significant RoIs by the patient's relatives. Each RoI can be given a probability to be the next location the patient will visit depending on the importance of the RoI and the time of movement.

Part IV

Conclusion

CHAPTER 10

Concluding remarks

10.1 Summary of Contributions

In this thesis, we have analysed trajectory data in the GPS and geo-social network based forms. Our analysis includes RoI discovery, location prediction and wandering behaviour detection. The thesis is divided into four parts: introduction (Chapters 1 and 2), contributions to deep learning models for location prediction (Chapters 3, 4, 5, 6 and 7), contributions to specific applications (Chapters 8 and 9) and conclusion (Chapters 10).

In Chapter 4, we have proposed a novel model, called STF-RNN, for predicting people's next movement based on mobility patterns obtained from GPS devices logs. Two main features are involved in model operations, namely, the space which is extracted from the collected GPS data and also the time which is extracted from the associated timestamps. The internal representation of space and time features are extracted automatically in the proposed model rather than relying on hand-craft representation. This enables the model to discover the useful knowledge about people's behaviour in more efficient way. Due to the ability of RNN structure to represent the sequences, it is utilized in the proposed model in order to keep track of user's movement history. These tracks helps the model discover more meaningful dependencies and as consequence, enhance the model performance. For all the experiments, we have used two large real-life mobility datasets (Geolife and Gowalla). Evaluation results show that our model improves the prediction effectiveness in comparison with the state-of-the-art models.

In Chapter 5, we have studied the performance of location prediction model through evaluating different architectural configurations. We have described a series of experiments to extend our previous prediction model (*i.e.* STF-RNN). We have proposed time encoding scheme to encode timestamps into particular time identifiers. A set of neural pooling functions are explored in order to extract rich features. We have evaluated the impact of different data inputs on the model final prediction performance. Based on that, different prediction models are proposed that vary in terms of the number and type of input features. Moreover, different input representation methods (*i.e.* embedding learning and one-hot vector), are investigated. We have shown when and where each method can show better results. We have concluded thorought experiments in all our proposed models on the two real-life datasets. Our main findings are as follows. First, multiple pooling functions offers rich sources of feature information, which leads to an improvement on the prediction performance. Second, the number of input features can play an important role in the prediction performance (given the selection of proper and relevant features). However, increasing the number of input features will increase the training time of the model (*i.e.* overhead, efficiency, etc.). LBSs services are very sensitive to speed and efficiency and we cannot trade-off those to improve prediction. Ideally, we want models to achieve the best in both (*i.e.* prediction accuracy and speed or efficiency). Third, using the leaving time input variable/feature only performs well in comparison with using both entering and leaving RoIs (*i.e.* together).

Fourth, using weekday types information is shown to be such a good input feature in improving the prediction performance (in all evaluated models). Fifth, time encoding scheme is useful to provide the prediction model with more information related to the movement behaviour characteristics. The results indicate that considering different timestamp information is always beneficial. However, as mentioned before, we need to balance this with efficiency. Sixth, learning input feature representation can have a significant positive impact on performance, and should be investigated. The best results are obtained when learning the representation of space data input only. Finally, space embedded vector size has a relatively little effect on the model performance. For the model with space learning, it may be worth using a large space embedded vector size where more location information is provided to the model.

In Chapter 6, we have proposed the model MAP (Move, Attend and Predict) to predict a person's future location based on his/her mobility pattern collected by a mobile device. This is achieved by means of a computationally efficient trainable deep neural network model. The proposed model learns which time interval in the trajectory sequences are relevant regarding a specific location. Experimental results, obtained from tests conducted on the two real-life datasets, demonstrate that our model outperforms state-of-the-art models in terms of precision and recall performance metrics.

In Chapter 7, we have proposed another model for the problem of predicting the future location of people movement, called ST-CNN, where CNN is used to discover long-term dependencies which increases the efficiency of the proposed model. Moreover, we have used the embedding learning technique to discover internal representations of the input data. Experiments on two datasets show that the proposed model leads to better results than state-of-the-art models.

In Chapter 8, we have proposed a new approach to discover and predict people's next location based on their mobility patterns, while being computationally efficient. The RoIs discovery method includes two levels. The first level is to group GPS points based on three threshold conditions: distance, time and region density, which identifies a set of CRoIs. The second level is to perform clustering using DBSCAN algorithm on the CRoIs to obtain the RoIs. Soundness and completeness are used as metrics to evaluate the proposed RoIs discovery method. We found that the proposed method is effective compared to other relevant methods in the literature. While the method is able to find most of the RoIs, the method's overall execution time is less than other methods. Moreover, a new model based on MC is proposed to overcome the drawback of classical MC. It considers both space and time contexts. We have shown how classical MC model can be extended to include movement times and how time will improve prediction accuracy. One unique finding in our research is related to the value of integrating users' mobility location/space with time context. In particular, time context is formulated in a way to add extra information to the space context. For better abstraction during building the model, a general transformation function is used to transform the n-order MC into first order. The latter performs better than space-based and time-based models, but the gap between the models' prediction performance is narrowing with the increase of the order. We have evaluated the proposed approach with real-world datasets. Overall, the approach where DRoI and second order MC were used, achieved better performance.

Finally, in Chapter 9, we have presented a real-time smartphone-based monitoring system to discover and predict elderly people behaviours by analysing outdoor trajectories. This is achieved by firstly analysing the elder's mobility data previously collected using our proposed model (ST-CNN) in order to predict the most probable locationshis/her might visit next. Time and spatial-related variables, such as the distance traversed, the direction of the movements and the time spent, are analysed in our ABD model that takes advantage of RNN. The effectiveness and the efficiency of our system for detection of abnormal behaviours are evaluated using different datasets comprising real-world GPS trajectories. Evaluation results show that the system is promising with respect to providing effective assistance for elderly people to reduce the potential risks.

10.2 Future Research Lines

The work presented in this thesis makes contributions to trajectory analysis and prediction. Several directions of future work have been identified during this work as follows:

1. One of our future work is to classify different users' behaviours/patterns based on their daily, weekly and monthly movement habits. We then plan to build different classes of prediction models.
2. Future research will include investigating the prediction model performance on larger datasets and incorporating different deep learning architectures such as a Gated Recurrent Unit or a Long Short-Term Memory unit. Moreover, the prediction model can be extended to include different contextual information such as human-human interactions, the time spent in a location and the distance between locations, to improve performance.
3. Since we only focused on mobility, as a component of human behaviour, we will consider, in future, our approaches to other scenarios of user behaviours' prediction, for instance, shopping and eating habits.
4. The proposed prediction models are unable to deal with new locations (*i.e.* the unseen locations). We call this problem Out-of-Locations which lies in the complete inability of location prediction models to sensibly assign non-zero probability to unseen locations previously. We plan to develop a model that addresses this limitation.
5. Since the theoretical framework corresponding to the deep learning components of *SafeMove* system have been tested, our future challenge will be focused on the complete development of an Internet enabled system and the corresponding mobile application, so as to build a proof-of-concept and conduct a test with potential users.

References

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Agrawal, R. and Srikant, R. (1995). Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, pages 3–14. IEEE.

Al-Molegi, A., Jabreel, M., and Ghaleb, B. (2016). Stf-rnn: Space time features-based recurrent neural network for predicting people next location. In *SSCI 2016*, pages 1–7. IEEE.

Al-Molegi, A., Jabreel, M., and Martnez-Ballest, A. (2018). Move, attend and predict: An attention-based neural model for peoples movement prediction. *Pattern Recognition Letters*, 112:34 – 40.

Al-Molegi, A., Martínez-Ballesté, A., and Jabreel, M. (2017). Geo-temporal recurrent model for location prediction. In *20th International Conference of the Catalan Association for Artificial Intelligence*, pages 126–135.

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. (2017). Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–971.

Algase, D. L., Beattie, E. R. A., and Therrien, B. (2001). Impact of cognitive impairment on wandering behavior. *Western Journal of Nursing Research*, 23(3):283–295.

Algase, D. L., Moore, D. H., Vandeweerd, C., and Gavin-Dreschnack, D. J. (2007). Mapping the maze of terms and definitions in dementia-related wandering. *Aging & Mental Health*, 11(6):686–698.

Asahara, A., Maruyama, K., Sato, A., and Seto, K. (2011). Pedestrian-movement prediction based on mixed Markov-chain model. *GIS '11 Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 25–33.

Asgari, F., Gauthier, V., and Becker, M. (2013). A survey on human mobility and its applications. *arXiv preprint arXiv:1307.0814*.

Ashbrook, D. and Starner, T. (2003). Using GPS to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing 7*, pages 275–286.

Barbounis, T. and Theocharis, J. (2006). Locally recurrent neural networks for long-term wind speed and power prediction. *Neurocomputing*, 69(4):466–496.

Barnes, S. J. and Scornavacca, E. (2004). Mobile marketing: the role of permission and acceptance. *International Journal of Mobile Communications*, 2(2):128–139.

Barwise, P. and Strong, C. (2002). Permission-based mobile advertising. *Journal of interactive Marketing*, 16(1):14–24.

Batista, E., Borràs, F., and Martínez-Ballesté, A. (2015). Monitoring people with MCI: Deployment in a real scenario for low-budget smartphones. In *Proceedings* *of the 6th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–6.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer.

Casino, F., Domingo-Ferrer, J., Patsakis, C., Puig, D., and Solanas, A. (2015). A k-anonymous approach to privacy preserving collaborative filtering. *Journal of Computer and System Sciences*, 81(6):1000–1011.

Cho, E., Myers, S. A., and Leskovec, J. (2011). Friendship and mobility: User movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1082–1090, New York, NY, USA. ACM.

Chorowski, J., Bahdanau, D., Cho, K., and Bengio, Y. (2014). End-to-end continuous speech recognition using attention-based recurrent nn: First results. *NIPS 2014 Workshop on Deep Learning*.

Cipriani, G., Lucetti, C., Nuti, A., and Danti, S. (2014). Wandering and dementia. *Psychogeriatrics*, 14(2):135–142.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Daoui, M., Belkadi, M., Chamek, L., Lalam, M., Hamrioui, S., and Berqia, A. (2012). Mobility prediction and location management based on data mining. In *Next Generation Networks and Services (NGNS), 2012*, pages 137–140. IEEE.

Daoui, M., Belkadi, M., Chamek, L., Lalam, M., Hamrioui, S., and Berqia, A. (2013). Mobility prediction and location management based on data mining. (December):1–4.

Dekar, L. and Kheddouci, H. (2008). A cluster based mobility prediction scheme for ad hoc networks. *Ad Hoc Networks*, 6(2):168–194.

Do, T. M. T. and Gatica-Perez, D. (2014). Where and what: Using smartphones to predict next locations and applications in daily life. *Pervasive and Mobile Computing*, 12:79–91.

Doyle, J., Bailey, C., Scanaill, C. N., and van den Berg, F. (2014). Lessons learned in deploying independent living technologies to older adults homes. *Universal access in the information society*, 13(2):191–204.

Ericsson, E., Larsson, H., and Brundell-Freij, K. (2006). Optimizing route choice for lowest fuel consumption–potential effects of a new driver support tool. *Transportation Research Part C: Emerging Technologies*, 14(6):369–383.

Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *In Proceedings of 2nd International Conference on KDD*, pages 226–231.

Fang, H., Hsu, W.-J., and Rudolph, L. (2009). Cognitive personal positioning based on activity map and adaptive particle filter. In *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, pages 405–412. ACM.

Feng, S., Cong, G., An, B., and Chee, Y. M. (2017). Poi2vec: Geographical latent representation for predicting future visitors. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 102–108.

Gambs, S., Killijian, M.-O., and Del Prado Cortez, M. N. n. (2012). Next place prediction using mobility Markov chains. *Proceedings of the First Workshop on Measurement Privacy and Mobility MPM 2012*, pages 1–6.

Ganti, R. K., Pham, N., Ahmadi, H., Nangia, S., and Abdelzaher, T. F. (2010). Greengps: a participatory sensing fuel-efficient maps application. In *Proceedings* *of the 8th international conference on Mobile systems, applications, and services*, pages 151–164. ACM.

Gao, H., Barbier, G., and Goolsby, R. (2011a). Harnessing the crowdsourcing power of social media for disaster relief. *IEEE Intelligent Systems*, 26(3):10–14.

Gao, H., Tang, J., and Liu, H. (2012). Mobile location prediction in spatio-temporal context. Nokia Mobile Data Challenge Workshop.

Gao, H., Wang, X., Barbier, G., and Liu, H. (2011b). Promoting coordination for disaster relief–from crowdsourcing to coordination. In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 197–204. Springer.

Gatmir-Motahari, S., Zang, H., and Reuther, P. (2013). Time-clustering-based place prediction for wireless subscribers. *IEEE/ACM Transactions on Networking*, 21(5):1436–1446.

Gerber, M. S. (2014). Predicting crime using twitter and kernel density estimation. *Decision Support Systems*, 61:115–125.

Giannotti, F., Nanni, M., Pedreschi, D., Renso, C., Rinzivillo, S., and Trasarti, R. (2009). Geopkdd – geographic privacy-aware knowledge discovery. In *The European Future Technologies Conference (FET 2009)*.

Gohil, M. H. (2014). Mobile location prophecy: An analytical review. *International Journal of Computer Networks and Wireless Communications*, 4(5):327–333.

Gonzalez, M. C., Hidalgo, C. A., and Barabasi, A.-L. (2008). Understanding individual human mobility patterns. *nature*, 453(7196):779.

Guang, Y., Yang, M., and Zhang, X. (2015). A new model for the movement pattern of vacant taxi. In *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pages 1050–1053. IEEE.

Gudmundsson, J., Laube, P., and Wolle, T. (2012). Computational movement analysis. In *Springer handbook of geographic information*, pages 725–741. Springer.

Gupta, M., Intille, S. S., and Larson, K. (2009). Adding gps-control to traditional thermostats: An exploration of potential energy savings and design challenges. In *International Conference on Pervasive Computing*, pages 95–114. Springer.

H. Kaaniche, F. K. (2010). Mobility Prediction in Wireless Ad Hoc Networks using Neural Networks. *Journal of Telecommunications*, 2(1):95–101.

Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. In *ACM sigmod record*, volume 29, pages 1–12. ACM.

Harris, D. and Harris, S. (2012). *Digital Design and Computer Architecture, Second Edition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

Hoey, J., Yang, X., Quintana, E., and Favela, J. (2012). Lacasa: Location and context-aware safety assistant. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*, pages 171–174. IEEE.

Hori, C., Hori, T., Lee, T.-Y., Sumi, K., Hershey, J. R., and Marks, T. K. (2017). Attention-based multimodal fusion for video description. *arXiv preprint arXiv:1701.03126*.

Hossain, S., Hallenborg, K., and Demazeau, Y. (2011). iroute: Cognitive support for independent living using bdi agent deliberation. In *Trends in Practical Applications of Agents and Multiagent Systems*, pages 41–50. Springer.

Huang, W., Lit, M., Hut, W., Song, G., and Xie, K. (2013). Hierarchical Destination Prediction Based on GPS History. pages 972–977.

Kang, J. H., Welbourne, W., Stewart, B., and Borriello, G. (2004). Extracting places from traces of locations. In *Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, pages 110–118. ACM.

Kedia, S. P. (2012). An object tracking scheme for wireless sensor networks using data mining mechanism. *2012 IEEE Network Operations and Management Symposium*, pages 526–529.

Killijian, M.-O., Roy, M., and Trédan, G. (2010). Beyond san francisco cabs: Building a*-lity mining dataset. pages 75–78.

Kim, B., Kang, S., Ha, J.-Y., and Song, J. (2015). Visitsense: Sensing place visit patterns from ambient radio on smartphones for targeted mobile ads in shopping malls. *Sensors*, 15(7):17274–17299.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kleinberg, J. (2007). Computing: The wireless epidemic. *Nature*, 449:287–288.

Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., Behar, J. V., Hern, S. C., and Engelmann, W. H. (2001). The national human activity pattern survey (nhaps): a resource for assessing exposure to environmental pollutants. *Journal of Exposure Science and Environmental Epidemiology*, 11(3):231–252.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Krumm, J. (2011). Ubiquitous advertising: The killer application for the 21st century. *IEEE Pervasive Computing*, 10(1):66–73.

Krumm, J. and Brush, A. B. (2011). Learning time-based presence probabilities. In *International Conference on Pervasive Computing*, pages 79–96. Springer.

Kumar, B. V. and Venkataram, P. (2002). Prediction-based location management using multilayer neural networks. *Journal of Indian institute of science*, 82(1):7–22.

Kumar, S., Kumar, K., and Kumar, P. (2015). Mobility based call admission control and resource estimation in mobile multimedia networks using artificial neural networks. In *Next Generation Computing Technologies (NGCT), 2015 1st International Conference on*, pages 852–857. IEEE.

L. Vintan, A. Gellert, J. P. and Ungerer, T. (2004). Person movement prediction using neural networks. *In First Workshop on Modeling and Retrieval of Context, Ulm, Germany*, 114(4):1–12.

Lai, C. K. Y. and Arthur, D. G. (2003). Wandering behaviour in people with dementia. *Journal of Advanced Nursing*, 44(2):173–182.

Landau, R. and Werner, S. (2012). Ethical aspects of using gps for tracking people with dementia: recommendations for practice. *International Psychogeriatrics*, 24(3):358–366.

Leca, C.-L., Nicolaescu, I., and Rîncu, C.-I. (2015). Significant location detection & prediction in cellular networks using artificial neural networks. *Computer Science and Information Technology*, 3(3):81–89.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Lee, N., Choi, W., Vernaza, P., Choy, C. B., Torr, P. H., and Chandraker, M. (2017). Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 336–345.

Lee, S., Lim, J., Park, J., and Kim, K. (2016). Next place prediction based on spatiotemporal pattern mining of mobile device logs. *Sensors*, 16(2):145.

Lee, S. H., Walters, S. D., and Howlett, R. J. (2008). Intelligent gps-based vehicle control for improved fuel consumption and reduced emissions. In *International*

Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pages 701–708. Springer.

Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., and Ma, W.-Y. (2008). Mining user similarity based on location history. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, page 34. ACM.

Lin, M. and Hsu, W.-J. (2014). Mining gps data for mobility patterns: A survey. *Pervasive and Mobile Computing*, 12:1–16.

Lin, Q., Zhang, D., Connelly, K., Ni, H., Yu, Z., and Zhou, X. (2015). Disorientation detection by mining GPS trajectories for cognitively-impaired elders. *Pervasive and Mobile Computing*, 19:71–85.

Lin, Q., Zhang, D., Huang, X., Ni, H., and Zhou, X. (2012). Detecting wandering behavior based on GPS traces for elders with dementia. In *Proceedings of the 12th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 672–677.

Liu, Q., Wu, S., Wang, L., and Tan, T. (2016). Predicting the Next Location : A Recurrent Model with Spatial and Temporal Contexts. pages 194–200.

Liu, S. (2018). User modeling for point-of-interest recommendations in location-based social networks: The state of the art. *Mobile Information Systems*, 2018.

Löwe, R., Mandl, P., and Weber, M. (2012). Context directory: A context-aware service for mobile context-aware computing applications by the example of google android. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 76–81. IEEE.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Maioli, F., Coveri, M., Pagni, P., Chiandetti, C., Marchetti, C., Ciarrocchi, R., Ruggero, C., Nativio, V., Onesti, A., D'Anastasio, C., and Pedone, V.

(2007). Conversion of mild cognitive impairment to dementia in elderly subjects: a preliminary study in a memory and cognitive disorder unit. *Archives of Gerontology and Geriatrics*, 44:233–241.

Marmasse, N. and Schmandt, C. (2000). Location-aware information delivery withcommotion. In *International Symposium on Handheld and Ubiquitous Computing*, pages 157–171. Springer.

Martínez-Ballesté, A., Al-Molegi, A., and Batista, E. (2018). On the detection of wandering using trajectories datasets. In *2018 9th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–6.

Martínez-Ballesté, A., Budesca, F. B., and Solanas, A. (2015). An autonomous intelligent system for the private outdoors monitoring of people with mild cognitive impairments. In *Advanced Technological Solutions for E-Health and Dementia Patient Monitoring*, pages 137–152. IGI Global.

Martino-Saltzman, D., Blasch, B. B., Morris, R. D., and McNeal, L. W. (1991). Travel behavior of nursing home residents perceived as wanderers and nonwanderers. *The Gerontologist*, 31(5):666–672.

Mathew, W., Raposo, R., and Martins, B. (2012). Predicting future locations with hidden markov models. *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 911–918.

Mikolov, T., Kombrink, S., Deoras, A., Burget, L., and Cernocky, J. H. (2011). Rnnlm - recurrent neural network language modeling toolkit. pages 196–201.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Monreale, A., Pinelli, F., and Trasarti, R. (2009). WhereNext : a Location Predictor on Trajectory Pattern Mining. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*, pages 637–645.

Morzy, M. (2007). Mining frequent trajectories of moving objects for location prediction. *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 667–680.

Mozer, M. (2004). Lessons from an adaptive house.

Mozer, M. C. (1998). The neural network house: An environment hat adapts to its inhabitants. In *InProc. AAAI Spring Symp. Intelligent Environments*.

Open Street Map (2018). OpenStreetMap: Public GPS traces.

Parija, S., Nanda, S. K., Sahu, P. K., and Singh, S. S. (2013a). Location prediction of mobility management using soft computing techniques in cellular network. *International Journal of Computer Network and Information Security*, 5(6):27.

Parija, S., Ranjan, R. K., and Sahu, P. K. (2013b). Location prediction of mobility management using neural network techniques in cellular network. In *Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013 International Conference on*, pages 1–4. IEEE.

Parija, S., Sahu, P. K., Nanda, S. K., and Singh, S. S. (2013c). A functional link artificial neural network for location management in cellular network. In *Information Communication and Embedded Systems (ICICES), 2013 International Conference on*, pages 1160–1164. IEEE.

Patterson, D. J., Etzioni, O., Fox, D., and Kautz, H. (2002). Intelligent ubiquitous computing to support alzheimers patients: Enabling the cognitively disabled. In *Adjunct Proceedings*, page 21.

Patterson, D. J., Liao, L., Gajos, K., Collier, M., Livic, N., Olson, K., Wang, S., Fox, D., and Kautz, H. (2004). Opportunity knocks: A system to provide cognitive assistance with transportation services. In *International Conference on Ubiquitous Computing*, pages 433–450.

Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., and Hsu, M. (2001). Prefixspan: Mining sequential patterns by prefix-projected growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–224. IEEE Computer Society.

Pelekis, N. and Theodoridis, Y. (2014). Mobility data management and exploration.

Perusco, L. and Michael, K. (2007). Control, trust, privacy, and security: evaluating location-based services. *IEEE Technology and society magazine*, 26(1):4–16.

Pollini, G. P. and Chih-Lin, I. (1997). A profile-based location strategy and its performance. *IEEE journal on selected areas in communications*, 15(8):1415–1424.

Quercia, D., Lathia, N., Calabrese, F., Di Lorenzo, G., and Crowcroft, J. (2010). Recommending social events from mobile phone location data. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 971–976. IEEE.

Rao, B. and Minakakis, L. (2003). Evolution of mobile location-based services. *Communications of the ACM*, 46(12):61–65.

Rashidi, P. and Cook, D. J. (2010). Mining and monitoring patterns of daily routines for assisted living in real world settings. In *Proceedings of the 1st ACM International Health Informatics Symposium*, IHI '10, pages 336–345, New York, NY, USA. ACM.

Rodríguez, M. D., Navarro, R. F., Favela, J., and Hoey, J. (2012). An ontological representation model to tailor ambient assisted interventions for wandering. In *Proceedings of the AAAI Fall Symposium: Artificial Intelligence for Gerontechnology*, pages 34–37.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Ryan, C. and Brown, K. N. (2012). Occupant Location Prediction Using Association Rule Mining. pages 23–28.

Shimizu, K., Kawamura, K., and Yamamoto, K. (2000). Location system for dementia wandering. In *Engineering in Medicine and Biology Society, 2000. Proceedings of the 22nd Annual International Conference of the IEEE*, volume 2, pages 1556–1559. IEEE.

Smart Health Research Group (2014). SIMPATIC: Sistema Intel.ligent de Monitorització Privada i Autònoma basat en Tecnologies de la Informació i les Comunicacions.

Solanas, A., Patsakis, C., Conti, M., Vlachos, I. S., Ramos, V., Falcone, F., Postolache, O., Pérez-Martınez, P. A., Di Pietro, R., Perrea, D. N., et al. (2014a). Smart health: A context-aware health paradigm within smart cities. *IEEE Communications Magazine*, 52(8):74–81.

Solanas, A., Patsakis, C., Conti, M., Vlachos, I. S., Ramos, V., Falcone, F., Postolache, O., Perez-Martinez, P. A., di Pietro, R., Perrea, D. N., and Martínez-Ballesté, A. (2014b). Smart health: A context-aware health paradigm within smart cities. *IEEE Communications Magazine*, 52(8):74–81.

Song, L., Kotz, D., Jain, R., and He, X. (2006). Evaluating next-cell predictors with extensive wi-fi mobility data. *IEEE Transactions on Mobile Computing*, 5(12):1633–1649.

Spinsanti, L., Berlingerio, M., and Pappalardo, L. (2013). *Mobility and Geo-Social Networks*, page 315333. Cambridge University Press.

Sposaro, F., Danielson, J., and Tyson, G. (2010). iwander: An android application for dementia patients. In *Engineering in Medicine and Biology Society (EMBC), 2010 annual international conference of the IEEE*, pages 3875–3878. IEEE.

Team, T. T. D., Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., et al. (2016). Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*.

Vemula, A., Mülling, K., and Oh, J. (2017). Social attention: Modeling attention in human crowds. *CoRR*, abs/1710.04689.

Vranova, Z. and Ondryhal, V. (2011). Utilization of selected data mining methods for communication network analysis. *Radioengineering*.

Vu, L., Do, Q., and Nahrstedt, K. (2011). Jyotish: Constructive approach for context predictions of people movement from joint wifi/bluetooth trace. *Pervasive and Mobile Computing*, 7(6):690–704.

Vu, L., Nguyen, P., Nahrstedt, K., and Richerzhagen, B. (2014). Characterizing and modeling people movement from mobile phone sensing traces. *Pervasive and Mobile Computing*, 17:220–235.

Wan, J., Byrne, C., O'Hare, G. M., and O'Grady, M. J. (2011). Orange alerts: Lessons from an outdoor case study. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*, pages 446–451. IEEE.

Wevers, K., Loewenau, J., Durekovic, S., and Lu, M. (2010). interactive-high precision maps for sustainable accident reduction with the enhanced dynamic pass predictor.

Xia, H., Qiao, Y., Jian, J., and Chang, Y. (2014a). Using smart phone sensors to detect transportation modes. *Sensors*, 14(11):20843–20865.

Xia, H., Qiao, Y., Jian, J., and Chang, Y. (2014b). Using smart phone sensors to detect transportation modes. *Sensors*, 14(11):20843–20865.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057.

Yang, C., Sun, M., Zhao, W. X., Liu, Z., and Chang, E. Y. (2017). A neural network approach to jointly modeling social networks and mobile trajectories. *ACM Transactions on Information Systems (TOIS)*, 35(4):36.

Yavaş, G., Katsaros, D., Ulusoy, Ö., and Manolopoulos, Y. (2005). A data mining approach for location prediction in mobile environments. *Data & Knowledge Engineering*, 54(2):121–146.

Ye, Y., Zheng, Y., Chen, Y., Feng, J., and Xie, X. (2009). Mining individual life pattern based on location history. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 1–10.

Yu, Y. and Chen, X. (2015). A survey of point-of-interest recommendation in location-based social networks. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Yuan, J., Zheng, Y., Zhang, L., and Xie, X. (2013). T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on Knowledge and Data Engineering*, pages 2390–2403.

Yuan, J., Zheng, Y., Zhang, L., Xie, X., and Sun, G. (2011). Where to find my next passenger. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 109–118. ACM.

Yümlü, S., Gürgen, F. S., and Okay, N. (2005). A comparison of global, recurrent and smoothed-piecewise neural models for istanbul stock exchange (ise) prediction. *Pattern Recognition Letters*, 26(13):2093–2103.

Zeiler, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Zhang, Y., Dai, H., Xu, C., Feng, J., Wang, T., Bian, J., Wang, B., and Liu, T.-Y. (2014). Sequential click prediction for sponsored search with recurrent neural networks. *arXiv preprint arXiv:1404.5772*.

Zhao, S., Zhao, T., Yang, H., Lyu, M. R., and King, I. (2016). Stellar: Spatial-temporal latent ranking for successive point-of-interest recommendation. pages 315–322. AAAI Conference.

Zheng, Y. (2015). Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–29.

Zheng, Y., Capra, L., Wolfson, O., and Yang, H. (2014). Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):38.

Zheng, Y., Xie, X., and Ma, W. (2010). Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin*, 2(33):32–40.

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

Zook, M., Graham, M., Shelton, T., and Gorman, S. (2010). Volunteered geographic information and crowdsourcing disaster relief: a case study of the haitian earthquake. *World Medical & Health Policy*, 2(2):7–33.

Acronyms

The acronyms are arranged in the order of the first appearance in the work.

GPS Global Positioning System, page 3

GSM Global System for Mobile communications, page 4

LBSs Location-based services, page 4

MCI Mild Cognitive Impairments, page 8

RNN Recurrent Neural Networks, page 9

STF-RNN Space-Time Features-based Recurrent Neural Network, page 9

MAP Move, Attend and Predict, page 9

RoIs Regions-of-Interest, page 10

MC Markov Chain, page 10

CNN Convolutional Neural Network, page 11

ABD Abnormal Behaviour Detection, page 11

ST-CNN SpaceTime-Convolutional Neural Network, page 13

RFID Radio Frequency Identification, page 16

DJ Density-Joinable, page 20

SVM Support Vector Machine, page 21

NN Neural Networks, page 21

n-MMC n-Mobility Markov Chain, page 21

MMM Mixed Markov-Chain Model, page 21

HMM Hidden Markov Model, page 22

UCC University College Cork, page 25

UIM University of Illinois Movement, page 25

MH Mobile Hos, page 26

RWM Random Waypoint Mobility, page 26

ST-RNN Spatial Temporal Recurrent Neural Network, page 26

GTD Global Terrorism Database, page 26

LZ Lempel-Ziv, page 27

LSTM Long Short-Term Memory, page 27

OK Opportunity Knocks, page 28

TP True Positive, page 34

FP False Positive, page 34

TN True Negative, page 34

FN False Negative, page 35

DBSCAN Density-Based Spatial Clustering of Applications with Noise, page 44

BPTT Backpropagation Through Time, page 53

ST-PA Space-Time Pooling-based Architecture, page 63

STE-RNN Space-Time of Entering RNN, page 66

STW-RNN Space-Time Week RNN, page 67

WD weekday, page 67

WE weekend, page 67

AllWD All Week Day, page 67

STES-RNN Space-Time Encoding Scheme RNN, page 68

Wo-RL Without Representation Learning, page 69

S-RL Space Representation Learning, page 69

T-RL Time Representation Learning, page 69

ST-RL Space-Time Representation Learning, page 69

STP Spatio-Temporal-Periodic, page 78

MWDH Month Week DayHours, page 71

MW5TD Month Week 5TimeslotsDay, page 71

WDH Week DayHours, page 71

W5TD Week 5TimeslotsDay, page 71

noAtten No Attention, page 84

ReLU Rectified Linear Unit, page 94

PPM Prediction by Partial Match, page 102

SPM Sampled Pattern Matching, page 102

CRoIs Candidate RoIs, page 102

st-GMM space-time-based Generalized Markov Model, page 103

DRoI Discovering RoI, page 104

DCRoI Discovering Candidate RoI, page 104

CRoIC Candidate RoI Clustering, page 106

GMM Generalized Markov Model, page 111

s-GMM space-based Generalized Markov Model, page 112

t-GMM time-based Generalized Markov Model, page 112

ind-st-GMM independent space-time-based Generalized Markov Model, page 114

dep-st-GMM dependent space-time-based Generalized Markov Model, page 114

w-st-GMM weighted space-time-based Generalized Markov Model, page 115

UNIVERSITAT
ROVIRA i VIRGILI