Semantic Trajectories and Predicting Future Semantic Locations

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Antonios Karatzoglou

aus Veria, Griechenland

Tag der mündlichen Prüfung: 16 Jan 2019

Erster Gutachter: Prof. Dr. Michael Beigl

Zweiter Gutachter: Prof. Dr. Heiner Stuckenschmidt

For my family

There are many that have earned my gratitude for their contribution during my PhD time. Firstly, I would like to express my sincere gratitude to my advisor, Prof. Michael Beigl, for the continuous support during my PhD study and related research. His guidance helped me in all the time of research and writing of this thesis. Beside my advisor, I would also like to thank my co-advisor Prof. Heiner Stuckenschmidt for his insightful comments and for helping shape this thesis. I thank my fellow colleagues at TECO for all the fruitful discussions that helped me overcome my initial *naivety* through their kind-hearted constructive advice and criticism. I thank all my former colleagues by Bosch from the UPA team for both triggering scientific concerns and bringing a valuable practical perspective into my research topic. In particular, I would like to thank Dr. Oliver Rogalla for his guidance and support helping me identify and compensate personal limitations and deficits. I would also like to thank Stefan, Christian, Mark, Jan and Alex for the great number of interesting and in-depth discussions. A sincere thanks goes to all my Seminar, Bachelor and Master students for their valuable contribution in the numerous experiments, user studies and implementations. Finally, I would like to express my deepest gratitude to my family, my parents and my brother, as well as my friends for their love, continued patience and endless moral and spiritual support throughout this long journey. This dissertation would not have been possible without them.

Abstract

Location awareness refers in connection with ubiquitous computing to the ability of a system to determine its position and perceive its environment. It is indispensable for achieving adaptable, context-tailored services and applications. In recent years, in order to provide their services in a timely or even forward-looking manner to their users, system developers and service providers went one step further and employ increasingly location prediction techniques. The technological leap in the last few years and the wide spread of smart mobile devices made this possible. Location prediction is further used more and more often for supporting resource management and decision making processes, such as in telecommunication or transportation networks. Moreover, knowing where the user is going to be next as well as his overall movement behaviour provides deep insights into the actual person and her actions. This kind of information can help systems and applications reach a higher personalization level and thus are extremely valuable (e.g., see virtual (digital) assistants and recommendation systems, to name but a few). For all these reasons, localization and location prediction in particular have gained significantly in importance over the past few years.

A rich variety of modelling and prediction techniques for human movement patterns has been studied in the literature so far. The majority is represented by either probabilistic or machine learning models applied on GPS or cell tower radio signal recordings. Recent works utilize semantic knowledge to enrich the recorded trajectories. The resulting trajectories, referred to as *semantic trajectories*, reduce the thousands upon thousand of GPS points to the essentials, represented by a number of few significant *semantic locations*. This brings transparency into the models and helps in reaching a higher level of understanding of human movement. Despite the benefits, research in modelling semantic trajectories and predicting upon them is still in its early stages.

The primary objective of this thesis is to fill this gap, join the aforementioned growing body of research and lay a solid foundation for future investigations by exploring a variety of ways for modelling semantic trajectories and predicting future semantic locations. These include probabilistic methods, like multi-dimensional Markov Chains, as well as Artificial Neural Network based

approaches such as Convolutional networks (CNN) and Attention-based Sequence to Sequence (Seq2Seq) learning. Above and beyond this high-level objective, this thesis' contribution can be summarized in the following points:

- Investigation regarding the influence of the *semantic representation level* that is used to describe the locations in the semantic trajectories on the predictive performance of the models.

- Investigation regarding the influence of the *degree of semantic enrichment* of the respective trajectories on the predictive performance of the models with respect to the amount and the type of the context- and user-specific features that are taken additionally into consideration.

- Investigation regarding the incorporation of semantic knowledge, represented by an *additional semantic layer*, into the training of neural networks with respect to both the models' convergence properties and their prediction performance using the example of semantic location prediction.

The approaches in the presented work are evaluated using a number of real-world datasets. Some were collected through own user studies and some are open and can be freely used for scientific purposes. A small part of the findings discussed in this thesis are based on a limited evaluation dataset and could in a way be denoted as not generalizable. Nevertheless, they provide a strong indication and together with the rest of the findings presented in this work lay a solid groundwork for future research.

The research in this thesis has identified certain advantages, especially in terms of precision and recall, favouring the neural networks, from which the recurrent as well as the convolutional architectures could stand out. However, in some individual cases, some probabilistic models could achieve equal or even higher scores. This can be mainly attributed to the size and properties of the corresponding available training and evaluation datasets and the fact that neural networks are much more sensitive to the properties of the data than probabilistic models.

The investigation has also shown that the semantic representation level has indeed an impact on the predictive performance. Semantic trajectories

represented at a higher level provide a better basis for accurate predictions than the ones represented at lower levels. A possible explanation for this might be the fact that human movement shows a greater regularity degree at higher levels than at lower ones.

Furthermore, investigations also confirmed that the degree of semantic enrichment through the use of additional context information, such as the users' activity, personality and emotional state, can lead to better results. However, at the same time we identified some limitations attributed most likely to the larger feature number in connection with the small training dataset and the associated curse of dimensionality phenomenon [Nas07]. In tangible terms, the benefits brought by the additional features are being partly eliminated by themselves at the same time due to the lack of a larger dataset that would support a more generalizable model and thus a higher prediction accuracy. The fact that the models showing the highest results belong to users who provided the largest and most consistently annotated datasets supports this assumption.

Finally, with regard to incorporating an additional semantic layer into the training process of neural networks, the results support our initial assumptions and show that the utilization of semantic knowledge can significantly improve the training behaviour of neural networks as well as their predictive performance. Moreover, the findings of our investigation provide strong evidence that the fusion of knowledge-driven models with data-driven ones could prove very useful in exploiting faster a much wider range of information contained in the available data.

Zusammenfassung

Der Begriff *Standortwahrnehmung* (engl. *Location Awareness*) bezieht sich in Zusammenhang mit sog. Ubiquitous Computing Systemen auf die Fähigkeit eines Systems seine Umgebung wahrzunehmen und seine Position im Raum zu erkennen. Solch eine Fähigkeit ist unerlässlich für das Erreichen von anpassungsfähigen, an den jeweiligen Kontext maßgeschneiderten Diensten und Applikationen. In den letzten Jahren, Dienstleister, um ihre Dienste an Nutzern rechtzeitig oder sogar vorausschauend anbieten zu können, gehen sie einen Schritt weiter und setzen vermehrt auf Standortvorhersage-Techniken. Der Technologiesprung der letzten Jahre und die weite Verbreitung von intelligenten mobilen Geräten hat dieses Unterfangen unterstützt. Darüber hinaus, Standortvorhersagesysteme werden immer häufiger zwecks einer effizienteren Resourcenverwaltung oder der Optimierung von Entscheidungsprozessen eingesetzt, wie zum Beispiel in Telekommunikations- oder Verkehrsnetzen. Schließlich, das Wissen des nächsten Ortes eines Nutzer und seine Bewegungsmuster gewähren einen tiefen Einblick in die Person an sich und ihre aktuelle und künftige Handlungen. Diese Art von Informationen kann Systeme zu einem höheren Personalisierungsgrad führen und sind sehr wertvoll (siehe z.B. digitale persönliche Assistenten und Empfehlungssysteme, u.a.). Aus diesen Gründen, haben Standortvorhersagemethoden in den vergangenen Jahren stark an Bedeutung gewonnen.

Die heutige Literatur umfasst eine reiche Vielfalt von Modellierungs- und Prädiktionstechniken für menschliche Bewegungsmuster. Die Mehrheit wird durch statistische oder Machine Learning basierte Verfahren repräsentiert, angewendet auf GPS oder Mobilfunkmast Signalen. Neuere Arbeiten gehen über die Nutzung von rein numerischen Daten hinaus und verwenden semantisches Wissen um die verfügbare Trajektorien anzureichern. Die resultierenden Trajektorien werden als *semantische Trajektorien* bezeichnet und reduzieren die abertausend aufgezeichnete GPS Punkte auf den wesentlichen Teil der menschlichen Bewegung, repräsentiert durch eine kleine Zahl signifikanter, *semantischer Orte*. Das verleiht den Prädiktionsmodellen eine gewisse Transparenz und hilft das Erreichen eines besseren Verständnisses der menschlichen Bewegung. Trotz der Vorteile, die Forschung um die Modellierung und Prädiktion semantischer Trajektorien befindet sich noch in einem sehr frühen Stadium.

Das Hauptziel dieser Doktorarbeit ist diese Lücke zu füllen, sich der wachsenden Zahl an Untersuchungen in diesem Gebiet anzuschließen und einen soliden Grundstein für zukünftige Untersuchungen zu legen. Zu diesem Zweck, die vorliegende Arbeit erkundet eine Reihe von Wegen zur Modellierung von semantischen Trajektorien und zur Prädiktion der nächstbesuchten Standorte der Nutzer. Diese beinhalten sowohl probabilistische Verfahren wie multidimensionale Markov Ketten, als auch Künstliche Neuronale Netze (KNN) wie Convolutional Networks (CNN) und Attention-basiertes Sequence-to-Sequence Learning (Seq2Seq). Jenseits dieser übergeordneten Zielsetzung, der Beitrag dieser Dissertation kann in den folgenden Punkten zusammengefasst werden:

- Untersuchung hinsichtlich der Auswirkung der semantischen Repräsentationsebene, welche für die Beschreibung von Standorten in den semantischen Trajektorien verwendet wird, auf die prädiktive Performanz der Standortvorhersagemodelle.

- Untersuchung hinsichtlich der Auswirkung des gewählten Grades der semantischen Anreicherung der verfügbaren Trajektorien auf die prädiktive Performanz der Standortvorhersagemodelle

- Untersuchung hinsichtlich der Integration von semantischem Wissen in das Training von Neuronalen Netzen durch das Hinzufügen einer zusätzlichen semantischen Ebene in Bezug auf das Konvergenzverhalten der Standortvorhersagemodelle und deren Prädiktionsperformanz.

Die verschiedenen vorgeschlagenen und erkundeten Ansätze der vorliegenden Arbeit wurden mit Hilfe einer Gruppe realer Datensätze evaluiert. Ein Teil davon ist frei verfügbar für wissenschaftliche Zwecke und ein Teil entstand aus eigenen Experimenten und Nutzerstudien. Dies hat in Einzelfällen dazu geführt, dass ein kleiner Teil der in dieser Arbeit diskutierten Ergebnisse auf eine relativ begrenzte Datenmenge basiert, was teilweise auf eine entsprechend begrenzte Generalisierbarkeit hindeutet. Dennoch, sie liefern ein schwerwiegendes Indiz und legen zusammen mit den restlichen Aussagen der Arbeit ein solides Fundament für zukünftige Untersuchungen.

Die Untersuchungen der vorliegenden Arbeit haben gewisse Vorteile seitens der Nutzung von Künstlichen Neuronalen Netzen identifiziert, besonders in

Hinsicht auf Präzision und Trefferquote. Dabei stachen insbesondere die Stärken von rekurrenten (RNN, LSTM) und faltenden (CNN) Architekturen hervor. Allerdings, in bestimmten Fällen konnten manche probabilistische Modelle ähnlich gut, oder sogar bessere Ergebnisse erzielen. Dies ist im Wesentlichen auf die Menge und die Eigenschaften der verfügbaren Trainings- und Evaluationsdatensätze zurückzuführen und die Tatsache, dass Neuronale Netze im Allgemeinen und im Vergleich zu statistischen Verfahren datenempfindlicher sind.

Es hat sich ebenfalls gezeigt, dass die semantische Repräsentationsebene in der Tat einen signifikanten Einfluss auf die Vorhersagekraft der Modelle hat. Semantische Trajektorien beschrieben in einer höheren semantischen Ebene bieten eine bessere Grundlage für genauere Vorhersagen als Trajektorien einer niedrigeren Ebene. Ein möglicher Grund dafür könnte die Tatsache sein, dass menschliche Bewegung einen höheren Regelmäßigkeitsgrad zeigt je höher die Ebene in der diese modelliert wird ist.

Des Weiteren haben Untersuchungen bestätigt, dass der Grad der semantischen Anreicherung der Trajektorien, indem zusätzliche Kontext-Information, wie die Aktivität der Nutzer, ihre Persönlichkeit und ihr emotionaler Zustand, in Betracht gezogen werden, zu besseren Ergebnissen führen kann. Allerdings, in manchen Fällen konnten auch bestimmte Einschränkungen festgestellt werden, die auf die größere Anzahl der betrachteten Trainingsmerkmale in Zusammenhang mit dem entsprechend kleinen verfügbaren Trainingsdatensatz zurückzuführen sind. Dieses Phänomen wurde von Bellman als Fluch der Dimensionalität bezeichnet [Nas07]. Konkret bedeutet dies, dass die Vorteile geboten von den zusätzlichen Merkmalen gleichzeitig teilweise durch sich selbst wieder eliminiert werden, angesichts des Fehlens eines größeren Datensatzes, welcher ein generalisierbareres Modell und somit eine höhere Genauigkeit unterstützen würde. Die Tatsache, dass die Prädiktionsmodelle mit der besten Performanz zu den Nutzern mit den meisten Annotationen zuzuweisen sind unterstützt diese Annahme.

Schließlich, in Hinsicht auf die Integration und Anwendung einer zusätzlichen semantischen Ebene in das Training von Neuronalen Netzen, die Untersuchungen dieser Arbeit untermauern die ursprüngliche Annahme und Grundidee und zeigen, dass das Einsetzen vom externen semantischen Wissen sowohl

zu einer signifikanten Verbesserung des Training-Verhaltens der neuronalen Netze, als auch zu einer höheren Vorhersagegenauigkeit führen kann. Darüber hinaus, diese Ergebnisse geben starke Hinweise dafür, dass die Fusion von wissensbasierten und datengetriebenen Modellen über den speziellen Fall der Standortvorhersage hinaus sich ebenfalls als sehr nützlich erweisen könnte, da diese einen schnelleren und tieferen Blick in die verfügbaren Daten ermöglicht.

Οὐ γὰρ ὡς ἀγγεῖον, ὁ νοῦς ἀποπληρώσεως, ἀλλ' ὑπεκκαύματος μόνον, ὥσπερ ὕλη, δεῖται…

„Der Geist ist nicht wie ein Gefäß, das gefüllt werden soll, sondern wie Holz, das lediglich entzündet werden will."

Πλουτάρχος, «Περὶ τοῦ ἀκούειν»

CONTENTS

1 Introduction **1**

2 Background Theory **11**

2.1 Introduction 12

2.2 Semantic Trajectories 12

2.3 Evaluation Methods and Metrics 13

2.3.1 k-Fold Cross Validation 15

2.3.2 Walk Forward Validation 16

Part I **18**

3 Modeling Semantic Trajectories Using Markov Chains: 2-Level Semantics and Multiple Dimensions **21**

3.1 Introduction 23

3.2 Related Work 23

3.3 Markov Chain Model 24

3.4 Applying Markov Chains on Semantic Trajectories of Different Semantic Level 25

3.4.1 Evaluation 27

3.5 A Multi-dimensional Markov Chain Model for Modelling Semantic Trajectories 28

3.5.1 Evaluation 30

3.6 Conclusion 36

4 Semantic-Enhanced Multi-Dimensional Markov Chains: Applying Purpose-of- Visit-Driven Semantic Similarity on Semantic Trajectories 39
4.1 Introduction . . . 41
4.2 Related Work . . . 44
4.3 Purpose-of-Visit-Driven Semantic Similarity (PoVDSSA) on Semantic Trajectories . . . 46
4.3.1 Semantic Enriched Location Data and Ontology-based Knowledge Base . . . 49
4.3.2 Purpose-of-Visit-Driven Semantic Similarity Analysis . . . 51
4.3.3 PoVDSSA-based Model Optimization Process . . . 53
4.4 Evaluation . . . 54
4.5 Conclusion . . . 59

5 Combining Markov Chains with Matrix Factorization for Overcoming Data Sparsity 61
5.1 Introduction . . . 63
5.2 Related Work . . . 64
5.3 Matrix Factorization . . . 65
5.4 FPMC-based Semantic Location Prediction . . . 66
5.4.1 Learning Algorithm . . . 70
5.5 Evaluation and Discussion . . . 71
5.5.1 Dataset . . . 71
5.5.2 Evaluation and discussion . . . 72
5.6 Conclusion . . . 76

Part II 78

6 Applying Artificial Neural Networks on Semantic Trajectories: FFNN, RNN and LSTM 81
6.1 Introduction . . . 83
6.2 Related Work . . . 84
6.3 Neural Network Based Semantic Location Prediction - Theory, Implementation, and Parameter Selection . . . 85
6.3.1 Feed-Forward Neural Networks (FFNN) . . . 86

6.3.2 Recurrent Neural Networks (RNN) . . . 87
6.3.3 Long Short-Term-Memory Neural Networks (LSTM) . . 89
6.3.4 Hyper Parameter Selection . . . 89
6.4 Evaluation . . . 90
6.5 Conclusion . . . 97

7 Applying Artificial Neural Networks on Semantic Trajectories: CNN 99
7.1 Introduction . . . 101
7.2 Related Work . . . 102
7.3 Convolutional Neural Networks (CNN) . . . 103
7.4 CNNs on Semantic Trajectories . . . 104
7.5 Evaluation . . . 107
7.6 Conclusion . . . 112

8 Applying Artificial Neural Networks on Semantic Trajectories: Sequence to Sequence (Seq2Seq) Learning 115
8.1 Introduction . . . 117
8.2 Related Work . . . 118
8.3 Background Theory . . . 119
8.3.1 Sequence to Sequence Learning (Seq2Seq) . . . 119
8.3.2 Attention . . . 120
8.4 Seq2Seq Learning on Semantic Trajectories . . . 121
8.5 Evaluation . . . 123
8.6 Conclusion . . . 129

9 Applying Artificial Neural Networks on Semantic Trajectories: Enhancing the Affective Sensitivity 131
9.1 Introduction . . . 133
9.2 Related Work . . . 135
9.3 Basic Concepts and Preliminaries . . . 138
9.3.1 Ontologies . . . 138
9.3.2 Personality and Emotional State Models . . . 141
9.4 Overview of Our Approach . . . 142
9.4.1 User Study . . . 142

9.4.2 Architecture . . . 146
9.4.3 Prediction Model Variations . . . 147
9.5 Evaluation and Results . . . 151
9.5.1 User Study Results . . . 152
9.5.2 Feeding LSCFs into the LSTM . . . 153
9.5.3 Combined Learning - Feeding Similar Locations Into the LSTM . . . 157
9.5.4 Training Convergence . . . 159
9.5.5 Single-User Models . . . 161
9.6 Conclusion . . . 163

10 Semantic-Enhanced Learning (SEL) on Artificial Neural Networks Using the Example of Semantic Location Prediction 165
10.1 Introduction . . . 167
10.2 Related Work . . . 169
10.3 Semantic Knowledge Bases And Ontologies . . . 171
10.4 Semantic Similarity . . . 172
10.5 Semantic-Enhanced Learning (SEL) . . . 174
10.5.1 Semantic Similar Location Layer (SemSimLoc) . . . 174
10.5.2 Low-Level High-Level Location Layer (LowHighLevelLoc) 176
10.5.3 Semantic-Enhanced Backpropagation (SEBP) . . . 176
10.6 Evaluation And Discussion . . . 178
10.7 Conclusion . . . 190

11 Conclusions and Outlook 193
11.1 Conclusions . . . 195
11.2 Outlook . . . 199

12 Publication List of the Author 201

LIST OF FIGURES

1.1 Mobile user in motion. 2
1.2 A typical semantic trajectory. 4
1.3 Overview of all explored methods in this thesis. 9

2.1 Example of an 1-day long Semantic Trajectory. 14
2.2 k-Fold Cross validation. 16
2.3 Time Series Split validation. 16

3.1 Semantic location transition probability matrix. 25
3.2 From left to right: Location transition probability matrix, Time-of-day-specific transition probability matrices, Day-of-week-specific sets of transition probability matrices. 29
3.3 Markov Chain selection process and location prediction. 30
3.4 Screenshot of our Android tracking and annotation app. 32
3.5 Left: Number of tracked GPS points per user. Right: Number of annotated locations per user. 33
3.6 Semantic label distribution of locations. 33
3.7 Multiple purposes of visiting a food location (beyond eating). . . 34
3.8 Number of (unique) multipurpose locations proportional to the total number of (unique) locations per user. 35
3.9 1. order multi-dimensional Markov Chain model vs. Ying's approach with regard to accuracy, precision, recall and F-score. 36

4.1 Purpose-of-Visit-Driven Semantic Similarity Analysis based location prediction (PoVDSSA) framework. 47
4.2 Layer diagram of the Purpose-of-Visit-Driven Semantic Similarity Analysis based location prediction framework (PoVDSSA). . 48
4.3 Purpose of Visit 50
4.4 Actions 50
4.5 Locations 50
4.6 Introducing a new entity PoVDF for representing an n-ary relation. 51
4.7 Comparison of our approach (PoVSSA) to Ying et al.'s approach with regard to accuracy, precision, recall and F-score. 55
4.8 Comparison of a number of different variants of our approach (PoVSSA) against the Markov model described in Chapter 3. . . 56
4.9 Comparison of various different configurations of our approach (PoVSSA) to the Markov model described in Chapter 3 for the user with the most annotated locations and activities. 59

5.1 User-Location matrix to user-specific Location-Location matrices. 68
5.2 Distribution of the high-level semantic locations in the filtered MIT dataset. 72
5.3 Development of the F1-Score during the training phase ($k_{I,L} = k_{U,I} = 128$ and $n = 1$) 73
5.4 F1-Score for a series of values of $k_{U,I}$ and $k_{I,L}$ 74
5.5 Average measured values of MC (3. Order), MF, MFC and FPMC ($k_{I,L} = k_{U,I} = 128$, 1. Order ($n = 1$)). 74
5.6 Generic vs. individual modelling in % ($k_{I,L} = k_{U,I} = 128$ and $n = 1$). (Highest scores obtained for each case) 76

6.1 Visualization of a typical deep multi-layer FFNN. 86
6.2 Unfolded RNN. 88
6.3 Distribution of the high-level semantic locations in the filtered MIT dataset. 91
6.4 Average results over all single-user models taking temporal information into consideration at the higher semantic level (MIT Reality Mining dataset). 92

6.5 Average results over all single-user models without taking temporal information into consideration at the higher semantic level. 93
6.6 Average results over all single-user models taking temporal information into consideration at the lower semantic level. 94
6.7 Average results over all single-user models without taking temporal information into consideration at the lower semantic level. 95
6.8 Average accuracy scores over 10 runs (3-month long single user dataset) 96
6.9 Maximum accuracy scores over 10 runs (3-month long single user dataset) 96

7.1 Typical CNN architecture for Image Classification. 103
7.2 An abstracted view on the core layers of our CNN-based location prediction approach. 105
7.3 Fixed-length trajectory extraction with a trajectory length of 3. 108
7.4 Comparison of evaluation results of our architecture with and without embedding layer vs. FFNN. 110
7.5 Evaluation results of CNN vs. FFNN, RNN and LSTM of Chapter 6 (average over all single user models at the lower semantic level). 110
7.6 Evaluation results of CNN vs. FFNN, RNN and LSTM of Chapter 6 (average over all single user models at the higher semantic level). 111
7.7 Comparison of the Multi-User model with the average performance of the Single-User CNN models. 112
7.8 Impact of time in the case of the low-level semantic representation. 112

8.1 Unfolded RNN- or LSTM-based Seq2Seq model. 120
8.2 Attention-based Seq2Seq learning on semantic trajectories. . . . 122
8.3 Multi-user model average prediction performance with and without embeddings (MIT dataset). 125
8.4 Top single-user model prediction performance (MIT dataset). . . 127
8.5 Long-Term multi-user model prediction performance (sequence length = 25, MIT dataset). 128
8.6 Multi-user model prediction performance (SDP dataset). 129

9.1 A snippet from our ontology used to store the LSCFs 140
9.2 Location Specific Cognitive Frame (LSCF) 141
9.3 Screenshots from the iOS (left) and the Android (right) version of our app. 144
9.4 Sentient Destination Prediction architecture. 146
9.5 Number of occurences of each unique LSCF. 151
9.6 Total amount of annotations per user 153
9.7 The F-Scores of the various tested models at the lower semantic representation level, i.e., the original labels from the dataset (after mapping them to our ontology)(70 semantic locations). . . 154
9.8 The F-Scores of the various tested models at the higher semantic representation level (10 semantic locations). 156
9.9 The (macro) accuracy scores of the various tested models at the lower semantic representation level, i.e., the original labels from the dataset (after mapping them to our ontology)(70 semantic locations). 157
9.10 The (macro) accuracy scores of the various tested models at the higher semantic representation level (10 semantic locations). . . 158
9.11 The weighted F-Scores by each method at the lower (left) and the higher semantic level (right) over the epoch number. 160
9.12 The weighted F-Scores by each similarity-based approach at the lower (left) and the higher semantic level (right) over the epoch number. 161

10.1 Part of our MIT Reality Mining Semantic Location Ontology. 172
10.2 SemSimLoc(+) and LowHighLevelLoc(+) approach. 175
10.3 Semantic-Enhanced Backpropagation (SEBP). 176
10.4 SemSimLoc (10 epochs, SDP). 180
10.5 SemSimLoc (100 epochs, SDP). 181
10.6 Test accuracy SDP (10 and 100 epochs). 182
10.7 SemSimLoc+ (10 epochs, SDP). 183
10.8 SemSimLoc+ (100 epochs, SDP). 184
10.9 Test accuracy, SDP, SemSimLoc+ (10 and 100 epochs). 185
10.10SemSimLoc (top) and SemSimLoc+ (bottom) (10 epochs, MIT, Shortest Path). 185

10.11Test accuracy: SemSimLoc (a) and SemSimLoc+ (b)(10 and 100 epochs, MIT, Shortest Path). 186
10.12LowHighLevelLoc (10 (left) and 100 (right) epochs), SDP. . . . 186
10.13LowHighLevelLoc (10 (left) and 100 (right) epochs, MIT). . . . 187
10.14Test accuracy: LowHighLevelLoc (10 (left) and 100 (right) epochs, SDP and MIT). 187
10.15SEBP (10 (top) and 100 (bottomt) epochs, SDP). 188
10.16SEBP MIT (10 (top)) and 100 (bottom) epochs). 189
10.17Test accuracy: SEBP (10 (top)) and 100 (bottom) epochs, SDP and MIT). 189

LIST OF TABLES

3.1 10-fold cross validation average and maximum accuracy results of the 1. order Markov model for the 12-week single-user dataset for both the top ($Level^{(1)}$) and the lower semantic level ($Level^{(3)}$). 28

3.2 10-fold cross validation average accuracy, maximum accuracy and weighted F-Score results of the 1. order Markov model for the 8-week multi-user dataset for both the top ($Level^{(1)}$) and the lower semantic level ($Level^{(3)}$). 28

5.1 Example of a User-Item rating matrix. This table describes the ratings that 5 users gave to 4 different items (e.g., movies). . . . 67

5.2 Matrix approximation by applying Matrix Factorization on the User-Item rating matrix of Table 5.1. 67

5.3 Reality Mining dataset statistics. 71

6.1 Final parameter setup obtained as described in subsection 6.3.4. MM refers to the Markov Model, which is used as reference in our evaluation. 90

6.2 Average statistic scores without and with taking time and day into account (1st and 2nd value respectively) (Reality Mining MIT dataset) 95

7.1 Impact of trajectory length. Filter Size: 2, Embedding Dimension: 100, Number of Filters: 50, Dropout Probability: 0.4, Batch Size: 100, Learning Rate: 0.001, Number of Epochs: 10. . 109

8.1 Range of investigated model parameter values. 124
8.2 Optimal model parameter values for the multi-user models. . . . 124

9.1 The best configuration for each approach at the lower semantic representation level with 70 location labels. 154
9.2 The best configuration for each approach at the higher semantic representation level with 10 location labels. 155
9.3 The best configuration for each combined learning approach at the lowest representation semantic level with 70 location labels. 158
9.4 The best configuration for each combined learning approach at the higher semantic representation level with 10 location labels. 158
9.5 Single-user model results (weighted F-Scores) for the 5 most frequently annotating users at the lower semantic representation level. 161
9.6 Single-user model results (weighted F-Scores) for the 5 most frequently annotating users at the higher semantic representation level. 162

10.1 LSTM hyper parameter values. 178

CHAPTER 1

INTRODUCTION

In recent years, rapid advances in technology have led to the rise of ubiquitous computing [Wei91] and ambient intelligence [Enc08]. The existence of smart environments and a flood of personal devices capable of sensing, collecting and processing information about mobile users and their environment came so to fruition. This type of information can be summarized in a single word, namely *context*. A. K. Dey defines context as follows:

> *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves* [Dey01].

Consequently, *context-aware* systems and applications are capable of offering intelligent, tailored to the users and their environment services. *Location* represents a particularly important type of context information. There exists a great many of applications and services by now, so-called *Location Based Services (LBS)*, that are premised on the users' location, such as location-based advertising and marketing. In addition, social networks experience an upgrade nowadays through utilizing the current location of their users (*Location-Based Social Networks (LBSNs)*). In the case of LBSNs, the information is being provided either manually by the users' entries (aka *check-ins*) or forwarded automatically from the users' personal devices. Location awareness can also contribute to a more efficient and sustainable resource management, for instance in the traffic and transportation domain or the telecommunication sector. Finally, we shouldn't forget the obvious benefits of location awareness in

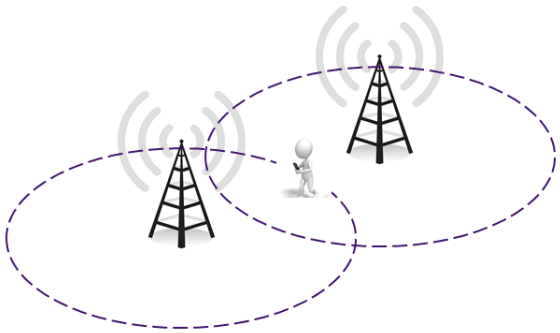

Figure 1.1: Mobile user in motion.

providing intelligent navigation solutions.

Nowadays, LBS providers go one step further in order to improve the quality of their services. Instead of just using the current location of their users, they increasingly base their services on a number of location prediction techniques, aiming in this way at providing users with more timely solutions. Such location prediction techniques include probabilistic methods, like Markov Chains (MC) (see Chapter 3) and Hidden Markov models (HMM), probabilistic graphical models, like Bayesian networks, Decision Trees and Artificial Neural Networks (ANN) of various type such as Feed-Forward Neural Networks, Recurrent Neural Networks, Long Short-Term Memory networks, Convolutional Neural Networks and other (see Chapters 6 and 7). The majority are thus data-driven and pattern-based methods that rely on observed human movement data.

Movement data describe the movement of objects in some geographical space over a certain period of time under certain object-specific and contextual conditions [AAB+11]. It has been shown, that humans feature regularities in their manner of moving in space and time. Individuals tend to visit a same small set of locations again and again generating reproducible patterns [GHB08]. Csaji et al. came also to a similar result by exploring the mobility patterns of 100.000 mobile phone users, namely that most people spend most of their time at only few locations [CBT+13]. However, human mobility patterns consist of random parts as well. For instance, there exist studies, including the aforementioned [GHB08], that highlight the explorative nature of humans and show that the number of visited locations grows over time [PSR+15]. This brings in turn a certain degree of uncertainty into the estimations of location prediction systems and sets certain predictability limits in human mobility [SQBB10].

The aforementioned small set of locations refers to locations that are closely associated to certain purposes for visiting them, which in turn comprise a number of human activities and corresponding experiences [Rel76]. Such *significant* locations can relative easily be extracted by applying temporal and spatial threshold-based rules like in Asbrook et al.'s work [AS03]. For instance, every place at which a user spends more than 30 min in a radius of 200m could be defined as such a significant location. However, finding the right threshold values isn't a trivial task because these depend on the situation in which the person finds himself. Consider for example a person who is stuck for hours in

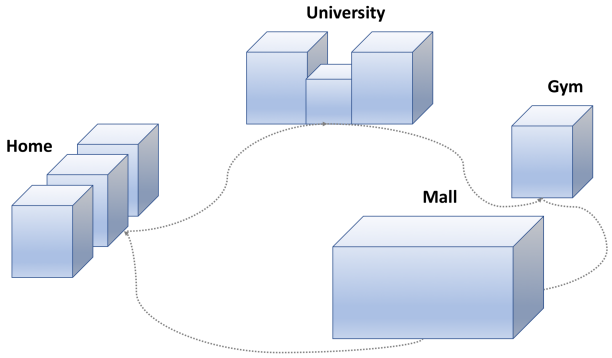

Figure 1.2: A typical semantic trajectory.

a traffic jam.

Within the scope of mining and analyzing human trajectories, Alvares et al. went one step further and introduced in 2007 a model that not only identifies, but enriches these locations semantically as well [ABK+07]. This is done by integrating data from thrid party geographic sources. The use of semantics provides the available trajectories with a certain degree of transparency and understandability and supports in this way respective query and analysis processes. The enriched trajectories are referred to as *semantic trajectories* and the degree of enrichment may vary: from simple labelling like *stop* and *move* to location types like *home*, *office* and *restaurant* up to including additional information like the locations' properties (e.g., *opening hours*) and the users' activity at the respective locations building in this way so-called *semantic episodes*. Thus, semantic trajectories reduce the common million-points GPS recordings into the essence of human movement represented by a set of few *semantic locations*. Spaccapietra et al. define a similar conceptual view upon raw trajectories in [SPD+08]. A typical semantic trajectory in its simplest form can be found in Fig. 1.2.

A few years later, in 2011, Ying et al. were the first to use semantic trajectories in a location prediction scenario [YLWT11a]. One of the greatest benefits when using semantic trajectories is that by going beyond mere GPS data and by capturing human movement at a higher conceptual level, a system is capable of learning these high-level patterns and consequently providing estimations about future locations even in regions where the user has never been before (and exist therefore no GPS data for the model to be trained on). This can

be achieved by projecting the learned semantic location patterns back to the raw trajectories. Moreover, using semantic trajectories for modelling human movement provides a much deeper insight into human behaviour. In general, movement data reveal beside information about the movement itself, information about the underlying mechanisms that trigger the particular movement [Dod16]. Semantically enriched data help us better understand these underlying mechnaisms. A better understanding of human behaviour lays in turn a solid foundation for a more accurate location prediction model. At the same time, modelling semantic trajectories and predicting the future semantic location of users can lead to a significantly higher level of personalization. Despite the aforementioned benefits, very little research has been done in this field to date, a fact that can be mainly attributed to the fact that location prediction on semantic trajectories is a relative new research field as we saw before.

This thesis aims primarily at filling this gap and *explores a wide range of ways for modelling semantic trajectories and predicting future semantic locations*. The here presented study comprises several probabilistic and neural network based models. These include single- and multi-dimensional Markov Chains as well as feed-forward, convolutional and recurrent artificial neural network architectures. Their performance is evaluated against the performance of state of the art approaches like the aforementioned Ying et al.'s framework using a number of real-world validation data. Depending on the investigated model's individual needs, a part of them can be evaluated using open, already existing datasets and for some it was unavoidable to collect our own data by designing and conducting a respective series of user studies. Apart from this high-level objective, this thesis investigates and attempts to provide an answer to the following particular points:

- *Semantic representation level*: Semantic trajectories can be semantically described in different ways, that is, at different semantic representation levels. Each level provides a different view upon the trajectories, which can be sometimes more and sometimes less abstract. This thesis evaluates whether and to what extent the semantic representation level affects the quality of semantic location prediction.

- *Degree of semantic enrichment*: The degree of semantic enrichment of

trajectories may vary depending on the number of information that is taken into consideration. In this work, we go beyond mere location types and explore incorporating additional dynamic features into the trajectories as well, with the aim of achieving higher prediction scores. These additional features can be divided into context- and user-specific ones. While the context-specific features cover information such as the activity of the user, the time of day and the day of week, the user-specific features orient towards personal characteristics such as the personality and the emotional state or mood of the user. We hypothesize, that taking this kind of features additionally into account would help the model to better adapt to the user and his behaviour, raise the personalization level and thus improve the overall prediction.

- *Semantic-enhanced learning*: In the literature, one comes across a number of methods for utilizing machine learning to support and optimize knowledge engineering. In ontology learning, for instance, both probabilistic and machine learning models are often applied for clustering and matching concepts or deriving relations between them, e.g., through association rule mining [WLB11]. However, very few have laid their focus on the other direction, namely the exploitation of explicit, structured semantic knowledge for optimizing the training of machine learning models. This thesis introduces a number of methods for incorporating semantics into the training process and evaluates these in terms of both training and prediction performance using the example of semantic location prediction.

This thesis is organized as follows. First, Chapter 2 defines the notion of *semantic trajectories* and provides insights into the evaluation methods and measures applied in this work. Then, the rest of the thesis is divided in 2 parts.

The first part (Part I) focuses on the probabilistic approaches and includes the Chapters 3, 4 and 5.

Chapter 3 describes a semantic location prediction approach that has a Markov Chain model as basis. Starting with a standard spatial 1-dimensional model and extending it accordingly, this chapter investigates the impact of

the semantic representation level, as well as that of using additional context dimensions, such as the time, the day and the users' current activity. It can be shown that adding context information can lead to a higher accuracy.

In Chapter 4 we propose a dynamic Purpose-of-Visit-dependent location clustering approach to support the predictive performance of the multi-dimensional Markov-Chain model of Chapter 3. The respective approach relies on a semantic similarity analysis of the location classes contained in our training dataset and is able to outperform the latter, provided that the available training data are large enough to cover a user's daily patterns and consistently annotated.

Finally, Chapter 5 discusses the Matrix Factorization technique (MF) as a means for predicting the next semantic location. The underlying idea of using Matrix Factorization is to take advantage of its strength in overcoming poor performance issues due to sparse data. It can be shown that a factorized Markov Chain model is indeed capable of outperforming both a standard Markov and a vanilla Matrix Factorization model.

In parallel with the investigation of the probabilistic models described in Part I, this thesis explores in Part II the use of Artificial Neural Networks for predicting the users' future semantic locations. It consists of Chapter 6, 7, 8, 9 and 10.

In Chapter 6, we compare three different network architectures, the Feed-Forward Neural Network (FFNN), the Recurrent Neural Network (RNN) and the Long Short-Term Memory network (LSTM) at two semantic trajectory representation levels. The outcome favours in the case of a general multi-user model both recurrent architectures, RNN and LSTM. However, all three models show deficits when it comes to modeling small single-user datasets and are being outperformed by a Markov Chain model. Solely the LSTM approach performs better than the Markov model at the highest representation level. All three models serve as baseline for the next chapters.

Chapters 7 and 8 address the use of Convolutional Neural Networks (CNN) and Sequence to Sequence (Seq2Seq) learning respectively for modelling semantic trajectories. While the CNN based approach shows with regard to accuracy a significant improvement compared to the FFNN, RNN and LSTM models of Chapter 6 the Seq2Seq Learning model can only show a slight pos-

itive impact on the accuracy. However, both approaches can at most reach similar or lower F-Score values than the ones of the LSTM model. At the same time, both Chapters highlight the benefits, which an additional Embedding layer may bring.

Chapter 9 is looking into the potential of taking additional personal characteristics, such as certain personality traits as well as the emotional and mental state, into consideration aiming at improving the accuracy of a LSTM-based prediction model. For this purpose, we extend the PoVDF approach of Chapter 4, build so called Location-Specific Cognitive Frames (LSCF) and feed them to a LSTM model. The evaluation results provide strong evidence that this kind of psychological features can lead to a better predictive behaviour. However, not every single information type leads to a similar enhancement of the model, especially when the available dataset does not cover each of them equally.

In Chapter 10, a number of ways for optimizing the training of neural networks by incorporating explicit semantic knowledge into it are proposed and evaluated using the example of semantic location prediction. In particular, we attempt to optimize the training and in turn the predictive behaviour of the LSTM model of Chapter 6. It can be shown that this kind of *Semantic-Enhanced Learning* results in shorter training times while keeping the test accuracy at the same level or even improving it.

Finally, Chapter 11 summarizes the findings of this thesis and discusses potential future work. Fig. 1.3 shows an overview of all explored approaches and methods in this thesis.

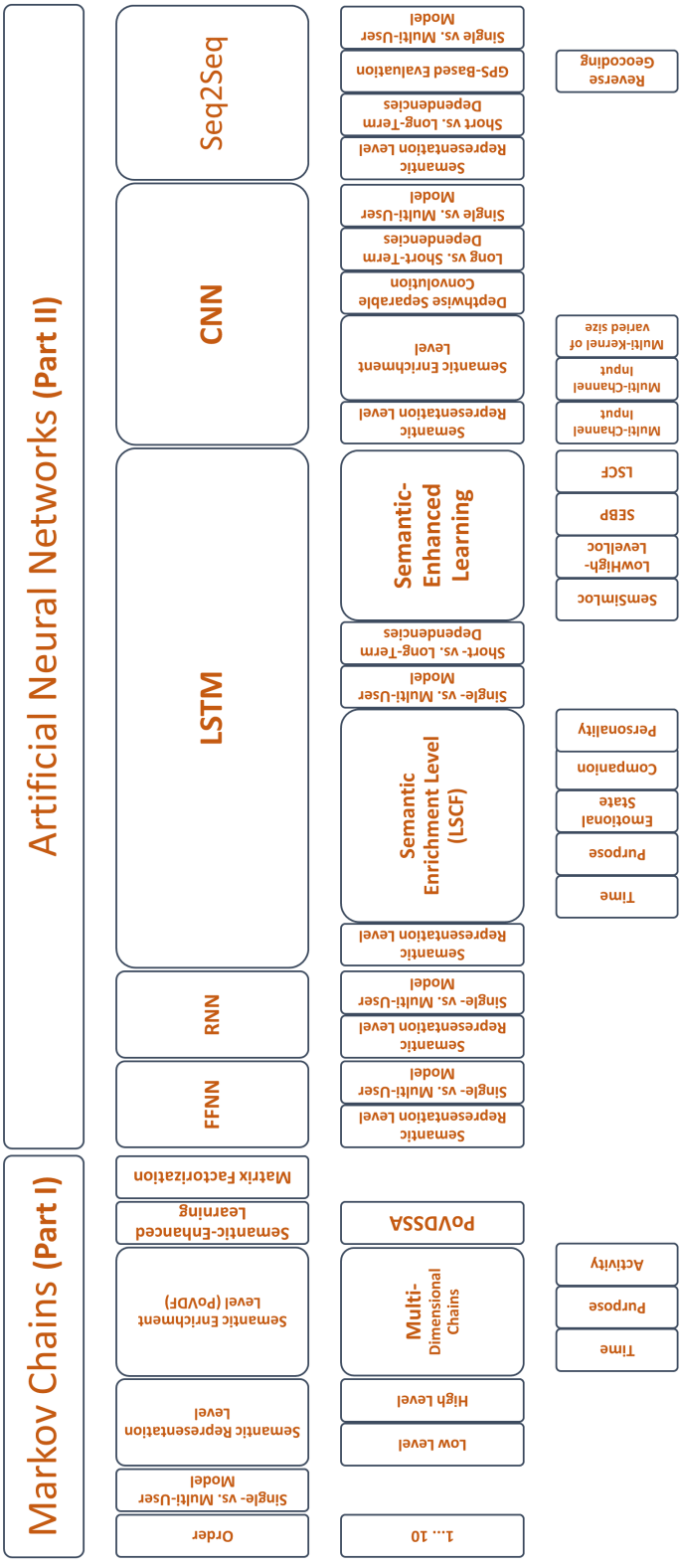

Figure 1.3: Overview of all explored methods in this thesis.

CHAPTER 2

BACKGROUND THEORY

2.1 Introduction

This chapter discusses the beginnings of *semantic trejectories* and provides a corresponding definition. The second part of the chapter addresses the evaluation methods and metrics applied in the presented thesis. The content of this chapter is based partly on an invited article in the Journal Sensors 2018, Special Issue Context and Activity Modelling and Recognition [KKB18a].

2.2 Semantic Trajectories

The term *trajectory* refers to spatio-temporal sequences, which define the movement of objects or persons under a given reference system (*frame*) during a certain time interval. GPS or Cell Tower ID sequences are such sequences. Equation 2.1 shows a common GPS trajectory, where each point in the trajectory is being described by a quadruple containing its latitude (lat_i), its longitude ($long_i$), its altitude (alt_i) and the corresponding time (t_i):

$$traj = (lat_1, long_1, alt_1, t_1), (lat_2, long_2, alt_2, t_2), ... \tag{2.1}$$

But in order to understand the logic behind such movement patterns and the behaviour of the travelling objects themselves, and to be eventually capable of predicting their next steps and act upon them accordingly, we need to go beyond plain position data and to acquire more knowledge about the respective trajectories. Alvares et al. and Spaccapietra et al. highlighted first the importance of utilizing *semantics* when analyzing trajectories in [ABK+07] and [SPD+08] respectively and introduced a *conceptual view* upon them. In their work, they point out the varying underlying purpose and the overall semantical meaning of trajectories. At the same time, they defined basic semantic elements such as *stops*, *moves*, *begin* and *end*. Each item in a semantic trajectory represents a *significant* location ([AS03]), a location at which a user is staying for more than a certain amount of time to carry out a certain activity, whereas the type of the activity affects the size of the respective time window. The notion of these *semantic locations* and their role in human mobility patterns conforms in a way with Hägerstrand's concept of a *space-time prism* defined by a set of so called *space-time anchors* [Häg70] that describes the trade-off between time and space depending on the respective activity.

However, Hägerstrand's approach is physical and views human mobility from a constraint satisfaction problem solving view. That is, it handles additional context information, such as activity, speed and the users' mobility capability, explicitly as constraints in order to derive the potential paths of a human user. This is done in the case of semantic trajectories and in our work in particular rather implicitly as we shall see later. In [YCP+13], Yan et al. define a *semantic trajectory* as

> *"a structured trajectory where the spatial data (the coordinates) are replaced by geo-annotations and further semantic annotations"*

resulting to a sequence of *semantic episodes*:

$$traj_{sem} = se_1, se_2, se_3, ... \tag{2.2}$$

An example of a semantic trajectory describing a user's daily routine would be for instance:
Bob is at home in the morning eating breakfast→ drives with his car to work → working (at the office) → drives to the gym in the afternoon → training at the gym → visits a restaurant in the evening where he meets his girlfriend → eating, drinking and socialising at the restaurant → leaves the restaurant and is heading towards home → stops at a super market to buy milk and eggs → drives back home → home.
Fig. 2.1 below illustrates this particular semantic trajectory.

2.3 Evaluation Methods and Metrics

The semantic location prediction problem represents a *multi-class* classification task (in contrast to the *binary* and the *multi-label* classification task) [SL09]. In this work, we use the following, for multi-class classification purposes typical, metrics:

Macro-Average Accuracy

The *macro-average accuracy* refers to the average *per-class* accuracy:

$$Accuracy_{macro} = \frac{1}{N} \cdot \sum_{l=1}^{N} \frac{TP_l + TN_l}{TP_l + FN_l + FP_l + TN_l}, \tag{2.3}$$

Figure 2.1: Example of an 1-day long Semantic Trajectory. *(Image based on [San19])*

with TP_l being the true positive, TN_l the true negative, FN_l the false negative and FP_l the false positive estimations of the classifier for a certain class l among a total of N classes. In contrast to *micro-average accuracy*, which is calculated over the cumulative sum of TP_l, TN_l, FN_l and FP_l and thus favours the classes that occur more often in a dataset, the macro-average accuracy treats all classes equally. The *macro-average accuracy@k* is a variant of the aforementioned accuracy metric, which returns the accuracy of the top k classes [BCMR12]. It is commonly used in recommendation systems.

Macro-Average Precision

The *macro-average precision* returns the average per-class precision, that is, the average over the number of all corrected instances given all the predicted labels for a given a class. It is defined by the equation below:

$$Precision_{macro} = \frac{1}{N} \cdot \sum_{l=1}^{N} \frac{TP_l}{TP_l + FP_l} \tag{2.4}$$

TP_l, TN_l, FN_l and FP_l represent as before the true positive and negative as well as the false negative and positive estimations respectively.

Macro-Average Recall

Analogously, the *macro-average recall* value represents the average per-class number of correct predictions over all the estimations that should be selected as correct ones and is defined as follows:

$$Recall_{macro} = \frac{1}{N} \cdot \sum_{l=1}^{N} \frac{TP_l}{TP_l + FN_l} \tag{2.5}$$

Macro-Average F-Score

The *F-Score* describes the relation between the data's positive labels and those estimated by a classifier [SL09]. It is calculated as a weighted mean of precision and recall. Analogously, the *macro-average F-Score* is the weighted mean of the *macro-average* precision and recall and is given by the following equation:

$$F - Score_{macro} = \frac{(\beta^2 + 1) \cdot Precision_{macro} \cdot Recall_{macro}}{\beta^2 \cdot Precision_{macro} + Recall_{macro}} \tag{2.6}$$

The β coefficient determines the relevance of precision over recall. Usually, as in this thesis, the β is selected to be 1 and the F_1-Score becomes the harmonic mean of precision and recall:

$$F_1 - Score_{macro} = 2 \cdot \frac{Precision_{macro} \cdot Recall_{macro}}{Precision_{macro} + Recall_{macro}} \tag{2.7}$$

In this thesis, we will be always referring to the F_1-Score when using the term F-Score. In addition, the Scikit-learn library offers a *weighted* F-Score variant that determines the average F-Score value weighted by support, that is, the number of the available samples per class [PVG+11]. This aims at further counteracting dataset imbalance.

2.3.1 k-Fold Cross Validation

Cross validation represents a method for evaluating machine learning models by dividing the available dataset into a training and a testing dataset. In *k-fold* cross validation, the available dataset is randomly splitted into k subsamples

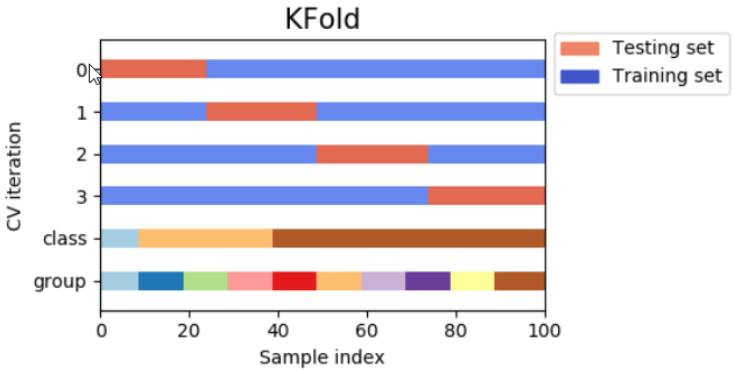

Figure 2.2: k-Fold Cross validation. (Source: `http://scikit-learn.org/`)

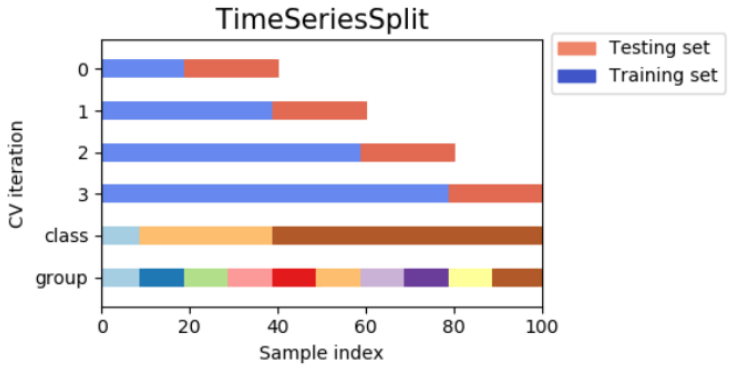

Figure 2.3: Time Series Split validation. (Source: `http://scikit-learn.org/`)

of equal size. Then, a single subsample is selected as a testing dataset and the rest $k-1$ subsamples are used for training. This process is repeated k times until each subsample can serve as a testing dataset (see Fig. 2.2). *Stratified* k-fold cross validation is a variant of cross validation that controls the subsample selection process in order to produce subsamples with a similar class distribution. Although having balanced subsamples is usually a good think when training classifiers, the stratified alternative is for sequence learning like in our case rather inappropriate because it hurts the temporal coherence of the treated sequences. The same holds for k-fold cross validation variants that shuffle the data at each training cycle.

2.3.2 Walk Forward Validation

Walk Forward validation is a validation technique appropriate for time series forecasting. At each run, it uses successively new available data for training in

order to predict future values. When applied on forecasting upon time series, it is common to perform a training every time new data arrive. In this work we use occasionally a slightly different version called TimeSeriesSplit()[1] that handles subsamples and is thus closer to the aforementioned cross validation method. Fig. 2.3 illustrates the respective process.

[1] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html

PART I

MARKOV CHAINS

This first part of this thesis covers a number of probabilistic approaches based on Markov Chains for modeling semantic trajectories.

Chapter 3 starts with the description and evaluation of a 1-dimensional spatial Markov model and highlights the importance of using additional context information, such as time and activity, by extending the model's dimensions accordingly. In addition, it examines the impact of the trajectories' semantic representation level on the predictive performance of the respective models. It can be shown that the higher the representation level, the better the performance in terms of accuracy and F-Score.

Chapter 4 proposes a semantic similarity analysis as means for supporting the multi-dimensional model of Chapter 3. The results in this Chapter indicate that integrating the semantic similarity analysis into the training and prediction process can lead to an overall improvement of the latter, a fact that motivated among others the investigations described in Chapter 10.

Finally, Chapter 5 explores the use of Matrix Factorization for modeling semantic trajectories and predicting the users' future semantic location. In particular, it evaluates a total of 3 different variants: a simple Matrix Factorization model, a combination of Matrix Factorization and Markov Chains as well as a user-centered variation of the latter. It can be shown that the user-centered approach is able to outperform both the first two models as well as the spatial Markov model of Chapter 3.

CHAPTER 3

MODELING SEMANTIC TRAJECTORIES USING MARKOV CHAINS: 2-LEVEL SEMANTICS AND MULTIPLE DIMENSIONS

Abstract

In this chapter, we investigate the performance of Markov Chains with respect to modelling semantic trajectories and predicting future semantic locations. In the first part, we examine whether and to what degree the semantic level of semantic trajectories affects the predictive performance of a spatial Markov model. It can be shown that the choice of the semantic level when describing trajectories has a significant impact on the accuracy of the models. High-level descriptions lead to better results than low-level ones. The second part introduces a multi-dimensional Markov Chain construct that considers beside locations additional context information, such as the time, the day and the users' activity. While the respective approach is able to outperform our baseline, we could also identify some limitations. These are mainly attributed to its sensitivity towards small-sized training datasets and the associated so-called curse of dimensionality effect due to the larger feature set. Chapters 4 and 5 propose a solution against this limitation.

The content of this chapter is based on two publications, at ICANN 2017 [KSJB17] and at PerCom CoMoRea Workshop 2018 [KKB18b], and on an invited article in the Journal Sensors 2018, Special Issue Context and Activity Modelling and Recognition [KKB18a].

3.1 Introduction

Probabilistic models such as Markov Chains and Bayesian models represent one of the first and rapidly established methods for modelling trajectories and predicting future locations. Markov models are especially common among the respective literature due to their computational efficiency and provide a solid basis for accurate predictions about future movement patterns of mobile users. Some use Markov models to model solely spatial information and some include additional context information into their models as well. The next section provides a brief insight into some of the most related work that utilize various types of Markov models in order to predict the next location of a user.

3.2 Related Work

Ashbrook et al. and Gambs et al. investigated in their work the performance of traditional Markov models with regard to location prediction accuracy [AS02, GKPC12]. Their results show that the order of the model may have a significant impact on the accuracy. High order models seem to perform better than lower order ones. However, choosing a too high order affects the model adversely. Asahara et al. demonstrate in [AMSS11] that a mixed Markov model (MMM), which considers users with similar movement behaviour to fall within respective groups, performs better than a simple Markov model (MM) and a Hidden Markov model (HMM). Ye et al. adapt and apply in their work Altman's mixed Hidden Markov model (MHMM) [Alt07] on semantic locations coming from a Location Based Social Network (LBSN) [YZC13]. Their mixed model incorporates both spatial and temporal information by (statistically) mapping location categories to a number of predefined time periods (e.g., *bar* with *night*). In addition, the users are clustered based on their movement and activity patterns. Finally, a separate model is trained for each cluster of users. It is shown that time in the form of coarse time windows as an additional context information, as well as the clustering of similar users helps to raise the prediction accuracy. Wesley et al. take in [MRM12] temporal beside spatial information into account as well. They propose a triple Hidden Markov model (HMM), that is, a set of 3 HMMs, with each modelling spatial information

coming only from one of the following temporal periods respectively: weekdays from 7am to 7pm, weekdays from 7pm to 7am, and weekends. Similarly to Ye et al., they show that this kind of *temporal attention* can lead to an improved prediction accuracy.

3.3 Markov Chain Model

The term *stochastic process* refers to an ordered collection of one or more random variables and is used usually to describe dynamic processes that change over time at random. A *Markov Chain* model (or simply *Markov Chain* or *Markov model*) defines a memoryless stochastic process; a stochastic process, which additionally satisfies the Markov property. According to the Markov property, predictions for the future based on a short history lead to similar results to those based on the whole history. Markov Chains are categorized by their order depending on how far back history is taken into account. A 1. order Markov Chain is determined by the following conditional (Markov) property [Bis06]:

$$p(z^{(m+1)}|z^{(1)}, z^{(2)}, ..., z^{(m)}) = p(z^{(m+1)}|z^{(m)}), \tag{3.1}$$

whereby $z^{(1)}$, $z^{(2)}$, ... is a series of successive random state variables (or just *states*) and p represents the *state transition probability* value. Thus, the prediction relies in this case solely on the current state and is independent from the former ones. A 2. order Markov model would analogously consider both the current, as well as the previous state, and so on. Higher order Markov Chains tend therefore to cluster the considered previous states together. In this work, the states correspond to semantic locations $L = \{l_1, l_2, ...\}$ and the Markov Chain model is used to model the respective semantic trajectories T_{sem} and subsequently to predict the future movement behaviour of the users U = $\{u_1, u_2, ...\}$.

In general, a Markov model can be described through its *state transition probability matrix*, or simply *transition matrix A*. Each element of the transition matrix contains the probability for changing from a certain state to another. In our case, a state transition refers to moving from a certain semantic location to another location. Fig. 3.1 illustrates an example of such a

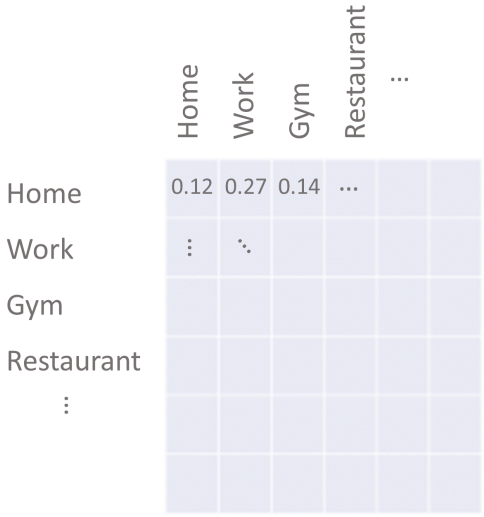

Figure 3.1: Semantic location transition probability matrix.

semantic location transition matrix. Each row and column represents a single semantic location. According to this matrix, there is a 27% chance for a user being at home to visit the gym next, while there is only a 12% chance to stay home. The transition matrix can refer either to a single user or to a set of users.

3.4 Applying Markov Chains on Semantic Trajectories of Different Semantic Level

As we saw in Chapter 2, semantic trajectories can be described in various ways depending on the semantic representation level used at each time. Moreover, the choice of the semantic level determines the modelling granularity. This section explores the impact of the semantic level on the modelling performance of a 1. order spatial Markov Chain model. In particular, in this section, we compare and evaluate two different semantic levels.

In order to define these two levels, we oriented ourselves on the Foursquare venue taxonomy[1]. The Foursquare venue taxonomy defines a 5-level hierarchical construct of location categories with the following 10 major top-level categories being at the highest level:

[1]https://developer.foursquare.com/docs/resources/categories

- Arts (Culture and Entertainment)
- College and University
- Events
- Food
- Nightlife
- Outdoors (Nature and Freetime)
- Professional and Other Locations
- Residence
- Shop and Services
- Travel and Transport

Each of the top-level categories comprises a set of subcategories like the one seen below for the *Food* category case:

- Food location
  - Greek restaurant
    - Ouzeri
    - Tavern
      - Fish tavern
      - ...

Here, in this section, we compare the top semantic level ($Level^{(1)}$) with the third level ($Level^{(3)}$), e.g., *food location* vs. *burger joint* or *Nightlife* vs. *Bar*. The choice fell on these particular levels in part due to the range and the distribution of the annotated location types found in our training and evaluation datasets, which are described below, in the evaluation Section 3.4.1.

3.4.1 Evaluation

We used 2 different real-world datasets in our evaluation, a 12-week long single-user dataset and a 8-week long dataset with 21 participants. The first dataset is diary-based and contains the annotated daily trips of a single user during a period of 3 months. The second dataset is based on an experimental field study, in which 21 mobile users were tracked for a period of 2 months and is described in detail in Chapter 9. Our datasets include all 10 $Level^{(1)}$ location categories and a total of 70 different $Level^{(3)}$ location types.

In order to evaluate the Markov model in terms of prediction performance with regard to the semantic level, we applied a 10-fold cross validation (see Section 2.3.1) for each semantic level. Table 3.1 provides the average and the maximum recorded accuracy of a 1. order Markov Chain model after 10 shuffled runs for the respective two semantic levels in the single-user case. It can be seen clearly that the Markov model performs better with high-level trajectories than with lower-level ones. It achieves an average accuracy of 75% and a maximum accuracy of 100% compared to the 61.1% and 94.2% of the low-level model respectively. This can be attributed to the fact that people tend to move in space based on rules and patterns of higher semantical order. For instance, a person might regularly visit a *food location* after going to *gym* regardless whether it is a *pizza house*, a *burger joint* or some *snack bar*. However, at the same time, a high-level scenario clusters finer location types together and brings therefore less classes for the Markov model to predict. For this reason, a better performance in favour of the high-level trajectory model can be additionally expected.

The results for the 21 multi-user dataset are set out in Table 3.2. Apart from the accuracy, Table 3.2 includes the weighted F-Score (see Section 2.3) as well. The results follow a similar trend as before, with the high-level model ($Level^{(1)}$) outperforming the low-level one ($Level^{(3)}$) both in terms of average and maximum accuracy. The same trend can also be identified with respect to F-Score.

	Avg Accuracy	Max Accuracy
$Level^{(1)}$	0.750	1.00
$Level^{(3)}$	0.611	0.942

Table 3.1: 10-fold cross validation average and maximum accuracy results of the 1. order Markov model for the 12-week single-user dataset for both the top ($Level^{(1)}$) and the lower semantic level ($Level^{(3)}$).

	Avg Accuracy	Max Accuracy	F-Score (weighted)
$Level^{(1)}$	0.385	0.559	0.386
$Level^{(3)}$	0.301	0.448	0.302

Table 3.2: 10-fold cross validation average accuracy, maximum accuracy and weighted F-Score results of the 1. order Markov model for the 8-week multi-user dataset for both the top ($Level^{(1)}$) and the lower semantic level ($Level^{(3)}$).

3.5 A Multi-dimensional Markov Chain Model for Modelling Semantic Trajectories

In this section, we propose a multi-dimensional Markov Chain model for modelling semantic trajectories and predicting future locations that takes both time and the user's activity additional into consideration.

For this purpose, we extend in a certain way the work of Wesley et al. [MRM12] and build an ensemble of Markov Chains with each of them corresponding to a specific feature configuration with regard to the feature's type and value range, to which we refer to here as *dimension*. In tangible terms, we define a multi-dimensional construct that comprises a set of semantic location transition probability matrices, one for each tuple combination *(time of day, day of week, activity)*. We regard time of day in 24 hourly time slots. Additionally, we clustered all annotated activities found in our training and evaluation dataset (see Section 3.5.1) into the following 12 high-level activities based on their content and frequency of occurrence:

- Working
- Training

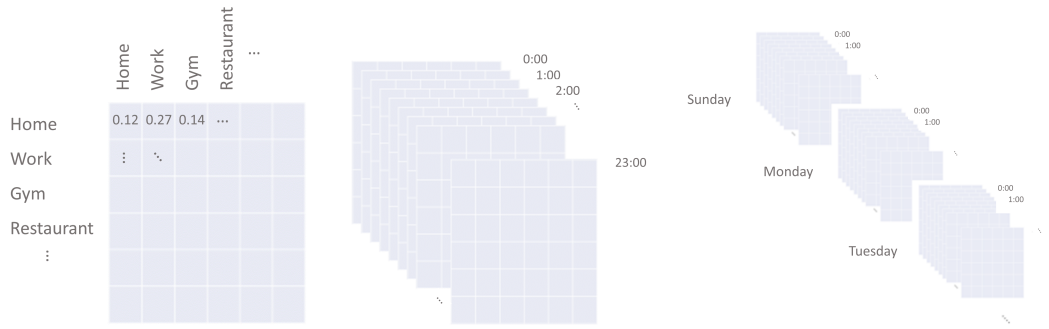

Figure 3.2: From left to right: Location transition probability matrix, Time-of-day-specific transition probability matrices, Day-of-week-specific sets of transition probability matrices.

- Trading
- Creating
- Shopping
- Socializing
- Celebrating
- Eating
- Travelling
- Getting ready to go
- Relaxing
- Visiting a doctor

Fig. 3.2 illustrates on the left an example of a location transition matrix as we had in Fig. 3.1, while the middle and the right part elucidate vividly the multi-dimensionality of our model regarding time of day and day of week ($24 \times 7 = 168$ matrices). If we consider the 12 activities as well, we come up to a total of $24 \times 7 \times 12 = 2016$ transition matrices. This would require a great amount of data to to cover all the possible combinations and get the elements of all transition matrices filled. And even if we had so much data,

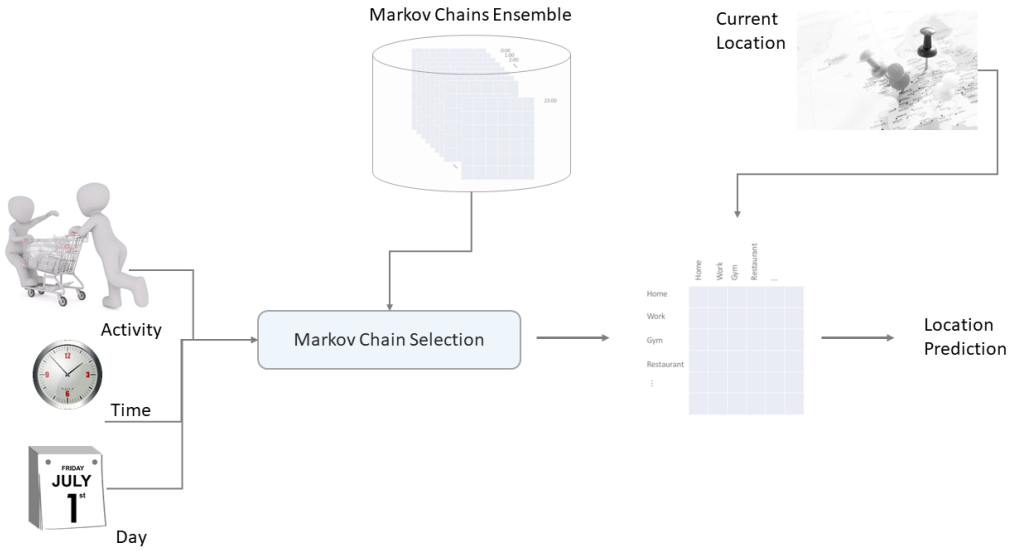

Figure 3.3: Markov Chain selection process and location prediction.

some combinations might still not exist (e.g., *(3am, Tuesday, doing fitness)*). In order to increase our chances of obtaining reasonably occupied transition matrices, we clustered the time into the following three time blocks:

- Morning: 5:00am - 11:59am
- Midday: 12:00pm - 16:59pm
- Evening-Night: 17:00pm - 4:59am

This results in a final total of $3 \times 7 \times 12 = 252$ transition matrices, that is, a total of 252 Markov Chains. Each of them is trained separately and covers a specific context scenario. During the prediction phase, our model picks out the appropriate Markov Chain depending on the current context (time, day, activity) and estimates the next semantic location. Fig. 3.3 displays the Markov Chain selection process based on the additional context information and the subsequent location prediction based on the current location of the user.

3.5.1 Evaluation

This section covers the outcome of our multi-dimensional Markov Chain approach. While the first part refers to our training and evaluation dataset and

how this was collected, the second part discusses in detail our evaluation results.

User study

In order to evaluate our multi-dimensional Markov Chain ensemble on a real-world dataset, we carried out a user study, in which we tracked the movement of 10 mobile users for a period of 5 weeks. Our participants aged from 18 to 28 years old and were mostly students. A tracking and annotation Android app was designed and implemented in order to collect the users' data. For this purpose, we used the AWARE instrumentation and context logging framework[2], which brings many benefits, such as an activity-based energy efficient and phone battery saving tracking algorithm. This is particularly important, since a power hungry app could affect adversely our study in two ways. On the one hand, it could lead to users closing the app and missing therefore valuable data. On the other hand, it might even influence their movement patterns, if the users had to plan more time somewhere in-between for charging their smartphone. So, building an energy efficient app was a basic prerequisite for us. Fig. 3.4 shows a screenshot of our Android app. The users were able to pause or close completely the app at any time. Each user was assigned an anonymous and random generated identification number (ID) in order to preserve the users' privacy. The data were collected and stored first locally, on the mobile phone of the user, and sent encrypted to our server at the end of the study.

During the study we asked the users to:

- Label their current location
- Enter the purpose of visiting the certain location (high-level activity)
- Define whether the particular location had been visited before for another purpose up to that point
- Rate how important the particular location is to them, and
- Provide additional descriptive information, like "sitting in a coffee shop with a friend after work", etc.

[2]http://www.awareframework.com

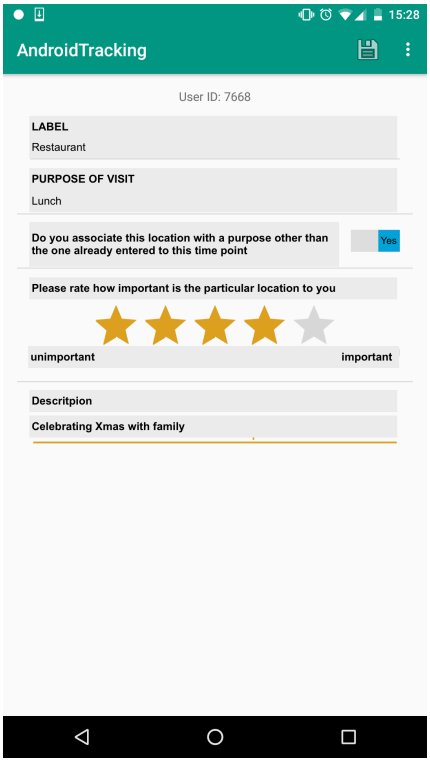

Figure 3.4: Screenshot of our Android tracking and annotation app.

Furthermore, 3 Amazon vouchers were raffled among the participants in order to increase their motivation.

Results

Our user study provided us with an average of approximately 4000 GPS entries per user. However, the number of annotations varies strongly among the users. Fig. 3.5 contains the distribution of the number of the tracked GPS points, as well as the number of the *annotated* tracked GPS points (significant locations) among the users. It can be seen that users 3, 5, 7 and 8 provided the least annotations. For this reason, we filtered these users out and our evaluation focused on the remaining 6 users. After analyzing their data, we came all in all to 431 different purpose-of-visit entries with respect to the following 9 high-level location types: *Residence/Home*, *Travel and Transportation*, *Nightlife*, *Shopping*, *Services*, *Food*, *Freetime*, *Education/University* and *Work*. As before, we oriented ourselves on the Foursquare location taxonomy described in Section 3.4 in order to categorize the users' locations. Fig. 3.6 shows the distribution

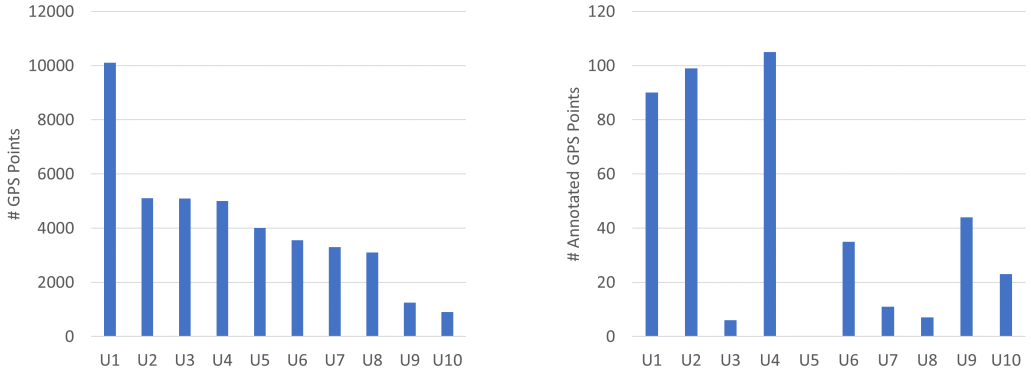

Figure 3.5: Left: Number of tracked GPS points per user. Right: Number of annotated locations per user.

of all annotated locations during our user study among all users.

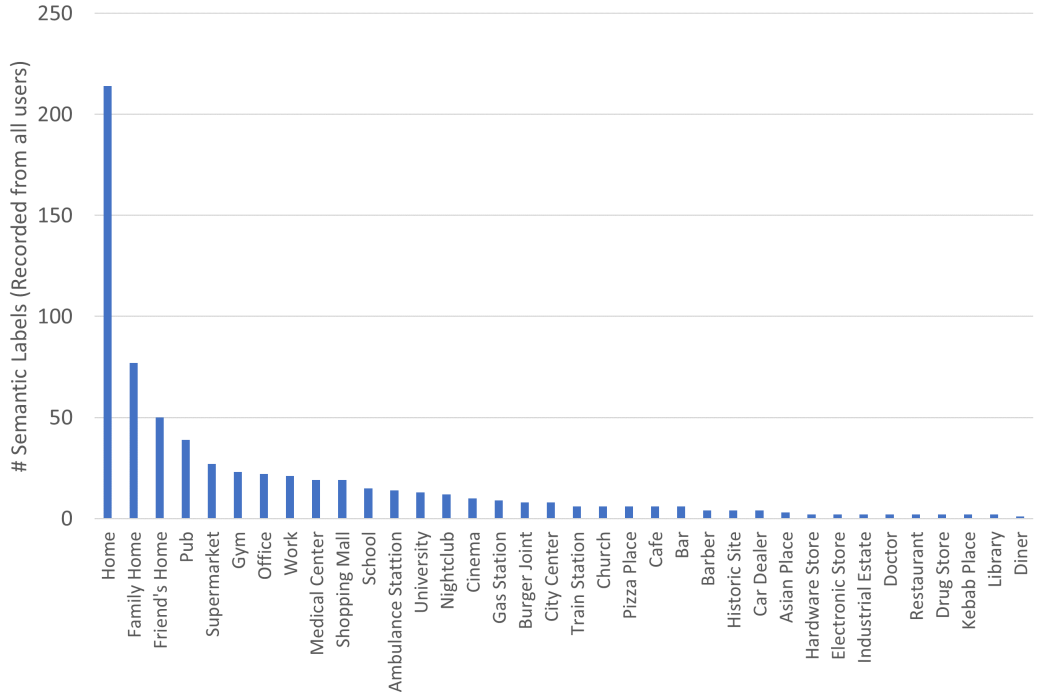

Figure 3.6: Semantic label distribution of locations.

What is interesting about the data is that often the same location type comes with more than one purpose of visiting that location. That is, users often entered different high-level activities for the same location based on the respective context each time. Fig. 3.7 illustrates this in the case of the lo-

cation type *food*. We can see that the users entered 21 different reasons for

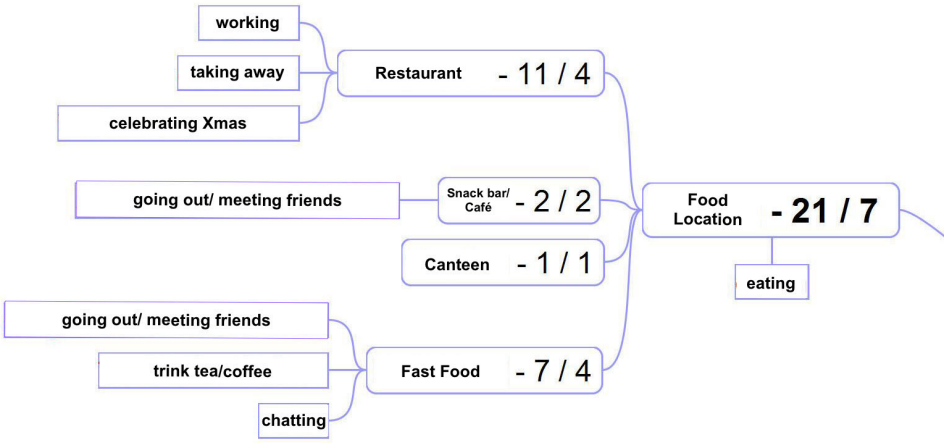

Figure 3.7: Multiple purposes of visiting a food location (beyond eating).

visiting a food location (including the respective sublocations). Apart from the obvious purpose “eat and drink", some participants visited a restaurant just shortly to get some take away food, some were working there and some visited a restaurant for socializing reasons to meet with friends and/or celebrate Christmas. It is obvious that each of the aforementioned purposes of visiting a certain place influences the visiting duration and attendance times in general, which in turn affects the overall movement patterns of the users. This supports our primary idea of using the users' high-level activity to extend the spatio-temporal Markov model. A person working in a restaurant shows a regular visiting pattern in specific days and times, while a person who is just passing by to get some take away food will stay there just for a few minutes (ideally). Furthermore, based on common sense knowledge, one could conclude that the latter person would probably take the food and go directly home instead for instance of visiting a club. Fig. 3.8 shows the proportion between the total number of the various (unique) semantic locations per user to the number of the locations at which more than one purposes of visiting that particular locations was entered.

We chose the k-fold cross validation to be our evaluation method and tested the following k-parameter values: [5, 10, 15, 20]. The results in this section refer to a k-value of 20. In addition, we compared our multi-dimensional 1. order Markov Chain model with the semantic trajectory based approach of

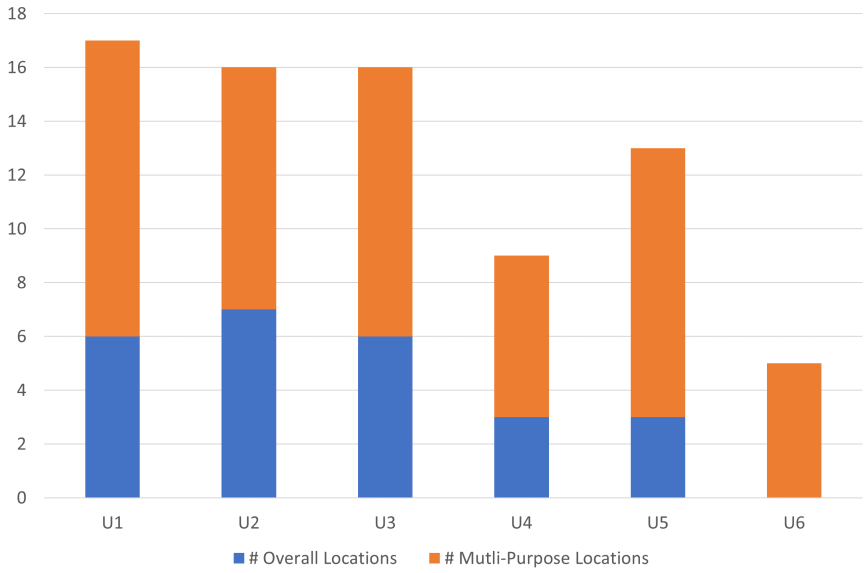

Figure 3.8: Number of (unique) multipurpose locations proportional to the total number of (unique) locations per user.

Ying et al. [YLWT11b] with a minimum support of 0.01. Fig. 3.9 shows how the multi-dimensional Markov Chain model performs against Ying et al.'s with regard to accuracy, precision, recall and F-score. We can see that the Markov model outperforms Ying et al.'s framework in all 4 cases. The absolute values are however generally low. This could be mainly attributed to the small size of our training dataset. The collected data were not able to cover all the possible location transitions given certain additional features contained in each Markov Chain, which leads to a reduced performance.

After evaluating separately the individual models for each user, the multi-dimensional approach could show its advantage against a simple one-dimensional Markov model for the cases in which the users provided the most data (users 1,2 and 4) reaching an up to 8% higher accuracy. For the remaining users, there was no improvement to be recognized. On the contrary, in these cases, the simple single-dimension Markov model was able to outperform the multi-dimensional one achieving approximately up to 20% higher scores. This underpins the aforementioned assumption based on the lack of a bigger dataset. A solution for this issue is discussed in Section 4.

The approach of Ying et al. seems particularly sensitive to the size of the available data. This can be explained by the fact that their core algorithm

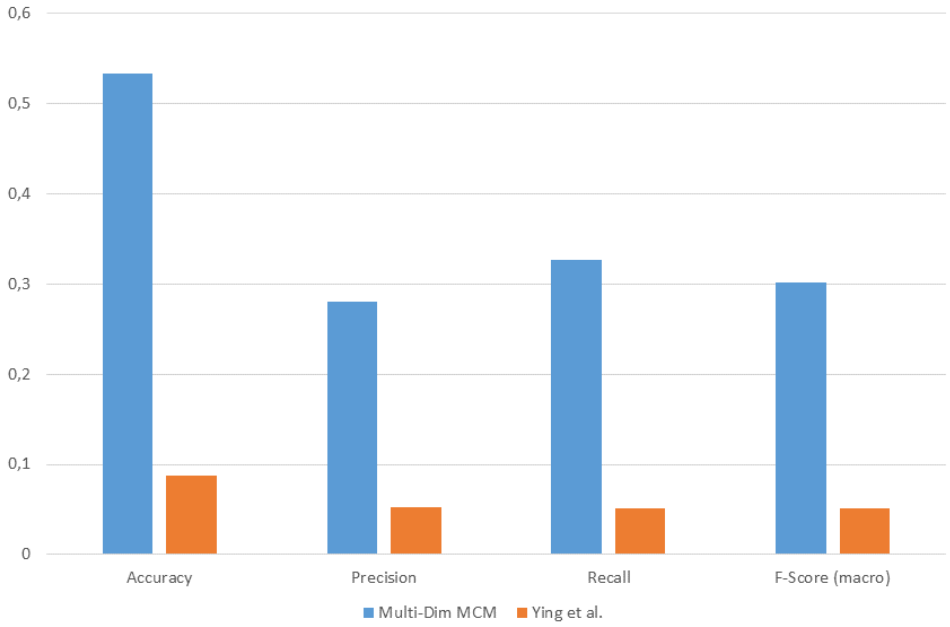

Figure 3.9: 1. order multi-dimensional Markov Chain model vs. Ying's approach with regard to accuracy, precision, recall and F-score.

is based on mining semantic trajectories out from the data based on their frequency of occurrence. In case of a small dataset, there are less trajectories to be found. Therefore, the necessary pre-fix trees on which their approach relies can't be reasonably built. This results in a constrained set of possible future locations from which their predictor can chose from and thus to a reduced performance.

3.6 Conclusion

In this chapter, we apply Markov Chains on semantic trajectories for predicting future locations. It consists of two parts. The first part discusses whether and to what extend the semantic level used for describing the locations affects the modelling and predictive performance of a 1. order Markov model. It could be shown that trajectories of higher semantic level can lead to more accurate predictions. On the one hand, this can be attributed to the fact that people's regular movements rely mainly on high-level patterns and rules. On the other hand, high-level trajectories contain less classes for the Markov

model to predict as the lower-level ones, a fact that supports additionally the prediction performance.

In the second part, we propose and describe a multi-dimensional context-aware Markov Chain construct for modelling semantic trajectories. Our approach takes, beside the current semantic location, the time of day, the day of week and the users' current activity additionally into consideration. We evaluated our idea on a real-word dataset and compared it with a state of the art semantic trajectory based algorithm. The multiple Markov Chain construct outperforms the baseline by yielding a higher performance in terms of accuracy, precision, recall and F-score. However, a certain sensitivity towards the small dataset size could also be identified. Chapter 4 proposes a solution against this limitation.

CHAPTER 4

SEMANTIC-ENHANCED MULTI-DIMENSIONAL MARKOV CHAINS: APPLYING PURPOSE-OF-VISIT-DRIVEN SEMANTIC SIMILARITY ON SEMANTIC TRAJECTORIES

Abstract

This chapter discusses a dynamic semantic-enhanced extension of the multi-dimensional Markov Chain construct introduced in Chapter 3 in an attempt, among others, to overcome the limitations of the latter due to the sparsity of the available training data. The presented approach takes the varying role of locations due each time to the respective purpose of visiting the particular location explicitly into consideration. We could say that this varying role matches in a way the users' perception when considering locations. For instance the *park* becomes a *fitness* area when you are jogging in it, a *food* location if you are having a picnic and a *working* location if you are working on your laptop.

In order to capture the aforementioned dynamics, we specify an entity, which we refer to as *Purpose-of-Visit-Dependent Frame (PoVDF)*, that incorporates time, location type and purpose of visit, within a respective ontology. Our framework is hybrid and combines both a data-driven, as well as a knowledge-driven model. To fuse these two models, we define a *Purpose-of-Visit-Driven Semantic Similarity (PoVDSS)* metric and use it as a fusing component between the two models. We evaluated our approach using the real-world dataset presented in Chapter 3.5.1. The multi-dimensional Markov model described in Chapter 3.3 and a semantic pattern mining based algorithm served as our baseline. Our evaluation shows that our PoVDF-based approach improves the location prediction accuracy of the model with the respective scores reaching up to 80%.

The content of this chapter is based on two publications, at Ubicomm 2017 [KB17] and at PerCom 2018 [KKB18b], as well as on an invited article in the Journal Sensors 2018, Special Issue Context and Activity Modelling and Recognition [KKB18a].

4.1 Introduction

Location information reveals to us humans more than just the *whereabouts*. It gives indirectly insight about the *what* and the *when*. While the *what* refers to the activity and the overall purpose of visiting a certain location, the *when* provides the temporal information, such as time, day and date, frequency and periodicity. For instance the location *night club* is put usually in context with a group of semantically high-level purposes, such as *socializing*, *having fun*, *chilling, relaxing*, and a set of elementary, lower-level activities, like *drinking*, *dancing*, *meeting friends and/or new people*. Moreover, a human would additionally associate some corresponding temporal information with it, such as *night*, *weekend*, *once a week* and maybe *all night long*. In order for us humans to be capable of interpreting locations at such a high-level and associating them with all this additional information, we rely both on a broad framework of semantics hidden behind them, as well as on a large portion of world and common sense knowledge. At the same time, *each human takes his/her own personal experience and knowledge into consideration*. This agrees also with the general definition of *knowledge*, which is defined in [oxf16] as:

> *Facts, information and skills acquired through experience or education; the theoretical or practical understanding of a subject*

A generic semantic framework together with a common sense knowledge base provide a mutual basis among different people for interpreting things similarly. In comparison, *personal experience can rather lead to different interpretations among people*. Let us clarify this in the location scenario by going back to the example mentioned before. In the particular example, the location *night club* was interpreted from the perspective of a guest, which is the most common one. But for the barkeeper, who may probably have to open the place at noon in order for the beverage suppliers to replace the empty bottles and carries the responsibility that everything works alright during the night, a club is a *working location*. It is bound now to a completely different high-level aim and set of elementary actions, like *working/earning money*, and *making drinks*, *serving drinks*, *talking to the customers* and *getting paid* respectively. A similar ambiguous effect would arise in the case of a restaurant between a guest and the cook or the waiter working there.

Let us now consider another example. The location *hotel* is for a tourist obviously closely linked to a stay location over the holidays, while for the receptionist it is a place of work with highly fixed attendance times. People, who are visiting a conference or are having a business lunch there, would also experience the hotel from a similar perspective during that time, since their visit is of professional nature. On the other hand, if the same people would enjoy a drink at the bar of the same hotel after their business meeting is over, they would associate the hotel rather more with a night life location, like a *bar* or a *club*. This example highlights another important issue, namely that people tend to perceive, interpret and associate locations to each other *dynamically, depending on the situation, in which they find themselves*. Nathan et al. support this theory with regard to human movement analysis by interpreting movement between locations as the outcome of the synergy of four components ([NGR+08]): the internal state of the individual, its motion capacity, its navigation capacity and potential external factors, whereby the internal state addresses the situation in which a person finds himself. This kind of location-specific "semantic ambiguity" can mostly be found in multi-purpose locations, a fact that was confirmed in the study presented in Chapter 3.5.1 as well. That is, locations, which offer a variety of reasons to visit them, such as the mall (*shopping*, *getting a haircut*, *meeting a friend*, *drinking a coffee*, ...), the hotel (as described in the aforementioned example), the park (*picnic*, *jogging*, *sitting on the bench reading a book*, ...), our home (*working*, *relaxing*, *celebrating*, *eating* (breakfast, lunch, ...), *drinking coffee*, *meeting friends*, *fitness training*, ...), etc.

Summarizing the above leads us to the following two aspects:

1. *The same location has potentially a different meaning to different people*
2. *A location may even have many different meanings to the same person depending on the situation*

Location prediction algorithms that utilize semantics and rely on so called *semantic trajectories* (see Section 2.2) go beyond plain numerical data, like GPS coordinates and Cell Tower ID sequences. The use of semantics gives them a number of advantages. The most significant one, is the fact that semantic trajectories carry more knowledge with them and are capable of capturing

the essence of human movement patterns. This can be particularly helpful for a location predictor in places that have not been visited before by the users and for which there are therefore no GPS recordings available on which the predictor could be trained. Beyond that, semantically enhanced location prediction systems gain transparency through the use of semantics. Due to the fact that the data collected and processed by the respective systems are human-understandable, the user has the opportunity to better understand how such systems work and why certain predictions come into effect. On the one hand, this leads to a better human-machine relationship. On the other hand, it assists indirectly the compliance with the data protection regulations, which in the meanwhile is vital for creating independent, fully autonomous and intelligent environments.

There exists a great number of location prediction models in the literature so far. However, up to this point and to the best of our knowledge, none of the semantic trajectory based approaches have been taking the *varying role and human perception of locations* into account. Instead, they constrain themselves to static semantic location types and inflexible associations between locations and users as described in the related work section below (Section 4.2). In this chapter, we introduce a semantic trajectory based location prediction approach that considers explicitly the dynamic, purpose-of-visit-driven varying role of locations in order to achieve a higher performance. Our approach relies on the hypothesis that

locations resemble one another in relation to the purpose of visit

and that

similar locations come also with similar location transitions as well.

In tangible terms, we hypothesize that a person who always visits a *take away restaurant* after visiting the *gym*, will also visit some *food location* too after climbing in a *boulder hall* or *jogging* at a *park*. While the *gym* and the *boulder hall* are associated with each other and belong to the same high-level location type *fitness location*, this holds only exceptionally for the *park* case due to the fact that it *shares the same purpose of visit*, namely *doing fitness*. The core idea of the approach presented in this chapter lies in a dynamic and context-aware clustering of semantic locations. For this purpose, we combine two different

modelling techniques, a *data-driven* and a *knowledge-driven* one, by using the *semantic similarity analysis* as a fusing component.

We used the collected dataset from the user study described in Section 3.5.1 to evaluate our approach. We can show that our framework is able to converge more towards human movement patterns and can therefore lead to a higher predictive performance compared to other semantic trajectory based approaches.

This chapter is structured as follows. Section 4.2 provides a brief summary and overview of the most relevant semantic-enhanced location prediction approaches, as well as some work related to the usage of semantic similarity and relatedness. Next, in Section 4.3 we describe in detail our approach, while in 4.4 we discuss thoroughly the outcomes of our evaluation and the overall performance of our framework. At last, Section 4.5 provides a short overview over the work in this chapter and summarizes our major results.

4.2 Related Work

This section discusses the most relevant research in the field of semantic-enhanced location prediction. The second part refers to some works that although they don't come from the same field, their semantic similarity and relatedness based approaches served as a basis for our own model.

In contrast to non-semantic methods, Ying et al. use in [YLWT11b] their own Geographic Semantic Information Database (GSID) to enrich semantically their recorded GPS or Cell Tower ID trajectories. GSID is a customized POI[1] database that stores semantic information of landmarks, geographic scopes and associated location types. From the resulting semantic trajectories, Ying et al. mine the most significant patterns, which in turn are converted into Semantic Pattern Trees that finally provide the basis for the next place prediction achieving in this way a higher accuracy. In [YLT14] they extend their approach by taking temporal information into account as well. Samaan et al. use spatial conceptual maps to describe buildings and road network elements semantically [SK05a]. In addition, a XML user context knowledge base, which contains the users' preferences, schedule, tasks and goals further

[1] Point of Interest

supports the location prediction. Their mobility prediction algorithm is probabilistic and relies on the Dempster-Shafer Theory [Sha92]. In [SKK04] and [SBKK05] they illustrate the same algorithm, only that now the locations are represented by Cell Tower IDs assigned by the corresponding cell towers. Ridhawi et al. apply a similar algorithm for tracking and predicting users indoors in order to support location-aware services in [RRKN11] and [RAKA09]. Their algorithm also uses the Dempster-Shafer Theory for returning the final future location estimation. However, in contrast to Samaan et al., the knowledge is structured and stored by means of OWL-based ontologies. These ontologies contain the profiles of the users, their location history and a group of activities. Long et al. in [LJJ12] propose a location clustering algorithm, which makes use of the Latent Dirichlet Allocation (LDA) method, a probabilistic model used normally to cluster documents based on the topics contained in them. They define so called *geographic topics* in an unsupervised manner from the check-ins at the corresponding venues of Foursquare[2] users based on their popularity. These topics replace static location categories like the ones provided by Foursquare. Additionally, they investigate how location clusters behave depending on whether it is a weekday or weekend, with the weekdays providing as expected the most regular basis. In [KKST15], Krishnamurthy et al. exploit Twitter[3] tweets to predict the location of its users. They introduce the concept of *localness* to express how "close" certain terms appearing in a tweet, so called *local entities*, are to particular cities. To this effect, they investigate several different measures, including two semantic relatedness measures, the Jaccard [Jac12] and the Tversky [Tve77] Indices. After determining the *localness* scores for each city of the corresponding local entities in the tweets of a user, it is possible to estimate the location of the user. Ye et a.l in [YZC13] use also data from Location-based Social Networks (LBSN). Their method is based on a Mixed Hidden Markov model (MHMM) and uses the semantic annotations of check-ins, i.e. the annotated location types as input. Finally, Wannous and Malki et al. propose a multi-ontology based approach in combination with a set of rules created by a group of experts to model and reason about movement and activity patterns of marine mammals

[2]https://foursquare.com/

[3]https://twitter.com

[WMBV16, MWBV12, WMBV13b, WMBV13a, WVMB15]. Their set of rules is subdivided into spatial, domain-specific and temporal rules respective each time to the applied ontologies.

In [ME09], Mabroukeh et al. utilize in a first step semantic information to assist the sequential web usage pattern mining process. Then, they use semantic relatedness for determining the transition probabilities of a Markov Model that predicts the next page visited by the user. Zhao et al. propose a time-dependent semantic similarity measure of web search queries by considering the temporal factor when mining click-through data in order to express the dynamic nature of queries over time [ZHL+06]. In addition, they place their trust in a probabilistic similarity measure that reflects the web queries' frequency distribution.

All existing approaches constrain themselves to static location categories and types without considering the dynamic and purpose-of-visit-dependent varying role of locations. This *lack of flexibility* is carried over to their methods of representing associations between locations and users through static and unalterable axioms or rules. Solely Long et al.'s work investigates a dynamic approach but it is based alone on popularity and not on the semantics behind the locations. In this chapter, we introduce a location prediction model that takes the aforementioned dynamics explicitly into account. We show that by doing so, we are able to improve the model's precision and accuracy.

4.3 Purpose-of-Visit-Driven Semantic Similarity (PoVDSSA) on Semantic Trajectories

The framework proposed in this chapter is illustrated in Fig. 4.1. Our approach merges semantics with machine learning and consists of two main parts. One part is responsible for the semantic processing of the available information (top branch), while the other one takes charge of the actual location prediction (bottom branch). The sensed data like location and time (e.g., in the form of GPS readings) follow both paths at the same time. On the one hand, these are being semantically annotated, enriched with further semantic information, such as the purpose of visit, and stored in the Semantic Annotated Database (SADB). The annotation of locations takes place semi-supervised partly by

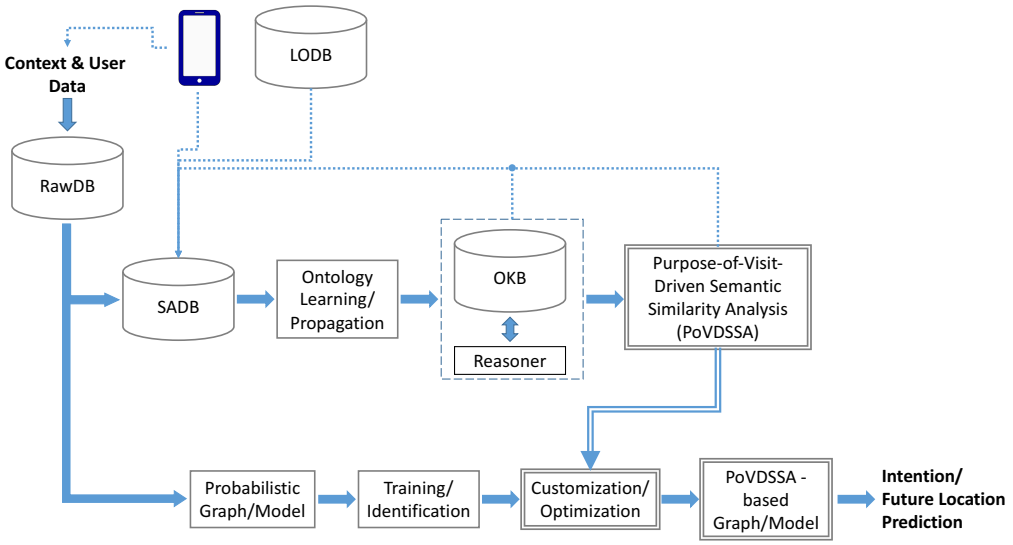

Figure 4.1: Purpose-of-Visit-Driven Semantic Similarity Analysis based location prediction (PoVDSSA) framework, whereby rawDB refers to the raw data Database, SADB refers to the Semantically Annotated Database, LODB to a Linked Open Database, OKB to the Ontology-based Knowledge Base and PoVDSSA to the Purpose-of-Visit-Driven Semantic Similarity Analysis component.

the user through an Android app running on the smartphone (see Section 3.5.1) and partly by utilizing a (geographic) Linked Open Database (LODB). In this way, both public as well as private locations, like the users' home or work, can be correctly identified among the recorded data. The same tracking Android app is also responsible for collecting additional semantic information like the purpose of visit mentioned above. The resulting data are then used to propagate our Ontology-based Knowledge Base (OKB) described in Section 4.3.1 and to build our so called *Purpose-of-Visit-Dependent Frame* objects (*PoVDF*). A reasoner assists additionally the creation and the extension of our ontology in case of lacking data by making use of the existing assertions (e.g., by means of subsumption reasoning based on Description Logic (DL)). *PoVDSSA* is the core component of our approach and refers to the *Purpose-of-Visit-Driven Semantic Similarity Analysis* that takes place in order to cluster locations dynamically depending on the respective purpose of visit. Thus, it is responsible for providing our approach with a dynamic context-aware view

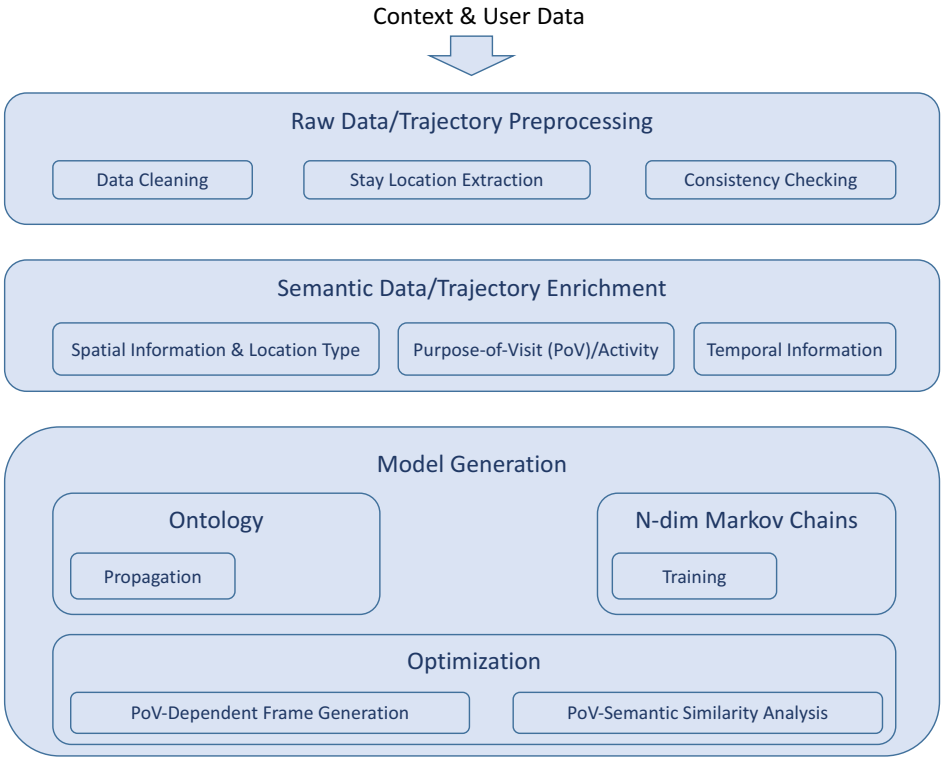

Figure 4.2: Layer diagram of the Purpose-of-Visit-Driven Semantic Similarity Analysis based location prediction framework (PoVDSSA).

of locations at anytime. A detailed description of PoVDSSA can be found in Section 4.3.2.

The bottom branch of our framework contains the actual location prediction model, which is in our case the multi-dimensional Markov Chain ensemble described in Chapter 3. In other words, the here presented method represents an extension of the former model. Other machine learning techniques like Artificial Neural Networks (ANNs) can be applied here as well though, as we shall see later, in Chapter 10. The training and prediction process looks as follows. First, the prediction model is trained with the available raw data clustered into significant locations such as in [AS02]. Next, the trained model expects an optimization through the customization of its (previously learned) parameters based on the semantic similarity analysis of locations described in detail in Section 4.3.3. Finally, the optimized PoVDSSA-based prediction model is able to provide an estimation about the future semantic locations that the user(s) intent to visit next. Fig. 4.2 summarizes the principal functions of

our approach into a set of three layers: the preprocessing layer, the semantic trajectory enrichment layer and the modelling and optimization layer.

4.3.1 Semantic Enriched Location Data and Ontology-based Knowledge Base

In order to comprise all the different facets of the associated to the locations semantic knowledge, we chose to use an ontology-based knowledge base. In this way, we are able to represent hierarchical and taxonomical relations of locations, purposes of visit, high level activities and elementary actions, as well as to define our own properties and relations to one another. At the same time, we use the same ontology to describe temporal information as well. We implemented our ontology in OWL using the Protege tool[4].

Our ontology consists of four major entities (classes):

1. Locations
2. Purpose of Visit
3. Actions
4. Temporal & Event

The *Locations* entity captures a taxonomy of various location types, like *night club, bar, restaurant, dinner, fast food restaurant,* etc. In order to build the particular taxonomy, we oriented ourselves on the Foursquare venue categorization[5] as in Chapter 3.

The *Purpose of Visit* entity covers the annotated by the users reasons for visiting each location. In tangible terms, it refers to the complex, high-level activities that may take place in a location, such as *working*, *celebrating* and *relaxing*.

The *Actions* entity includes the elementary actions of which the high-level activities are composed. For instance, the high-level activity *celebrate a birthday* is related to the low-level actions *meet friends, meet family, eat, drink, etc.*

[4]http://protege.stanford.edu

[5]https://developer.foursquare.com/categorytree

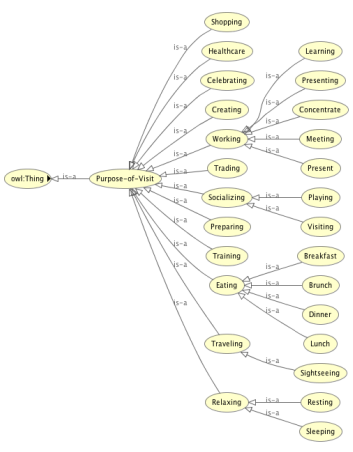

Figure 4.3: Purpose of Visit

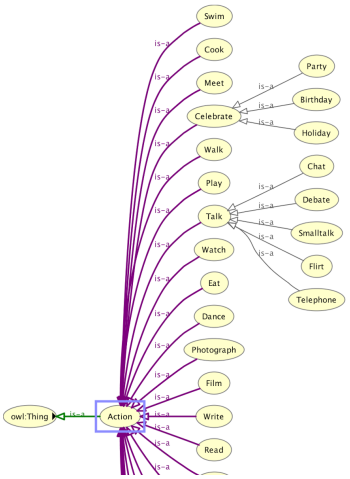

Figure 4.4: Actions

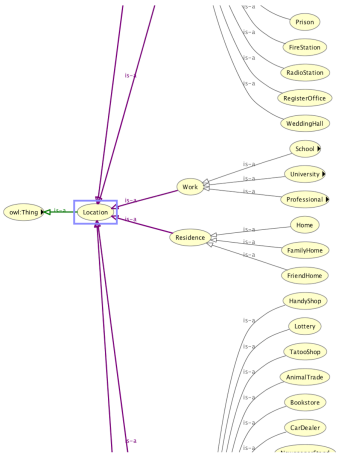

Figure 4.5: Locations

Finally, the *Temporal & Event* entity describes time from a human point of view, considering a human-like time granularity (e.g., *morning*: 6am-11am). For describing duration (time intervals with start and end) and time in general we made use of the standard OWL Time Ontology[6]. This is necessary for defining timeslots and blocks, which in turn provide the aforementioned temporal granularity of our semantic trajectories. Beside time in general, particular attention is paid to the temporal entity event, which refers to special events like *anniversaries, birthdays, public holidays, etc.* that are strongly related to irregular behaviour. Fig. 4.3, 4.4 and 4.5 illustrate parts of the aforementioned classes in our ontology.

As already mentioned in the introductory Section 4.1, the location prediction framework introduced in this chapter aims at capturing the dynamic role of locations based on the respective current context. For this purpose, we need to go beyond using just the location type and link the location entity with the rest of the concepts of our ontology: the time, the high-level purpose of visiting the particular location and the corresponding activities. Moreover, we consider this context information as additional attributes of the respective location. However, OWL supports solely *binary relations*, that is, relations between two individuals. Relations that handle more than two individuals are referred to as *n-ary relations*, with $n > 2$. In order to link more than two indi-

[6]https://www.w3.org/TR/owl-time/#toc

viduals, there exists a number of standardized workaround solutions, so called *ontology design patterns (ODP)* [GP09]. One way to overcome the binary relation problem is to introduce a new class for representing the desired relation [W3C18]. In this work, we define a new entity, which we name *Purpose-of-Visit-Dependent Frame (PoVDF)* (see Fig. 4.6). The underlying idea is to use

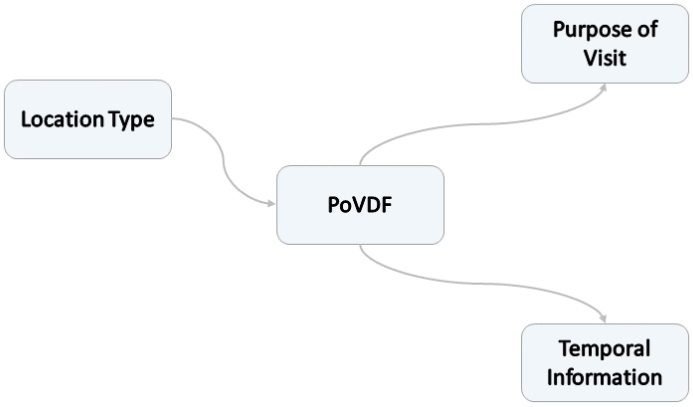

Figure 4.6: Introducing a new entity PoVDF for representing an n-ary relation.

this class for clustering locations that belong together due to sharing common attributes, such as the same reason and time for visiting them. The name refers in part to Minsky's cognitive *Frames* in the 70s', who used this term to encapsulate situation- and experience-based knowledge [Min74, Min75].

4.3.2 Purpose-of-Visit-Driven Semantic Similarity Analysis (PoVDSSA)

Humans tend to employ cognitive frames, that is, certain mental constructs, in order to interpret things, entities and their experiences about them [HNFB10]. Taxonomization and building groups and relations between them and the included entities help clarifying concepts and are therefore of high importance. In order to build groups, to *cluster* things, we draw on the fundamental notion of *Similarity*. Similarity is defined as follows [oxf17b]:

> *Having a resemblance in appearance, character (characteristics), or quantity, without being identical*

According to this definition, two objects are similar, when they share the same characteristics. While this definition refers rather to the similarity between two physical objects, it can analogously be extended to a more general one that expresses a characteristic-based similarity between two objects in a knowledge graph or an ontology. This kind of similarity can then consequently be referred to as *semantic similarity*. Likavec et al. inspired from Tversky's work [Tve77], defined and investigated such a property-based semantic similarity among ontological objects in [LOC15]. In our work, we adopt Likavec's method and define a similar equation to cluster semantically the visited locations of the users based on the purpose of visit and the corresponding time. So, we treat both the time, and the purpose of being at a location as characteristic features of that particular location, which in turn reflects the PoVDF concept mentioned in section 4.3.1. Equation 4.1 illustrates the property-based semantic similarity adapted to our use case:

$$Sim(l_1, l_2) = \frac{CP(l_1, l_2)}{DP(l_1) + DP(l_2) + CP(l_1, l_2)}, \tag{4.1}$$

whereby l_1 and $l_2 \in L$ represent two different semantic locations, CP refers to the common properties (e.g., with respect to the purpose of visit and the time and day) of the particular locations and DP gives the distinctive purposes that are associated only to the one location and do not appear in conjunction with the other. However, formula 4.1 captures solely a single moment. That is, it actually describes the similarity between two particular *stays* s_1 and s_2 at the locations l_1 and l_2 only for a certain moment and not the overall location similarity. A *stay* refers here to a single visit of a certain location at a certain time (and day) for a certain reason. In order to calculate the overall semantic similarity between locations, we compute the average pairwise similarity of all existing stays at l_1 and l_2 as shown in formula 4.2. This reflects our definition of a *Purpose-of-Visit-Driven Semantic Similarity (PoVDSS)*.

$$PoVDSS(l_1, l_2)_{pair\varnothing} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} Sim(s_i, s_j)}{M * N}, \tag{4.2}$$

whereby M and N provide the number of stays at the location l_1 and l_2 respectively. Our PoV-driven semantic similarity takes all different location and purpose-of-visit hierarchy levels that are modelled in our ontology into consid-

eration and calculates a min-max normalized aggregated value between 0 and 1.

By clustering locations in this way we are able to go beyond a simple type-specific categorization of locations and cluster even locations of different type together. Let us consider an example to clarify this statement. The locations “park", “gym" and “restaurant" would probably land in three different categories if we tried to cluster them by taking only the location type into account. In contrast, our approach provides a more dynamic clustering by considering additionally the purpose of visit. In this case, “park" and “gym" would be considered temporarily similar if the person visits the park for jogging due to the fact that jogging is a fitness activity and thus common to the gym's overall purpose of visit. Analogously, “park" would be found similar to the “restaurant" if the person has a picnic at that park.

4.3.3 PoVDSSA-based Model Optimization Process

This section describes the optimization process for the multi-dimensional Markov Chain model of Chapter 3, which we use here as our location predictor. In this case, the optimization process adapts and updates the location transition probability matrix based on the Purpose-of-Visit-Dependent Semantic Similarity Analysis (PoVDSSA) illustrated in the former section as described below.

The PoVD semantic similarity analysis takes place each time when a prediction is to be made. It investigates how similar the current semantic location l_{cur} is to each of the other locations found in our propagated knowledge base. At the end, it provides us with a number of locations and their corresponding similarity scores. Next, we rank the results based on their score. The top location represents the current location itself because it corresponds to an absolute similarity of 1.0 and thus we disregard it and chose the second from the top location as the most similar location l_{maxSim}. At each prediction time step, we weight the Markov matrix' transition probability row of the location with the highest similarity l_{maxSim} by multiplying each element with the maximum similarity score ($SimScore_{max}$). Finally, we use the resulted row to update the transition probability row that corresponds to the current location by applying

the following formula:

$$\begin{aligned} TP(l_{cur})_{i,new} = & TP(l_{maxSim}) \times SimScore \\ & + offset \times TP(l_{cur})_{i,old} \end{aligned} \tag{4.3}$$

The transition probability is being updated either if the respective similarity score exceeds a certain threshold min_{Sim} or in the case of a missing element in the transition probability matrix due to the sparse dataset filling in this way the respective gaps. The updating algorithm is described in detail below:

Algorithm 1: Markov transition probability updating process.

Data: Current location $l_{cur} \in L$, Current Context C (Purpose of Visit, Time, ..),
Multi-Markov-Chain model M, Set of all locations $L = [l_1, \dots l_n]$
Result: Updated transition probabilities for location l_{cur}

$min_{Sim} \leftarrow 0.1, \dots, 0.9$;
$probabilities_{[l_{cur} \rightarrow *]} \leftarrow M.getProbabilities(l_{cur}, C)$;
$probabilities \leftarrow probabilities_{[l_{cur} \rightarrow *]}$;
$sim_{l_{cur},*}[sim_{l_{cur},k_1}, \dots, sim_{l_{cur},k_n}] \leftarrow getSimilarities(l_{cur}, C)$;
$sim_{l_{cur},*}.sortReverse()$;
while $sim_{l_{cur},*}.hasNext()$ **do**
    // location $k \in L$, shows highest similarity score to l_{cur}
    $sim_{l_{cur},k} \leftarrow sim_{l_{cur},*}.next()$;
    if $sim_{l_{cur},k} \geq min_{Sim}$ **then**
        $probabilities_{[k \rightarrow *]} \leftarrow M.getProbabilities(k, C)$;
        $probabilities \leftarrow updateProbabilities(probabilities_{[l_{cur} \rightarrow *]}, probabilities_{[k \rightarrow *]}, sim_{l_{cur},k})$;
        break;
    end
end
return *probabilities*;

Thus, the updated transition probabilities for the current location depend on the one hand on the transition probabilities of the most similar location and the corresponding similarity score. On the other hand, they still depend on the old values (provided the fact that these exist), much less now though, due to the *offset* factor. This provides us with a smooth and adaptable reward-penalize function. The *offset* is a hyperparameter whose value is determined through a grid search.

4.4 Evaluation

In order to evaluate our approach, we used the preprocessed data collected during the user study described in Section 3.5.1. We used k-fold cross valida-

tion for our evaluation and tested the following k-parameter values: [5, 10, 15, 20]. We compared our approach with a multi-dimensional semantic 1. order Markov Chain Model as well as with the semantic trajectory-based approach of Ying et al. [YLWT11b] with a minimum support of 0.01. Fig. 4.7 illustrates the performance of our PoVDSSA-based approach in comparison to Ying et al.'s framework over various k values with respect to macro accuracy, macro precision, macro recall and macro F-score (see Section 2.3). As can be seen

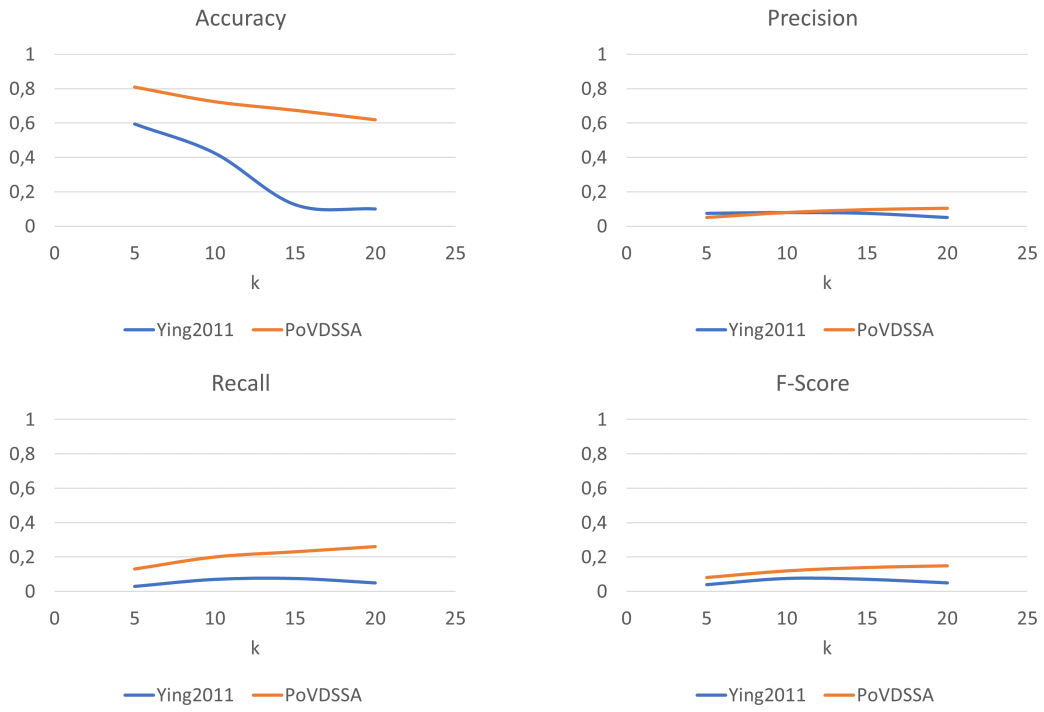

Figure 4.7: Comparison of our approach (PoVSSA) to Ying et al.'s approach with regard to accuracy, precision, recall and F-score.

from the figure, our approach performs better than Ying at al.'s framework almost every time with respect to all of our four metrics. It leads clearly to both a higher accuracy performance and a higher recall value. This means that our system is not only more accurate, but its estimations are more consistent, especially when the k value is high. That is, it seems that it is easier for our model to find the relevant locations during the estimations. This could be mainly attributed to the fact that our approach is able to replace and fill out missing current location transitions through existing transitions coming from the corresponding similar locations. In this way, it can partly handle the

data sparsity problem mentioned in Chapter 3. The accuracy in both methods is decreasing if we use a finer data breakdown (higher k value). This can be explained by the fact that our data consist of sequences. The finer we split the sequences, the more incoherent (in terms of chronological order) become the resulting training data. Thus, it becomes harder for the models to train. Furthermore, the upper right part of the figure shows that our approach is only slightly better than our baseline with regard to precision. This reflects the fact that both share the same ratio of predicting the respective future locations correct and incorrect (True Positives and False Positives respectively). In other words, the estimations of both models scatter at a comparable level. Apart from the accuracy, the models show overall low performance scores. These can be mainly attributed to the size and the quality of our training and evaluation dataset.

Fig. 4.8 displays the performance of different variants of our approach compared to the Markov model described in Chapter 3 in relation to the similarity threshold value, which determines whether a transition probability update should occur or not (see Algorithm 1). In particular, we use the Markov

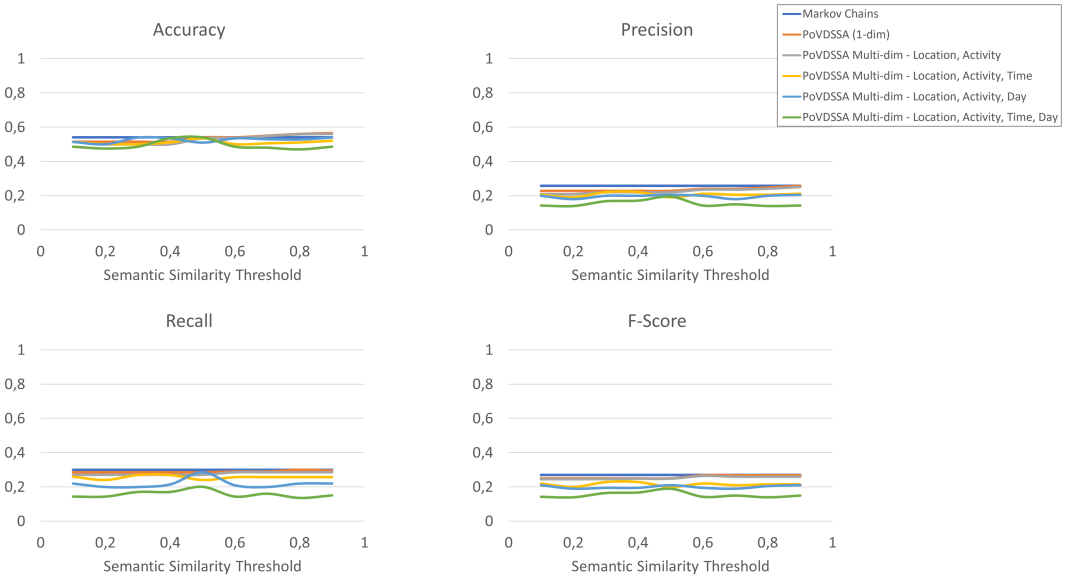

Figure 4.8: Comparison of a number of different variants of our approach (PoVSSA) against the Markov model described in Chapter 3.

model as baseline and we explore the impact of the diverse dimensions on the performance of our PoVDSSA-based approach.

- *PoVDSSA 1-dim*: a model that considers solely the location
- *PoVDSSA Mutli-dim - Activity*: a model that considers location and activity
- *PoVDSSA Multi-dim - Activity, Time*: a model that considers location, activity and time of day
- *PoVDSSA Multi-dim - Activity, Day*: a model that considers location, activity and day of week
- *PoVDSSA Multi-dim - Activity, Time, Day*: a model that considers location, activity, time and day

We can see that most variants of our model achieve a similar performance as our baseline, without being able to outperform it though. Solely in terms of accuracy perform all models equally with the 1-dimensional PoVDSSA-based model achieving actually a slightly higher score. Interestingly, with respect to recall, precision and consequently F-Score, increasing the number of dimensions seems to be having a negative effect. The higher the feature dimension number, the worst the performance. Especially the additional temporal dimensions (time of day and day of week) seem to lead to lower scores. The spatial (PoVDSSA 1-dim) and the Location-Activity (PoVDSSA Multi-dim - Activity) model yield the overall best results compared to the rest of the model variants. At the same time, the model that takes all the available context information (location, activitiy, time of day and day of week) into account shows the poorest performance.

What also stands out from Fig. 4.8 is that the value of the semantic similarity threshold seems also to be playing a significant role, particularly for the time-dependent models, the ones that take temporal information into account. In the case of the latter, a higher threshold leads to better results. In tangible terms, this means that models that lay more weight in the similarity between locations achieve better results, when these models don't take temporal information into account. However, a too high threshold limits the number of the existing potential similar transitions, which in turn downgrades the performance.

Using time and day as class attributes for comparing 2 location types appears to be not useful. On the contrary, it seems to be having a negative influence. In particular, the information *day of week* appears to be more critical than the information *time of day*. This can be attributed to the size of our dataset. It contains movement patterns of 5 weeks. This means that there exist only 5 daily trajectories for each single day of week. This number is to low for our model to be able to make serious assumptions based on the day of week. In comparison, each hourly slot exists 35 times in our dataset.

In general, at first glance, the updating of the location transition probability matrix based on the semantic similarity of the respective locations doesn't seem to be leading to any kind of improvement. Moreover, it seems that our models face largely the same difficulties as our non-semantic-enhanced Markov model, namely the sparsity of our collected dataset. After further analysis of our results, we found that the uneven distribution of the labelled data among users, the overall missing or false labelling of the purpose of visit and the fact that our 5-week long data cover very few location-time-day-activity combinations have led to a situation in which our PoVDSSA-based updating approach was triggered only very rarely. Too rarely to cause some significant positive impact.

However, our analysis also showed that the larger and the more consistently annotated the dataset was, the better our approach performs. Fig. 4.9 refers to the results of the user with the most annotated locations and activities. In this case, almost all of our PoVDSSA-based variants perform better than our baseline. Once again, the *PoVDSSA 1-dim* and the *PoVDSSA Multi-dim - Activity* outperform the rest, achieving an almost twice as high recall and F-Score value than the multi-dimensional Markov approach of Chapter 3. Furthermore, as before, the model that considers the most context information (*PoVDSSA Multi-dim - Activity, Time, Day*) shows the worst performance and lies with respect to precision, recall and F-Score below the Markov baseline. But surprisingly, this time, in terms of accuracy, it provides with almost 70% the highest results, a fact that at least indicates the importance and the added value in taking additional context information into account. It is also apparent from this figure that in this case, the semantic similarity threshold plays a less significant role.

All in all, we could see that our Purpose-of-Visit-Dependent approach is

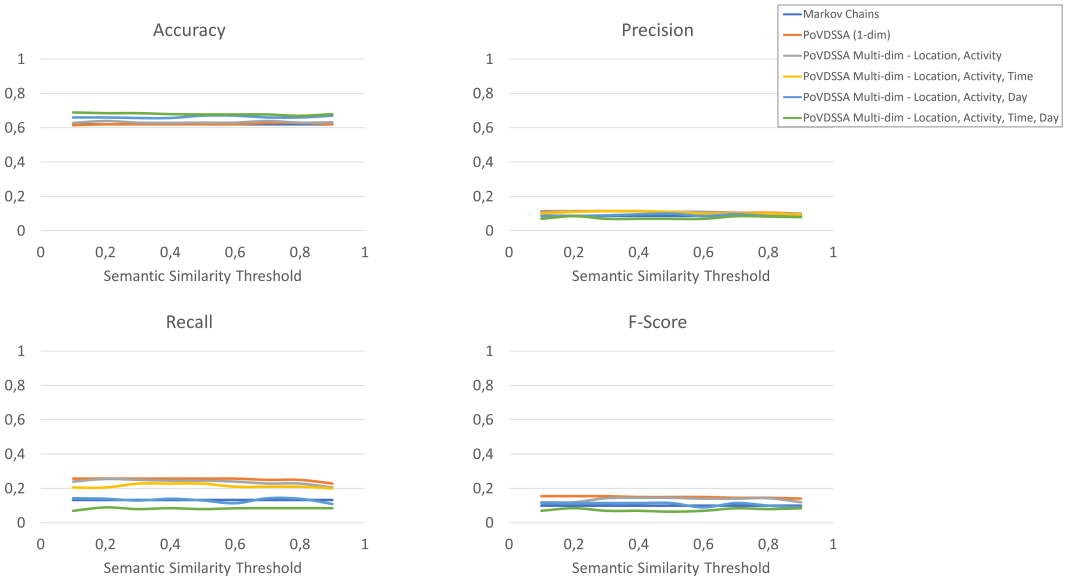

Figure 4.9: Comparison of various different configurations of our approach (PoVSSA) to the Markov model described in Chapter 3 for the user with the most annotated locations and activities.

capable of leading to overall better results than the Markov model in most of the cases with a consistently labelled dataset. However, it shows a similar sensitivity towards training data sparsity and quality as the multi-dimensional Markov model of Chapter 3. We could also see that temporal information actually impairs our system instead of improving it. This can be explained by the fact that, when it comes to human movement, it is the sequence itself, which plays the most significant role and not the absolute information of time or day. This could be further explained by our sparse dataset. We would expect better results with more data that cover much more different situations at various points of time in the users' movement behaviour.

4.5 Conclusion

In this chapter, we propose and investigate *Purpose-of-Visit-Driven Semantic Similarity (PoVDSSA)* as part of a semantic location prediction method in combination with a multi-dimensional Markov Chain model. We were driven by our hypothesis that dynamic situation-dependent location clustering can enhance the overall performance. We implemented our PoVDSSA-based al-

gorithm and tested it on the real-life dataset described in Chapter 3. We evaluated our framework against two other approaches, namely to Ying et al.'s semantic trajectory based approach and to a conventional Markov Chain model. We show that our approach outperforms both Ying et al.'s and the Markov-based approach. However, some drawbacks, due mostly to the lack of a bigger dataset, could also be identified.

In Chapter 9, we apply an extended and more personalized version of the presented context-aware semantic similarity analysis approach that takes personality and emotions additionally into account in order to optimize the predictive performance of artificial neural networks using the example of location prediction.

CHAPTER 5

COMBINING MARKOV CHAINS WITH MATRIX FACTORIZATION FOR OVERCOMING DATA SPARSITY

Abstract

As we already saw in Chapters 3 and 4, data sparsity can be a significant handicap for data-driven semantic location prediction approaches. In this chapter, we introduce a semantic location prediction approach that provides user-specific predictions based on their past trajectories, while attempting to overcome data sparsity at the same time. For this purpose, we adopt an item recommendation method called Factorized Personalized Markov Chains (FPMC). FPMC relies on a combination of Matrix Factorization and Markov Chains. We evaluate our algorithm using the 9-month long Reality Mining dataset of MIT and compare it against the user-independent standard Matrix Factorization (MF) and the Factorized Markov Chains (FMC). It can be shown that our FPMC-based approach is able to surpass clearly the performance of both aforementioned methods.

The content of this chapter is based on a publication at WiMob 2017 [KLB17].

5.1 Introduction

There exist two main common issues that researchers in the field of mobility analysis have to deal with. The first refers to deriving knowledge out of people's sequential geolocation data, as well as modelling and utilizing it appropriately. In other words, this means to go beyond numbers (GPS coordinates or Cell Tower IDs) and really understand (conceptually) where a person is located at or is heading to, which can be described as *content-specific mobility modelling and analysis* and modelled by means of semantic trajectories. The second issue refers to the fact that researchers usually have sparse, incomplete datasets for building their models, a fact that is partly attributable to data privacy concerns and regulations and partly to the fact that no system exists that is able to capture *every* single relevant information. For instance, when a Markov model is used for location prediction, it is characterized by the corresponding location transition probability matrix as described in Chapter 3. Each element of this matrix contains the probability of either moving from a certain location to another location or not moving at all. Thus, in order for a Markov model based approach to work properly, the transition probability matrix has to be fully occupied. Only then is the model able to provide accurate predictions at any case. However, the available datasets that are used for training the Markov models don't usually cover all possible cases. This results in sparse, semi-occupied transition probability matrices, which in turn makes predictions either not executable or erroneous.

In this chapter, we propose a Factorized Personalized Markov Chain (FPMC) based location prediction algorithm, which models semantic trajectories using multi-dimensional Markov Chains. The multi-dimensionality isn't referring here to the same multiple dimensions described in Chapters 3 and 4 though. Instead of using further dimensions to catch additional context information, the multi-dimensionality is used here to provide personalization in the prediction outcomes, while the Markov model follows up with the dynamics of the people's movement behaviour as before. In addition, we use Matrix Factorization (MF) for dealing with the sparse Markov transition probability matrices, as well as the aforementioned general sparse data issue. At last, we test and evaluate our approach using the MIT Reality Mining dataset [ESP06].

The chapter is structured as follows. Section 5.2 provides some insight in the most related work so far. Next, section 5.3 gives a brief glimpse in the theory of Matrix Factorization, followed by section 5.4, where we describe our location prediction approach. In the two last sections 5.5 and 5.6 we show the results of our evaluation and draw our final conclusions respectively.

5.2 Related Work

In this section, we will take first a short look at a group of probabilistic location prediction papers that propose ways to deal with the data sparsity issue. Although Matrix Factorization is often used against data sparsity in various fields, the combination of Matrix Factorization with semantic trajectories and semantic location prediction seems to be novel, as no papers on this specific topic could be found. Therefore, in an attempt to cover Matrix Factorization, the second part of this section will reference mainly papers coming from a related field, in which they are commonly used: Point of Interest (POI) recommendation and item recommendation.

Gao et al. extend the spatial model by adding temporal context to their probabilistic model [GTL12]. For this purpose, they calculate the likelihood of the corresponding temporal conditions when visiting a certain location. The observed temporal information refers in this case to the time of day (in hourly slots) and the day of week. However, available training and validation datasets are usually sparse and do not cover all necessary combinations (*location*, *hour of day*, *day of week*). That is, not every location has been visited at any times on any days. This leads to a sparse location transition matrix with missing transition probability values and thus to either erroneous or not executable predictions. To solve this problem, Gao et al. assume and utilize 2 Gaussian distributions representing the odds that a certain location will be visited at a certain time of day on a certain day respectively. Their approach, evaluated on the Nokia mobile data challenge dataset [LGPA+12], outperforms the baseline models, including the Markov predictor with "Fallback" presented in [SKJH04], which also aims at overcoming the data sparsity problem as well by replacing empty transition matrix elements with values from a second probabilistic model.

Koren et al. give a thorough overview of several different techniques of Matrix Factorization in [KBV09]. One paper that stands particular out of the rest, is the work of Bell et al. [BKV08], which refers to a Matrix Factorization based algorithm that won the Netflix Price[1], a competition for proposing a movie recommendation engine for the users of the company's streaming service. Their approach was capable of providing accurate recommendations by identifying users with similar taste and reducing the dimensionality through Matrix Factorization. Matrix Factorization has recently been also used in location recommendation, which is closely related to location prediction. Using Check-In data from LBSNs, Cheng et al. suggests in [CYLK13] a method that combines Markov Chains with Matrix Factorization. It allows personalized recommendations for several users while taking the current region of the user into account. Duong-Trung et al. propose a Geo Matrix Factorization model for predicting the location of Twitter users at the time they post a Tweet.

The approach presented in this chapter, focuses, in contrast to the aforementioned work, on semantic trajectories and builds upon the semantic movement history of users to predict their next semantic location.

5.3 Matrix Factorization

In linear algebra, *Matrix factorization (MF)* refers in general to the decomposition of a matrix into a product of matrices:

$$A = B \cdot C \cdot ... \tag{5.1}$$

Due to the nature of the problem, numerical approximation methods are widely used for determining the matrices on the right. In the field of *latent factor modelling*, a great variety of realizations base on Matrix Factorization, especially since its surprisingly good performance at the Netflix competition in 2009 mentioned previously [BKV08]. Latent factor models find use in personalized search machines and recommendation systems, like in item recommendation applied in the field of e-commerce. They aim at characterizing both users and items with a small set of the most significant factors. The respective factors are usually inferred either by trying and testing, e.g. using cross-validation,

[1] http://www.netflixprize.com

or by applying pattern mining techniques. So, in connection with latent factor modelling, Matrix Factorization can be perceived as one form of principal component analysis (PCA) close to singular value decomposition (SVD). In contrast to SVD, Matrix Factorization can be defined for incomplete matrices as well. A basic Matrix Factorization model as described in [KBV09] defines the link between users and items in a *factor space* of reduced dimensionality f as inner product. Each item i in this space is characterized by a vector $q_i \in \mathbf{R}^f$ whose elements reflect to what extent the particular item is associated to each factor. Analogously, a vector $p_u \in \mathbf{R}^f$ describes the relation of user u to the same factors. Their inner product returns an estimation of the degree of relation between item and user, such as a rating of item i by user u:

$$\widehat{rating_{u-i}} = {q_i}^T \cdot p_u \tag{5.2}$$

A significant advantage of Matrix Factorization compared to other techniques, is its ability to overcome missing information (see Tables 5.1 and 5.2); an ability that played a major role in choosing it for our use case as we will see later. By trying to minimize the squared error of equation (5.3) between known ratings (of items by users) we end up learning the complete vectors q_i and p_u, including estimations about possible missing elements.

$$e = (rating_{u-i} - \widehat{rating_{u-i}})^2 = (rating_{u-i} - {q_i}^T \cdot p_u)^2 \tag{5.3}$$

Two common numerical methods for finding the minimum squared error are *stochastic gradient descent (SGD)* [BC11] and *Least Squares (LS)*.

5.4 FPMC-based Semantic Location Prediction

Rendle et al. introduced in [RFST10] the *Factorized Personalized Markov Chains (FPMC)*, a next basket-of-items recommendation system that combines Markov Chains with Matrix Factorization. Their algorithm is able to provide personalized item recommendations while taking the purchase order into account at the same time. Moreover, by applying Matrix Factorization, their algorithm is able to deal with missing user-item interactions, e.g., item ratings. Their work served as basis for our approach presented in this chapter.

	I1	I2	I3	I4
U1	5,00	3,00	-	1,00
U2	4,00	-	-	1,00
U3	1,00	1,00	-	5,00
U4	1,00	-	-	4,00
U5	-	1,00	5,00	4,00

Table 5.1: Example of a User-Item rating matrix. This table describes the ratings that 5 users gave to 4 different items (e.g., movies). The missing values in the matrix are caused by the fact that not every user has rated every item. (Source: [Alb10])

	I1	I2	I3	I4
U1	4,97	2,98	**2,18**	0,98
U2	3,97	**2,40**	**1,97**	0,99
U3	1,02	0,93	**5,32**	4,93
U4	1,00	**0,85**	**4,59**	3,93
U5	**1,36**	1,07	4,89	4,12

Table 5.2: Matrix approximation by applying Matrix Factorization on the User-Item rating matrix of Table 5.1. (Source: [Alb10])

Our approach relies on the hypothesis that FPMC is a fitting algorithm for providing a personalized semantic location prediction due to the similar nature of the problem and the same objectives, which can be summarized as follows:

- Personalization
- Consideration of purchase order $\approx$ Consideration of location sequence (trajectories)
- Handling of missing elements $\approx$ Handling sparse data (e.g., new, unvisited locations)

In the following, we will be using an analogous formulation to the one found in [RFST10]. H_u represents the sorted set of locations visited by user u. The locations are sorted by date and time. $H_{u,t}$ and $H_{u,t+1}$ represent locations

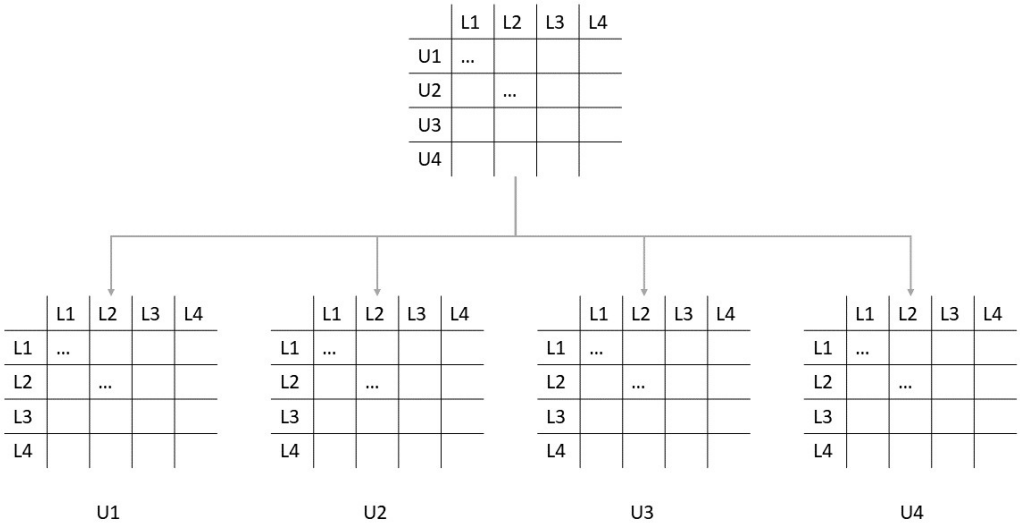

Figure 5.1: User-Location matrix to user-specific Location-Location matrices.

visited by user u at the time t and $t+1$ respectively. Not the exact time, but the relative chronology of the visits is hereby modelled.

Analog to [RFGST09], we transform the classical User-Location problem to a user-specific Location-Location problem as shown in Fig. 5.1. Each user receives an own matrix that associates locations with each other. In particular, the location matrix contains the information whether a specific user prefers (has visited more often) a certain location over another location.

At the end, our goal is to determine a personalized ranking over all possible future location pairs at time t.

$$<_{u,t} \subset \ I \times I = I^2, \tag{5.4}$$

whereby I refers to the set of all locations. The location on the top of the ranking represents lastly our final estimation $\widehat{H_{u,t+1}}$.

Our location prediction model relies on Markov Chains. Accordingly, the probability that the user visits location i after visiting locations $j_1, ..., j_n$ is:

$$P(i = H_{u,t+1} | j_1 = H_{u,t}, ..., j_n = H_{u,t+1-n}) \tag{5.5}$$

and for Markov Chains 1. Order ($n = 1$):

$$P(i = H_{u,t+1} | j = H_{u,t}) = a_{j,i}, \tag{5.6}$$

whereby $a_{j,i}$ is an element of the corresponding transition matrix $A \in [0,1]^{I \times I}$.

For displaying higher order Markov Chains by a transition matrix without adding further dimension to it and make it easier to read, we leverage the same method as in [RFST10] and aggregate the n last visited locations $j_1, ..., j_n$ in $l \in L$:

$$l = (j_1, ..., j_n) \in I^n = L \tag{5.7}$$

Thus, for the transition matrix $A \in [0,1]^{L \times I}$ of Markov Chains higher order holds:

$$a_{l,i} = P(i = H_{u,t+1} | l = (H_{u,t}, ..., H_{u,t+1-n})) \tag{5.8}$$

Through Matrix Factorization we obtain two factor matrices $V^{L,I}$ und $V^{I,L}$. The elements of A can then be calculated by:

$$a_{l,i} = \langle v_i^{I,L}, v_l^{L,I} \rangle \tag{5.9}$$

In this case, A refers to a *Factorized Markov Chains (FMC)* based model.

So far, personalization has not been handled yet. The outcome of FMC above would provide user independent next location estimations. But our model should be able to predict the future location $H_{u,t+1}$ of user u, given his location history up to time t: $H_{u,t-n+1}, ...H_{u,t}$. For this purpose, the probability of every location i in I and $l \in L$ is calculated. By adding a user component in our model we extend the transition matrix to a transition tensor $\mathcal{X} \in [0,1]^{|U| \times |L| \times |I|}$:

$$x_{u,l,i} = p(i = H_{u,t+1} | l = (H_{u,t-n+1}, ..., H_{u,t})) \tag{5.10}$$

By applying Canonical Decomposition (CD) (Parallel factor analysis (PARAFAC)), a form of the Tucker Decomposition, we obtain a model of pairwise interaction between the dimensions U, I^n and I (see [RFST10]):

$$x_{u,l,i} = v_u^{U,I} \cdot v_l^{I,U} + v_l^{I,L} \cdot v_i^{L,I} + v_u^{U,L} \cdot v_i^{L,U} \tag{5.11}$$

The result are six factor matrices, one pair for every dimension. These contain all knowledge needed and need less memory and computational effort:

- $V^{U,I} \in \mathbf{R}^{|U| \times k_{U,I}}$ models the features of user u
- $V^{I,U} \in \mathbf{R}^{|I| \times k_{U,I}}$ models the features of next location i

- $V^{I,L} \in \mathbf{R}^{|I| \times k_{I,L}}$ models the features of next location i
- $V^{L,I} \in \mathbf{R}^{|L| \times k_{I,L}}$ models the features of last location l
- $V^{U,L} \in \mathbf{R}^{|U| \times k_{U,L}}$ models again the features of user u
- $V^{L,U} \in \mathbf{R}^{|L| \times k_{U,I}}$ models the features of past location l

The first two matrices model the link between user and next location. They give information about the places a particular user would likely visit. The next two, model the association between next and past location I und L. They give information about the next location i given the past visited location l. This information is independent of the user though and therefore not user-specific. The last two, model the association between user and past location. This information is irrelevant for our predictor and can be neglected without affecting the predictor's outcome, as already proved in [RFST10]. $k_{U,I}, k_{I,L}, k_{U,L}$ represent the latent factor dimensions. These affect the prediction result as we will see later. Furthermore, the higher the dimensions, the more values have to be calculated, which in turn has an impact on the computational time of the learning algorithm of section 5.4.1.

To predict the future location of a user, the following equation has to be calculated for every possible location $i \in I$ first:

$$x_{u,l,i} = v_u^{U,I} \cdot v_l^{I,U} + v_l^{I,L} \cdot v_i^{L,I} \tag{5.12}$$

The resulted locations are subsequently being sorted in descending order. As mentioned above, $V^{U,L}$ und $V^{L,U}$ don't influence the ranking of the locations and can be removed from the equation. The location i with the highest value $x_{u,l,i}$ will be finally selected as the future location estimation $H_{u,t+1}$:

$$\widehat{H_{u,t+1}} = arg_i \; max \; x_{u,l,i} \tag{5.13}$$

5.4.1 Learning Algorithm

There exists a variety of different ways to estimate the factor matrices in Matrix Factorization. Most of them are based on numerical approximation, as already mentioned in 5.3. After testing briefly the common least square method on a sample dataset and comparing it with S-BPR, we decided to

use the latter for learning our factors. The Sequential Bayesian Personalized Ranking (S-BPR) algorithm was introduced in [RFST10] and is an adaption of the Bayesian Personalized Ranking algorithm developed by Rendle et al. in one of his former works [RFGST09]. It is based on the gradient descent approach [BC11] and goes through following steps:

1. Factor matrix initialization of $V^{U,I}, V^{I,U}, V^{I,L}, V^{L,I}$ based on the Gaussian distribution

2. At every iteration, a location is being drawn from the location history set of user u and one that does not belong to it.

3. Gradient descent is being applied on every factor matrix $V^{U,I}, V^{I,U}, V^{I,L}, V^{L,I}$

4. Previous steps are being repeated until some convergence is observed and an optimum is reached

5.5 Evaluation and Discussion

5.5.1 Dataset

We tested and evaluated our semantic location prediction algorithm on the MIT Reality Mining real-world dataset of initially 100 cell phone users collected over 9 months by Eagle and Pentland [ESP06]. After analyzing the data, we could see that not all study participants were equally consistent in recording their data. For this reason, we preprocessed the data and selected a total of 26 users that were at most detailed and consistent with their recordings ($>$ 5000 check-ins). However, it should be noted here that the data still consists of location transitions, where consecutive locations belong to different days, weeks or even months. 90% of the data was used for training and 10% for testing.

Dataset	$\|U\|$	$\|L\|$	Check-Ins	Check-Ins / User
MIT	26	317	14741	566.96

Table 5.3: Reality Mining dataset statistics.

Table 5.3 provides the respective statistics and Fig. 5.2 the overall (high-level) location type distribution. More details about this high-level categorization can be found in Section 6.4.

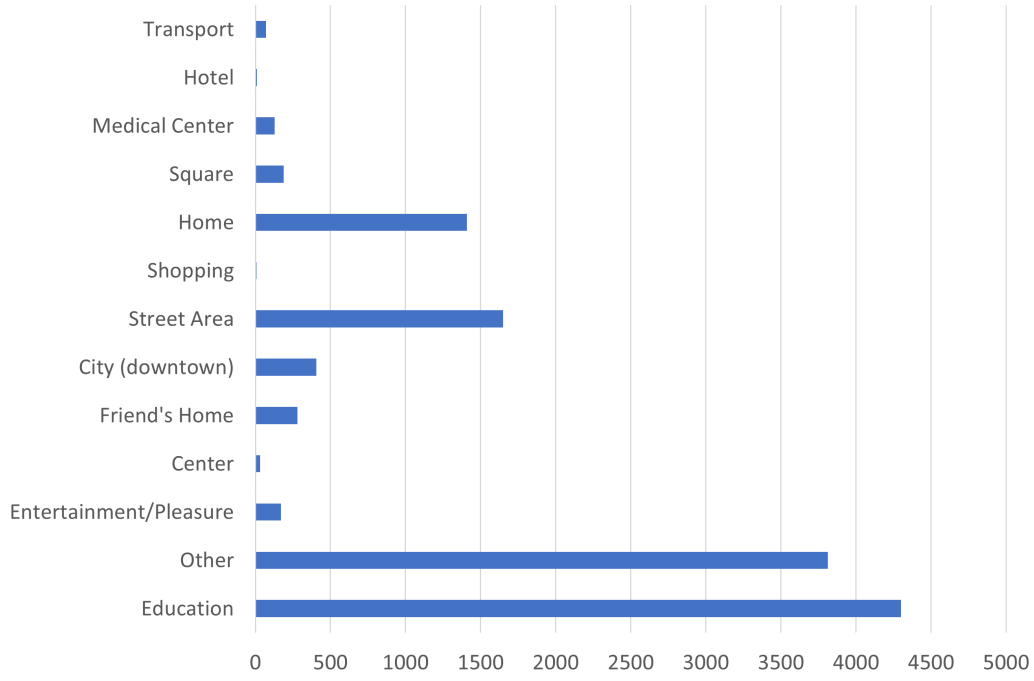

Figure 5.2: Distribution of the high-level semantic locations in the filtered MIT dataset.

5.5.2 Evaluation and discussion

We compared our FPMC-based approach with a standard Matrix Factorization (MF) and a Factorized Markov Chain (FMC) model. We trained and tested each approaches until a certain degree of convergence was observed. For this purpose, the macro accuracy, precision, recall and F1-Score were measured and logged periodically, every 100.000 training steps (= 1 training set). Fig. 5.3 shows that the F1-Score of FPMC reaches the highest value after 10 training sets. After that it falls slightly and stabilizes to a lower value.

During our evaluation we investigated the impact of the latent factor dimensionality on the outcome of our algorithm. We tested following factor dimensions and Markov Chain orders respectively:

$$k_{I,L} = k_{U,I} \in \{8, 16, 32, 64, 128\} \tag{5.14}$$

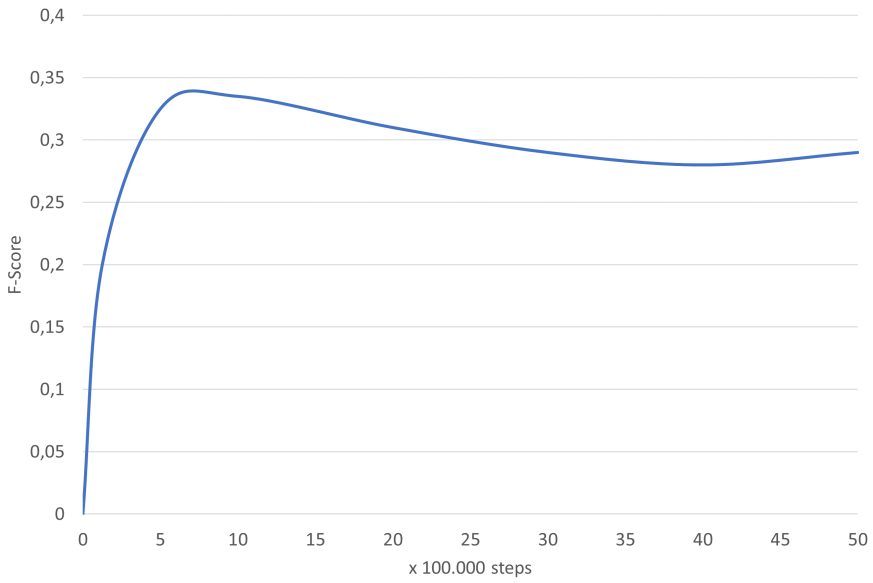

Figure 5.3: Development of the F1-Score during the training phase ($k_{I,L} = k_{U,I} = 128$ and $n = 1$)

$$n = \{1, 2, 3\} \tag{5.15}$$

A correlation between the number of factors and the performance of the predictors could be indeed established as can be seen in Fig. 5.4. The higher the factor dimension gets, the higher the value of the F1-Score. However, the growth levels out from a certain point on. This holds for all three approaches. Furthermore, the same figure clearly demonstrates the general sovereignty of FPMC over MF and FMC. FPMC shows consistently higher values in contrast to the other two methods regardless of the dimensionality factor. Fig. 5.5 summarizes the respective average values reached by all three approaches. The respective values have been obtained with a factorization dimension of 128 and a history of $n = 1$. Higher order Markov Chains led in combination with Matrix Factorization to an at most equal or worst performance. A spatial 1-dim Markov model (MC) of 3. order serves here as our baseline. The MF model takes solely the location preferences of the users into account, while FMC considers additionally the order in which locations were visited. It can be seen that FMC reaches a higher F1-Score than MF. Thus, making additionally use of the order in which users visit places makes a significant

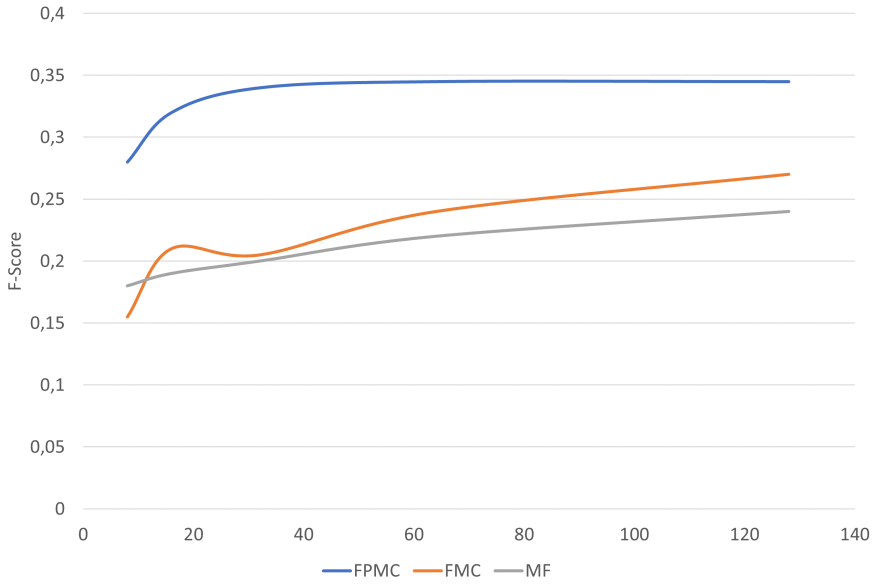

Figure 5.4: F1-Score for a series of values of $k_{U,I}$ and $k_{I,L}$

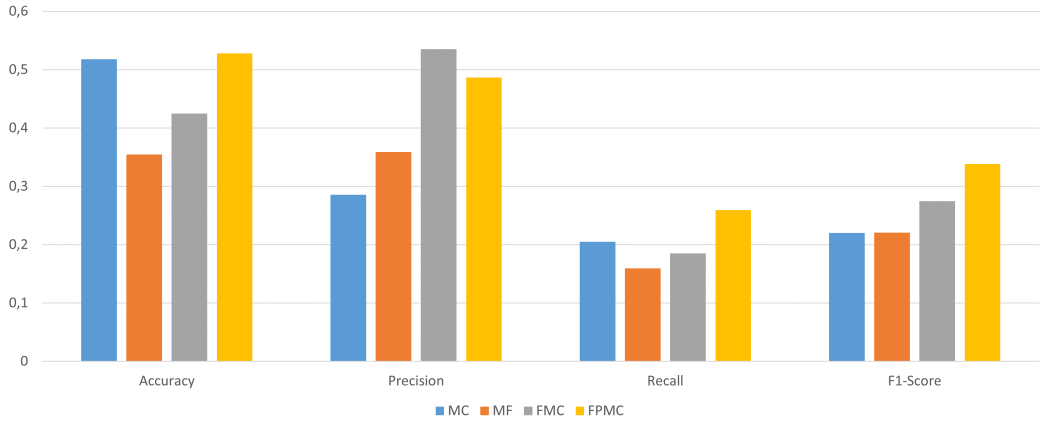

Figure 5.5: Average measured values of MC (3. Order), MF, MFC and FPMC ($k_{I,L} = k_{U,I} = 128$, 1. Order ($n = 1$)).

difference. Our user-tailored FPMC model, which considers both preference as well as the location order performs best, outperforming both the standard Matrix Factorization (MF) and the Factorized Markov Chains (FMC) with respect to prediction accuracy, recall and F1-Score. Solely in terms of precision is the FPMC inferior to the FMC model. Compared to our baseline, the standard Markov model, the FPMC approach results with respect to all 4 evaluation metrics in higher scores, a fact, which confirms our initial hypothesis that Matrix Factorization helps improve the overall performance.

However, the absolute average values seem at first sight not very promising. This can be mainly attributed to the logic behind our algorithm and the nature of the dataset, which despite our preprocessing shows a lack of a sufficient number of consecutive locations. This fact could be confirmed by applying the Sequential Pattern Mining (SPAM) algorithm on the MIT dataset. SPAM is a method for mining sequences in data described by Ayres in [AFGY02]. We obtained low SPAM scores, which means that the data lacks of regular movements among the users. Beyond that, our algorithm is a first approach for testing the power of Matrix Factorization in the field of semantic location prediction together with Markov Chains. Our model represents solely semantic locations and user-specific order of visiting them (or not). More information could lead to better results. General context information, such as time of day, day of week, duration, transportation mode, weather and user-related information like their profile, their profession, their preferences and their appointment calendar could be helpful, as could already been partly confirmed in Chapters 3 and 4. Nonetheless, it could be shown that Matrix Factorization can better handle missing locations and new places compared to MC, FM and FMC.

Finaly, we compared the predictability between a general-based and a user-specific trained algorithm. As one would expect, Fig. 5.6 shows that a personalized approach, that is, a model trained and tested on a single user, yields a better prediction performance. We see that a single-user FPMC is able to reach accuracy and precision scores up to approximately 66%. User-centric applications can take advantage from these results.

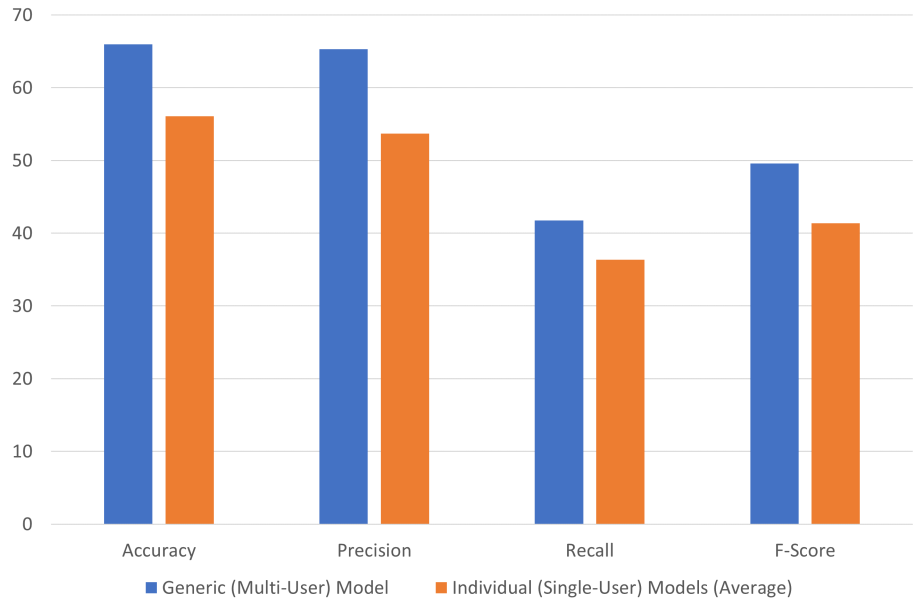

Figure 5.6: Generic vs. individual modelling in % ($k_{I,L} = k_{U,I} = 128$ and $n = 1$). (Highest scores obtained for each case)

5.6 Conclusion

In this chapter, we investigated Matrix Factorization as part of a method for providing predictions about the users' future semantic locations in combination with Markov Chains. We were driven by our hypothesis that personalized sequential shopping item recommendation correlates in many points with the semantic trajectory based location prediction. For this purpose we adopted and adapted the FPMC item recommendation algorithm for our use case. We implemented and factorized a multi-dimensional transition matrix that takes every user and their trajectories separately into account. At the same time, thanks to the way Matrix Factorization works, information from other users can be leveraged to fill the gaps, that is, the missing information (e.g., unvisited locations, etc.), in the prediction model of a certain user.

We implemented our FPMC model and evaluated it against a standard Markov Chains, a standard Matrix Factorization (MF) and a Factorized Markov Chains (FMC) model. We could show that our FPMC-based location prediction algorithm clearly outperforms all three reference systems. We obtained in average almost 30% higher F1-Score and recall values. Additionally, our

approach achieved approximately a 17% and 10% better accuracy as the MF and FMC respectively. However, while the relative performance of our approach is good, this does not hold for its absolute performance. This can be explained for the most part by the nature of our test dataset that lacks regular and common trajectories among all users. On the other hand, this can be also attributed to our model, which considers solely locations and sequences of locations (trajectories), without for instance taking other information like time of day or day of week into account, that seemed to be helpful in the previous Chapter. Furthermore, adding biases in the factor equation or directly in the Markov model could further improve the model. On the whole, Matrix Factorization and FPMC in particular indicate a promising basis for future investigations in the field of semantic location prediction.

PART II

ARTIFICIAL NEURAL NETWORKS

This second part of this thesis discusses the use of Artificial Neural Networks (ANNs) in our semantic location prediction scenario.

Chapter 6 compares a Feed-Forward (FFNN), a Recurrent (RNN) and a Long Short-Term Memory (LSTM) neural network at two different semantic representation levels in terms of predictive performance. In summary, the LSTM model leads to the overall best behaviour. However, it also shows some deficits, especially at the lowest representation level.

Chapter 7 proposes the use of a Convolutional Neural Network (CNN) for modeling semantic trajectories. The evaluation results highlight certain advantages with respect to accuracy with the CNN being able to outscore the FFNN, the RNN and the LSTM of Chapter 6. However, in terms of precision and recall, the CNN based approach can't reach the levels of the LSTM. Chapter 7 also investigates and highlights the benefits of using an additional embedding layer as the input of the model.

Chapter 8 evaluates the Sequence to Sequence (Seq2Seq) learning approach as an extension of the LSTM model of Chapter 6 with respect to short- and long-term prediction. As in Chapter 7, the Seq2Seq model can in certain cases show a slightly better accuracy than the one of the FFNN, RNN and the standard LSTM of Chapter 6, but it lacks in terms of precision, recall and F-Score, both for the short- as well as the long-term prediction case.

Chapter 9 attempts to extend the spatial LSTM model of Chapter 6 by taking beside the location itself, additional context information into consideration as well. The respective set of information that is being evaluated in this chapter comprises beside time, day and activity from Chapter 3, psychological features such as the personality and the emotional state of the user. The results indicate a slight improvement compared to the spatial LSTM of Chapter 6. However, at the same time they highlight the need for a larger dataset due to the great number of feature dimensions that come with this approach.

Finally, motivated by the results of Chapter 4, Chapter 10 introduces a number of methods for optimizing the training and predictive behaviour of artificial neural networks by incorporating explicit semantic knowledge into it. It can be shown that this kind of *Semantic-Enhanced* learning can lead to shorter training times while keeping the prediction accuracy at the same or even bringing it to a higher level.

CHAPTER 6

APPLYING ARTIFICIAL NEURAL NETWORKS ON SEMANTIC TRAJECTORIES: FFNN, RNN AND LSTM

Abstract

In this chapter, we investigate the use of Artificial Neural Networks (ANN) in the field of semantic location prediction. We evaluate three different ANN types: Feed Forward Neural Networks (FFNN), Recurrent Neural Networks (RNN) and Long Short-Term Memory Networks (LSTM) on two different real-world datasets. At the same time, we explore the impact of the applied semantic representation level when modelling semantic trajectories by testing two different semantic levels as we did in Chapter 3. A Markov Chain model based predictor serves in both cases as our baseline. Neural networks show an overall good performance in predicting future semantic locations, with LSTM achieving the highest average score of 76,1%. Furthermore, it can be also shown that both time as feature and the semantic representation level have a significant effect on the results.

The content of this chapter is based on two publications, at ICANN 2017 [KSJB17] and at ACM SIGSPATIAL 2018 [KJB18]

6.1 Introduction

As we already saw before, there exists a big variety of approaches for modelling trajectories and estimating future locations. Each approach points out certain advantages and disadvantages depending on its model and the nature and size of the available data. For instance, it is well established from a variety of studies that *Artificial Neural Networks (ANN)* perform very well in the field of time series analysis and forecasting. Especially *recurrent* architectures show a good performance when it comes to modelling time series and other kind of sequences, such as word sequences (sentences) in the text mining domain, due to their feedback looping nature. This makes them an ideal candidate for modelling spatio-temporal sequences. In addition, mobility and trajectory analysis is generally considered to be a complex, nonlinear problem, which (non linearity) ANNs are also capable of handling well.

In this chapter we investigate the use of artificial neural networks in modelling semantic trajectories of mobile users and predicting their future semantic locations. For this purpose we implement and compare three different types of neural networks: a *Feed-Forward (FFNN)*, a simple *Recurrent (RNN)* and a special type of recurrent networks, the *Long Short-Term Memory (LSTM)* network. All three models are trained and evaluated on two different datasets: a 3-month long single-user dataset and the multi-user MIT Reality Mining dataset of [ESP06]. For each case, we evaluate two different models, one that takes temporal information (time and day) into account and one that doesn't. A probabilistic Markov model serves as a further baseline. Furthermore, similar to Chapter 3, we explore the impact of using different semantic levels when modelling our trajectories. It can be shown that neural networks perform overall well in predicting the next semantic location, with the LSTM model outperforming in average the competition.

The rest of this chapter is organized as follows. Section 6.2 highlights some of the most related work in location prediction with ANNs. Next, Section 6.3 goes deeper into the theoretical background and the implementation of our models. Section 6.4 gives a comprehensive view on our evaluation results, while lastly Section 6.5 outlines our overall conclusions.

6.2 Related Work

This section discusses some of the most important related works that apply neural networks for predicting future locations.

Biesterfeld et al. were one of the first that used neural networks to address the topic *location prediction*. In their work [BEJ97], they investigated several variants of feed-forward and recurrent networks. They showed that neural networks perform generally well when it comes to motion pattern representation, with the feed-forward networks achieving surprisingly the best results. In [VGPU04], within the scope of building an indoor location prediction system, Vintan et al. apply a 3-layer feed-forward Perceptron and investigate both separate models for each of the user, which they call *local predictors*, as well as a joint model composed of all of them, the *global predictor*. Their results indicate a higher accuracy on the part of the individualized models by showing an accuracy up to 92%. In addition, a learning rate of 0.1 and a history length of 2 visited rooms proved to be the best parameter values. Within the scope of the Nokia Mobile Data Challenge in 2012 [LGPA$^+$12], Etter et al. compared in [EKK12] three different models for solving a location prediction task. A 3-layer feed-forward ANN achieved the best results, outperforming both a Dynamic Bayesian Network (DBN) and a Gradient Boosted Decision Trees (GBDT) that served as baseline. Petzold et al. arrived at a similar conclusion in [PBTU06] some years earlier, as they investigated how various location prediction approaches perform on a dataset of 4 users. Both neural networks, a feed-forward and an Elman recurrent model, landed again at the top of the accuracy score list surpassing a Dynamic Bayesian Network (DBN), a Markov Chain model and a Finite Automata state predictor. An interesting hybrid two-stage approach comes from Vukovic et al. in [VJ15]. They combine a probabilistic model with a feed-forward model through a regularity checking component that decides whether the current movement of a user is regular or not based on her history. Then, in the next step, in case it is regular, a feed-forward neural network takes over the prediction task by considering both current location and time and day. Otherwise the next location is estimated based on a probabilistic model regardless of the current location. They evaluated their approach on the MIT Reality Mining dataset [ESP06] and show

varying results; while some users achieve accuracy values of 20-30%, other users come up with 80-85%. The performance dependency on the user is a common issue in location prediction and can be mainly attributed to the degree of regularity that each user shows in his movement patterns, as well as to the quality and consistency of the respective available data. Song et al. describe in their recent work [SKS16] a deep LSTM-based neural network architecture for simulating and predicting large-scale human mobility and transportation. Their model uses data from approx. 1.6 million users collected over three years in Japan in combination with transportation network infrastructure information. They evaluated their system against a Hidden Markov Model (HMM) and a Gaussian Model (GM) as well as with a shallow (1-layer) LSTM network and a Time-Delay Neural Network (TLDM). The deep LSTM model achieved the best accuracy scores. Yao et al.'s work, which also focuses on modelling semantic trajectories [YZHB17] is based on a recurrent neural network model as well. Their model features an embedding layer in order to enhance the performance of their model. A similar embedding layer based approach is presented in Chapters 7 and 8. Finally, Plummer and Wong et al. prove in their work the overall suitability of artificial neural network models for modelling time series and providing short- and long-term predictions upon them [Plu00, WXC10].

6.3 Neural Network Based Semantic Location Prediction - Theory, Implementation, and Parameter Selection

In this chapter, we implement and compare in total three different semantic location predictors based on three different ANN architecture types respectively: a Feed-Forward, a simple Recurrent and a LSTM network. This section describes our implemented models and their corresponding parameters, while giving a short glimpse behind the theory of each at the same time. In order to have a fair comparison, we used the same 3-layer architecture (one hidden layer) for all. The number of input and output neurons $N_i = N_o = N$ complies with the number of different semantic locations $l_i \in L \subset \mathbb{N}$ found in each training and test dataset respectively. We used *one-hot encoding* in order to

represent the $|L|$ semantic locations for both our input as well as our output vector. Our predictors were evaluated once with and once without taking temporal information (time and day) into account. In the case, in which time and day are being considered, our (one-hot) input vector grows by $7 + |timeslots|$, with $|timeslots| = 24$, since we chose to aggregate time into hourly slots.

6.3.1 Feed-Forward Neural Networks (FFNN)

Feed-Forward neural networks represent the simplest type of neural networks, but nonetheless they feature a good performance across multiple domains. A typical multi-layer FFNN can be seen in figure 6.1.

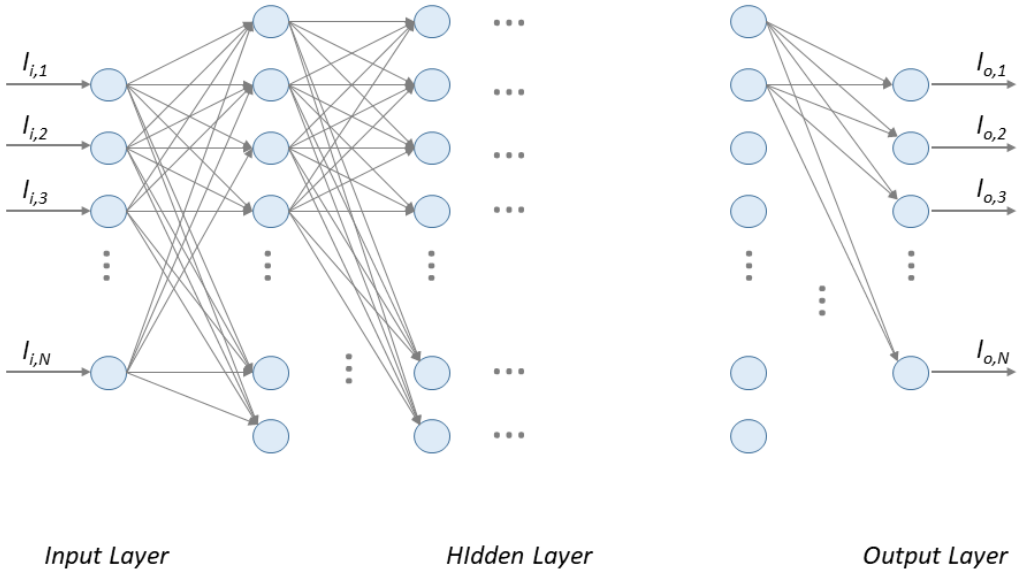

Figure 6.1: Visualization of a typical deep multi-layer FFNN.

In order to be able to take beside the current location past locations additionally into account, we extended the input vector and hence the number of input neurons by the factor h for covering the movement history of the users resulting in a $|L| \cdot h = N \cdot h$ long vector (not to be confused with h_t, which represents the state of a hidden neuron at time t (see Eq. 6.1)). For instance, if we want that our model considers both current and the previous location, then $h = 2$ and thus we need $N \cdot 2 = 2N$ input neurons.

A Feed-Forward Neural Network constitutes the simplest type of a multi-layered neural network. FFNNs are directed acyclic graphs (DAGs), in which

the signal flows through the network in only one direction, forward, from the input layer through the hidden layer(s) to the output layer. Eq. 6.1 describes how the output state of a (hidden layer) neuron in such a Feed-Forward network is being updated at each time step.

$$h_t = f_{act}(W_{x \to h} \cdot x_t + b) \tag{6.1}$$

The current state h_t depends on the current signal x_t fed into it and eventually on a bias b. $W_{x \to h}$ and f_{act} represent the weight matrix and the activation function respectively. Typical activation functions are the step function, the sigmoid and the tanh function, as well as the rectifier found in Rectified Linear Units (ReLUs). The *Sigmoid* function of Eq. 6.2 and the *Backpropagation (BP)* are used in this chapter as our model's activation function and learning algorithm accordingly.

$$\sigma(z_k^{(i)}) = \frac{1}{1 + \exp^{-z_k^{(i)}}}, \tag{6.2}$$

with

$$z_k^{(i)} = W_k^T \cdot a^{(i-1)} + b_k, \tag{6.3}$$

whereby $z_k^{(i)}$ represents the weighted and biased average of the *activations* $a_k^{(i-1)} = g^{(i-1)}(z_k^{(i-1)})$ from the previous layer $i-1$ for neuron k in layer i. W_k and b_k represents the corresponding weight matrix and bias respectively.

6.3.2 Recurrent Neural Networks (RNN)

In contrast to Feed-Forward networks, Recurrent Neural Networks (RNNs) take the previous states h_{t-1} additionally into account (see Eq. 6.4). $W_{h \to h}$ represents the internal state weight matrix. This makes them particularly efficient when it comes to modelling sequences, such as DNA sequences as well as speech and spatio-temporal sequences (trajectories) like in our case.

$$h_t = f_{act}(W_{x \to h} \cdot x_t + W_{h \to h} \cdot h_{t-1} + b) \tag{6.4}$$

The recurrent behaviour can be expressed either through a feedback loop or in an unfolded form as illustrated in Fig. 6.2. We used a similar one-hot encoding as previously by the FFNN, except that now, due to the nature of RNN, we don't have to explicitly extend the input vector by the previously visited places h. In addition, *Backpropagation Through Time (BPTT)* was selected to be our

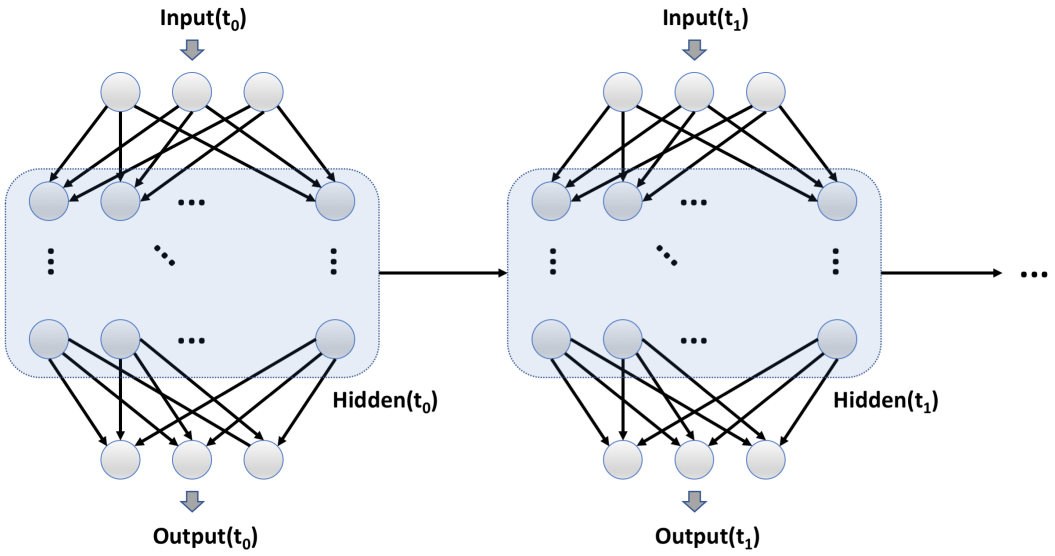

Figure 6.2: Unfolded RNN.

learning algorithm. BPTT was first introduced by [RHW85]. The underlying idea is to unroll the RNN, calculate and add up the error among the considered training steps, roll the network up and update its weights.

Recurrent networks bring certain limitations with them, such as the *exploding* or the *vanishing gradient* issue when trained with the gradient descent algorithm. The *exploding gradient* refers to a situation, in which accumulated error gradients result in extreme high weights that makes the updating and thus the network itself unstable. This can be avoided by simply clipping the gradient value at a certain fixed maximum value. The *vanishing gradient* on the other hand describes the fact, where the error gradients become so small that the model can't learn any more (or it would take too long for learning something) [Hoc98]. Moreover, in the case of RNNs, the more the network gets evolved in time, the less impact a certain input way back in the past has on the current output of the model. In our case, where we apply a RNN for modelling sequences of locations and for providing an estimation about future locations, this means that locations that lie in the far past would influence the location prediction outcome less than the later locations, a fact which in turn affects substantially the overall prediction performance, especially for long term relationships. This *decaying error* effect with respect ot recurrent network types was highlighted in [HS97a]. There exists a number of methods that attempt

to overcome this issue. The Long Short-Term Memory approach is one of the most efficient ones and it is described briefly in the next section.

6.3.3 Long Short-Term-Memory Neural Networks (LSTM)

Long Short-Term Memory networks constitute a special type of recurrent neural networks, which are capable of storing longer sequences without showing the fading long-term dependency effect that appears in the simple recurrent architectures. They were first introduced by Hochreiter et al. in [HS97a] and [HS97b].

LSTM networks extend the standard recurrent networks by featuring a memory storing and managing mechanism, the *gated cell*. This mechanism gives them the ability to store inputs and previous states for a long period of time, as well as to delete them when necessary.

$$h_t = f_{act}(c_t) \cdot o_t \tag{6.5}$$

Eq. 6.5 shows how the current state of a hidden layer neuron is influenced by the current output gate o_t of the cell, which in turn depends on the forget f_t and the input i_t gate value (Eq. 6.6).

$$c_t = f_t \cdot c_{t-1} + i_t \cdot f_{act}(W_{x \to h} \cdot x_t + W_{h \to h} \cdot h_{t-1} + b) \tag{6.6}$$

$W_{x \to h}$, $W_{x \to h}$ and f_{act} represent again the corresponding input-hidden, hidden-hidden layer weight matrices and the activation function respectively. In tangible terms, Eq. 6.6 explains how a LSTM cell is able to forget previous cell states and store new input. The remember-forget memory feature leads to better modelling results and raised LSTMs to one of the top choices for modelling and predicting temporal sequences.

6.3.4 Hyper Parameter Selection

Taking primarily Vintan et al.'s findings in [VGPU04] into consideration, who showed that individual models outperform a general model, we concentrate ourselves on creating individual models for each user. We applied a 10-fold cross-validation on the Reality Mining dataset described in Section 6.4 in order to find the best hyper parameter setup for each of our models. In tangible

terms, we trained our models in 90% of the data and tested them on the remaining 10% for each possible combination of following parameter values: numbers of hidden neurons: [16, 32, 64, 128], epoch: [10, 20, 50, 100, 150], learning rate: [0.005, 0.01], history: [1, 2, ..., 16] and batch size: [1, 10, 50]. As already mentioned at the beginning of Section 6.3, we kept a similar shallow 3-layer architecture for all network types. A brief investigation showed no significant improvement when we used deeper models. This can be partly attributed to the relative small size of our dataset and the low number of features (unique location classes). The training dataset was build up through random selection of sequences of a certain length, which depends on the history h that our model takes each time into consideration. The setup that led to the average highest accuracy and F-Score performance was selected for each of the three ANN types respectively. This procedure took place twice, one for each semantic representation level (see Section 6.4). Table 6.1 summarizes the results.

	ANN type	hidden neurons (or cells)	epoch	learning rate	weekday	time	history	batch size
low sem. level	FFNN	32	100	0,01	no	no	2	1
	RNN	32	10	0,01	no	no	15	1
	LSTM	16	10	0,005	no	no	2	50
	MM	-	-	-	no	no	1	1
high sem. level	FFNN	32	10	0,01	no	no	9	1
	RNN	64	150	0,01	no	no	7	1
	LSTM	16	1,00	0,005	no	no	2	50
	MM	-	-	-	no	no	1	1

Table 6.1: Final parameter setup obtained as described in subsection 6.3.4. MM refers to the Markov Model, which is used as reference in our evaluation.

6.4 Evaluation

This section discusses the performance of FFNN, RNN and LSTM with regard to predicting the next semantic location. All three models were evaluated on two different datasets: the 3-months long single-user dataset of Chapter 3 and the multi-user MIT Reality Mining dataset [ESP06] described in Chapter 5. Each dataset contains semantically labelled locations recorded by the users, like “Hilton hotel", “Media Lab”, “MIT main campus”, “Dentist” etc.

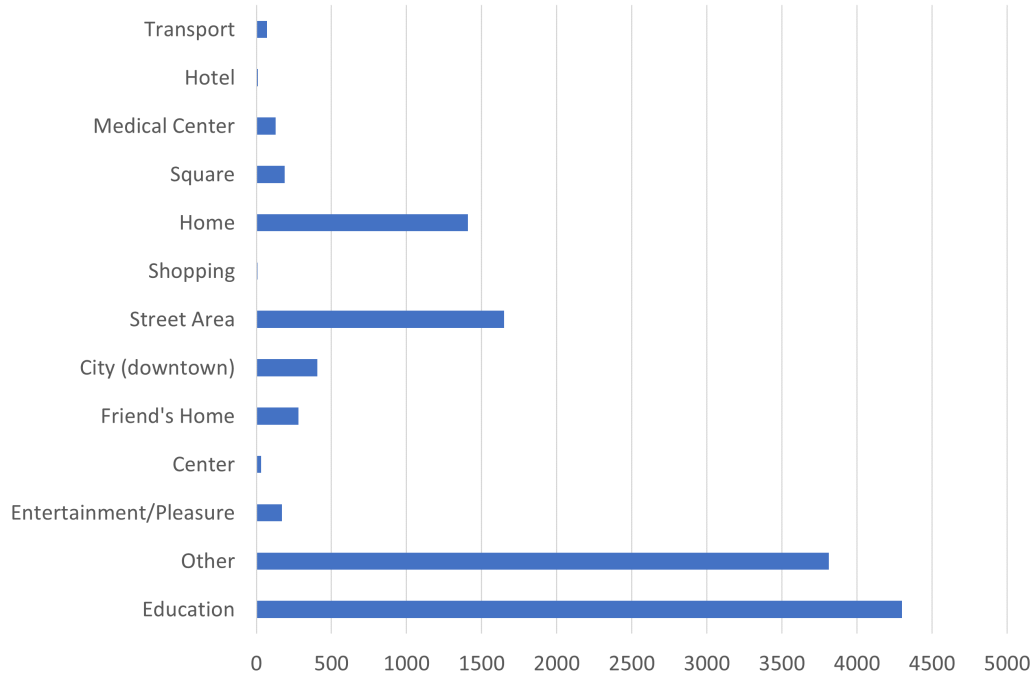

Figure 6.3: Distribution of the high-level semantic locations in the filtered MIT dataset.

In order to investigate whether and to what degree different semantic levels affect the prediction, we processed both datasets to derive a higher conceptual view upon them. For this purpose we applied a semantic clustering algorithm and assigned a *higher* label to each of the locations. We oriented ourselves on the location taxonomy of *foursquare*[1], which led us to the following higher semantic types: "Home", "Friend's home", "Education", "Medical", "Transport", "Shopping", "Square", "Street", "City", "Pleasure (Entertainment)", "Center", "Hotel" and "Other". These semantic annotations subsume the *lower* ones and represent therefore a higher-level location interpretation. Fig. 6.3 illustrates the overall high-level semantic location distribution.

In order to evaluate our predictive models, we applied the same 10-fold cross-validation process, as we did for finding the optimal hyper parameter setup described in Section 6.3.4. Each predictor was trained and tested separately on each user, which led us to 26 individual models per predictor. The parameter setup for each predictor can be found in table 6.1. Accuracy, precision, recall and F-Score were used as metrics for our evaluation. Fig. 6.4 and 6.5 show

[1]https://developer.foursquare.com

the results of our three models with and without taking time and day into consideration. What stands out is that in general, temporal information

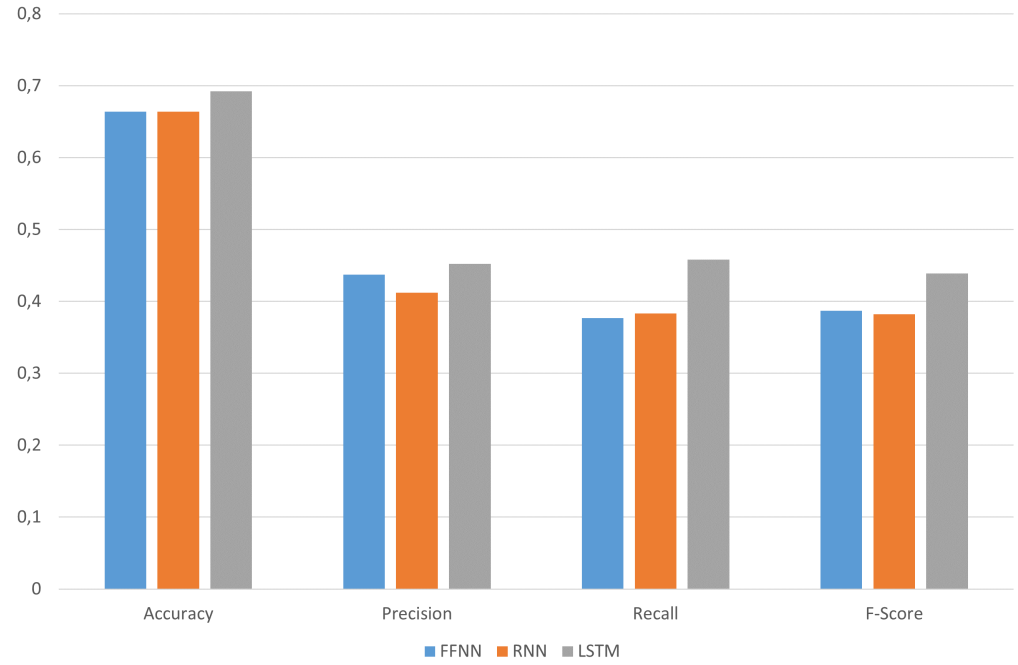

Figure 6.4: Average results over all single-user models taking temporal information into consideration at the higher semantic level (MIT Reality Mining dataset).

does not lead to significant higher scores as one would expect. In contrary, in most of the cases, temporal information seems to affect negatively the scores, especially for the LSTM model. This finding justifies our feature selection choice (time and weekday: "no") in table 6.1. Thus, one could infer that it is not the absolute time and day that is important, but the sequence itself. That is, it seems that *when it comes to semantic trajectory based prediction, the order of visiting the various semantic location classes is more important than the absolute time at which the respective locations are visited.*

Fig. 6.6 and 6.7 illustrate the respective results for the lower-level semantic representation case. If we compare the 4 figures, it is apparent that all three models perform much better with higher-level semantically annotated locations. This matches the findings in Chapter 3 and can be again attributed on the one hand to the fact that human mobility subjects rules and patterns of higher semantical order. On the other hand, we must again not forget that

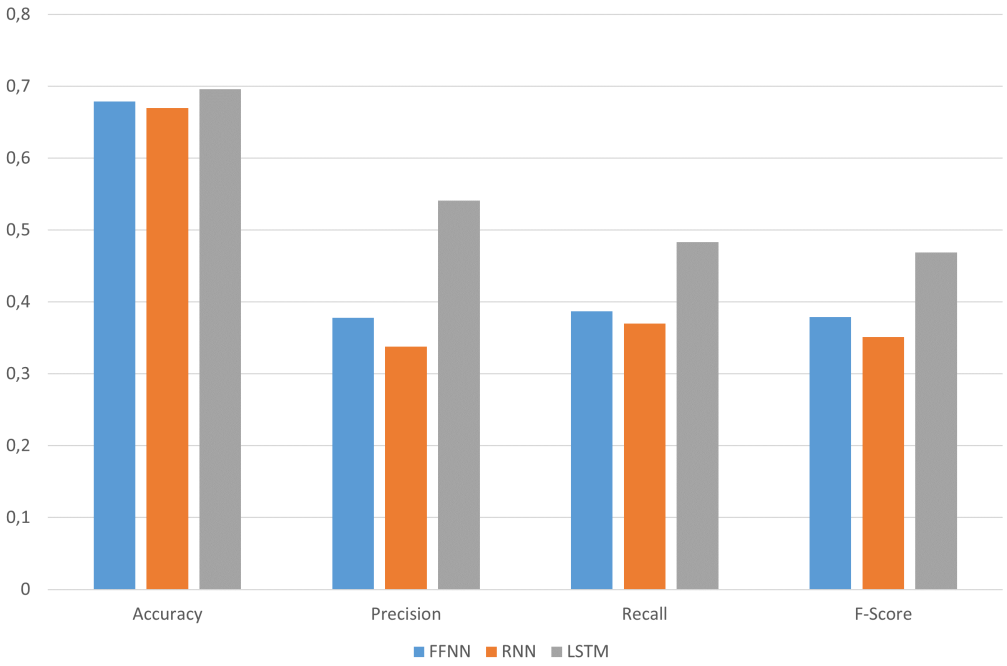

Figure 6.5: Average results over all single-user models without taking temporal information into consideration at the higher semantic level.

in the high-level case, the models have to deal with semantically clustered and therefore overall less locations, a fact that additionally favours the performance of the high-level models. What is also striking in these figures is that LSTM outperforms clearly the other models with regard to precision, recall and F-Score. This means that it is not only more accurate, but also more consistent, that is, its predictions scatter less. This can be attributed primarily to its solid memory-based architecture.

A closer inspection of the figures shows that other than expected both recurrent networks do not stand out significantly over the FFNN. In the high semantic level case, LSTM outperforms the other two and RNN shows the worst performance, whereas surprisingly the opposite is true for the lower one. Thus, the architecture type seems to play a significant role. Interestingly, the particular RNN takes beside the current location, 15 previous locations as input additionally into account (see Table 6.1). Thus, a RNN in combination with a long history seems to perform better than the LSTM, despite its limitation, the vanishing gradient, to which we referred to in Section 6.3.3. In tangible terms, we could say here that surprisingly, for the low-level case, it

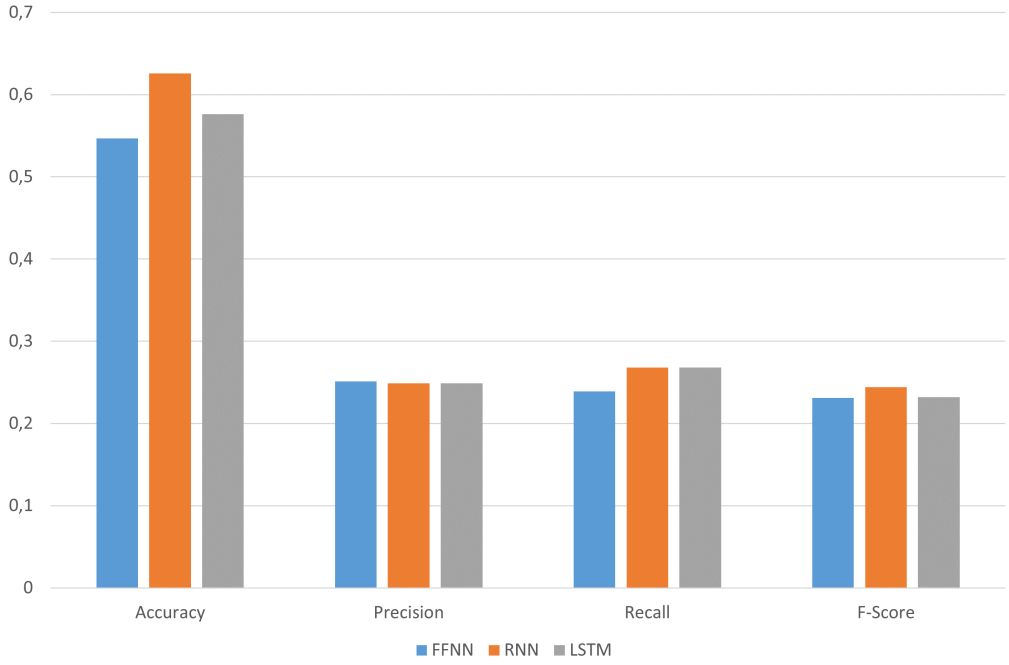

Figure 6.6: Average results over all single-user models taking temporal information into consideration at the lower semantic level.

seems that considering many locations and being *forgetful* at the same time leads oddly enough to the best result.

In addition, it must be also mentioned here, that the FFNN model provided respectively high results with a much higher epoch value (100, see Table 6.1) than RNN and LSTM (both 10), which makes it more CPU-intensive and time-consuming. However, recurrent networks are generally also slow to train due to their recurrent nature, especially when they train on longer sequences.

Table 6.2 provides a summary of the aforementioned statistics from the evaluation with the multi-user MIT dataset. It contains the average ascertained results among all users.

Fig. 6.8 illustrates our average findings of our evaluation with the single-user dataset. In addition to the neural networks, we compared each model against a Markov Chain (MM) based predictor. Table 6.1 shows the parameters of the Markov model that yield its best results and were used for our comparison.

Fig. 6.9 shows the maximum recorded prediction accuracy. The results correlate with the results displayed in the previous figures. The average scores achieved with the high-level semantic trajectories are again clearly higher com-

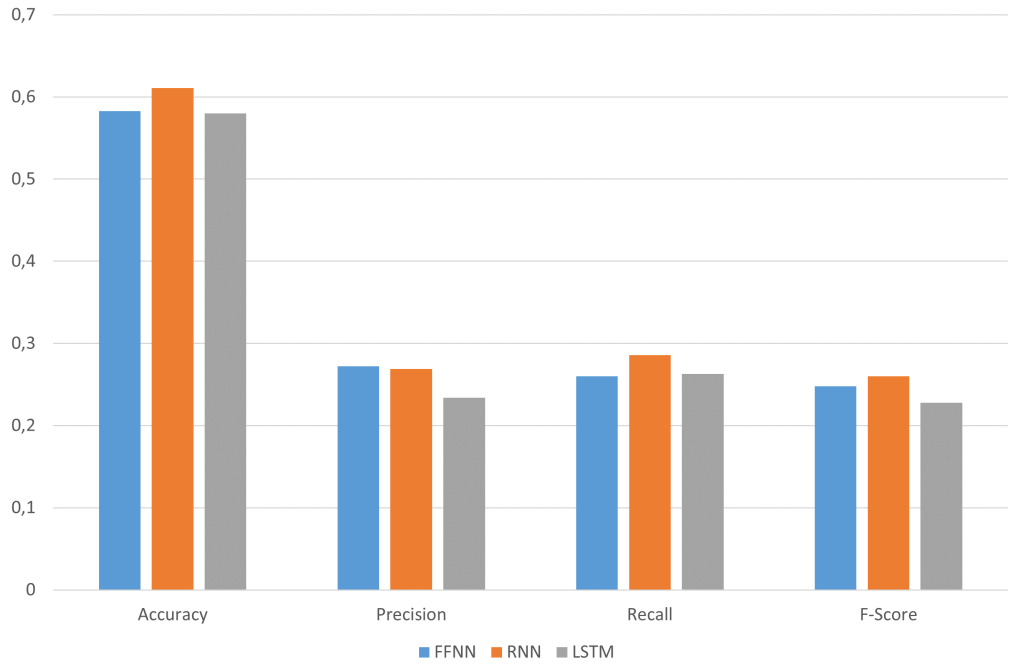

Figure 6.7: Average results over all single-user models without taking temporal information into consideration at the lower semantic level.

	ANN type	Accuracy	Precision	Recall	F-Score
low sem. level	FFNN	0,583 | 0,547	**0,272** | 0,251	0,260 | 0,239	0,248 | 0,231
	RNN	**0,611** | **0,626**	0,269 | 0,249	**0,286** | 0,268	**0,260** | 0,244
	LSTM	0,580 | 0,576	0,234 | 0,249	0,263 | 0,268	0,228 | 0,232
high sem. level	FFNN	0,679 | 0,664	0,378 | 0,437	0,387 | 0,377	0,379 | 0,387
	RNN	0,670 | 0,664	0,338 | 0,4124	0,370 | 0,383	0,351 | 0,382
	LSTM	**0,696** | **0,692**	**0,541** | 0,452	**0,483** | 0,458	**0,469** | 0,439

Table 6.2: Average statistic scores without and with taking time and day into account (1st and 2nd value respectively) (Reality Mining MIT dataset)

pared to the low-level ones. In addition, the LSTM model outperforms this time in both cases the FFNN and the simple RNN. The LSTM outperforms even the Markov Model in the high-level data case, which interestingly shows generally similar high accuracy values. The MM seems moreover to perform better than the rest ANN-based models on the low-level location types. In individual cases, the MM achieved a 100% accuracy. A possible explanation might be the size of the available single-user data. Artificial neural networks show their best performance in modelling big data, like the comparably huge

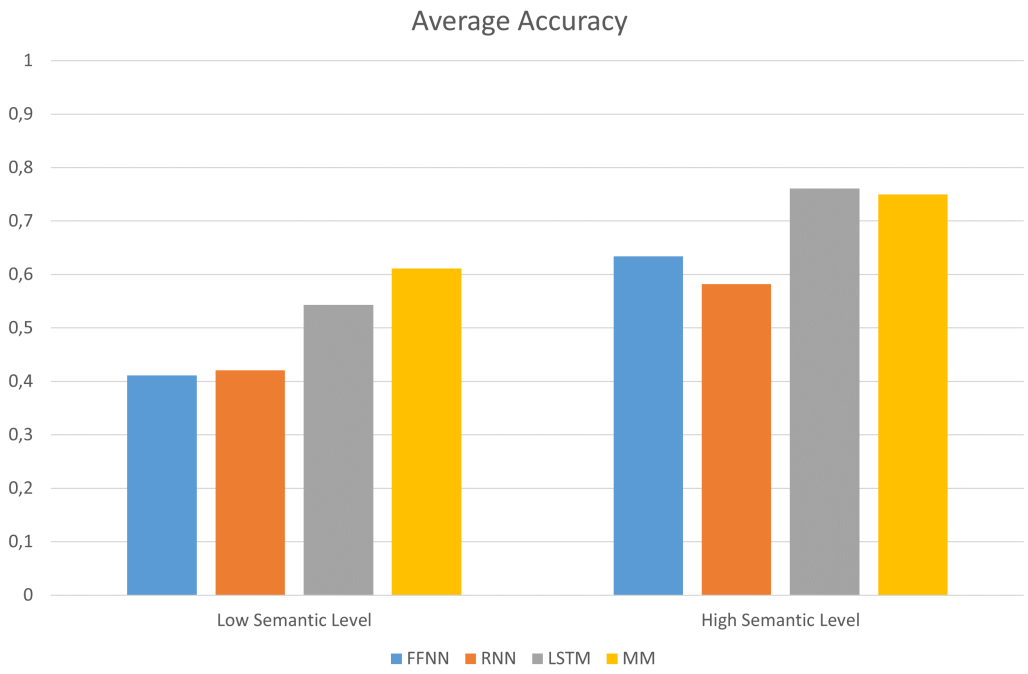

Figure 6.8: Average accuracy scores over 10 runs (3-month long single user dataset)

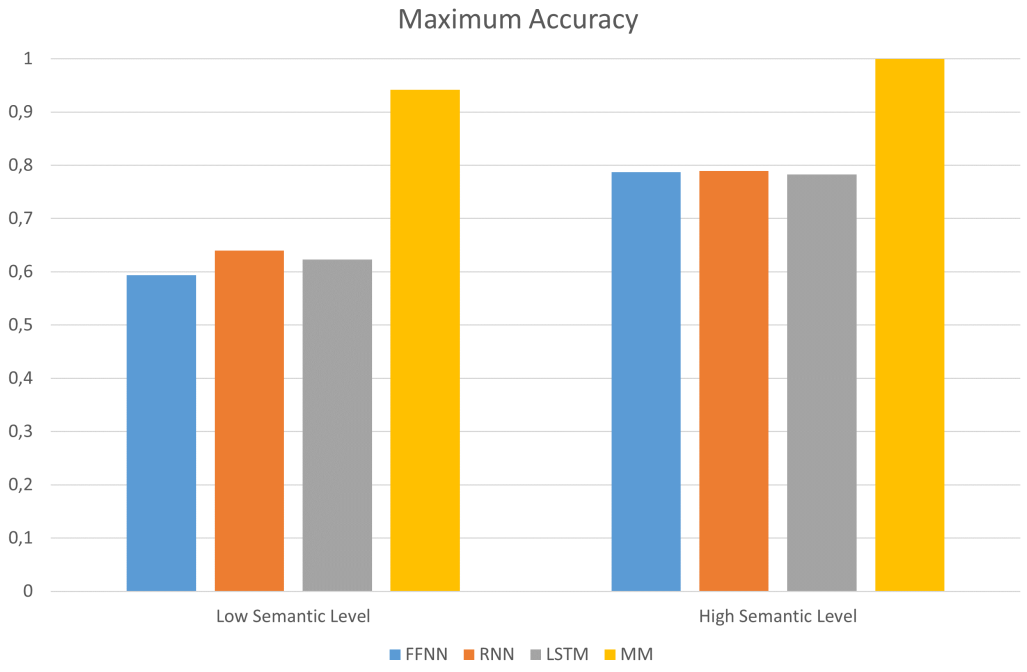

Figure 6.9: Maximum accuracy scores over 10 runs (3-month long single user dataset)

dataset in [SKS16], and our single-user dataset was probably not big enough.

6.5 Conclusion

The purpose of this chapter was to investigate the use of artificial neural networks in the field of semantic trajectory based location prediction. For this purpose three different neural network architectures were implemented: FFNN, RNN and LSTM. An evaluation at two semantic representation levels was carried out on two different datasets, a 3-month long single-user dataset and the MIT Reality Mining dataset. In addition, we compared our models against a Markov Model based predictor. Our results illustrate a good average performance of the ANNs, especially at the higher semantic layer, with LSTM reaching an average accuracy of 76,1% outperforming the competition. Furthermore, our results indicate a significant contribution of high-level semantic annotations of locations to the prediction. All in all, this study strengthens the idea of using ANNs for location prediction and highlights the importance of the semantic representation level in this field at the same time. However, significant performance differences could be identified between the different network types, a fact that promotes additionally the investigation of further neural network architectures. In Chapter 7, we explore the use of Convolutional Neural Networks, a network type commonly used in image classification and recognition, and evaluate the respective model against the three best models presented in this chapter.

CHAPTER 7

APPLYING ARTIFICIAL NEURAL NETWORKS ON SEMANTIC TRAJECTORIES: CNN

Abstract

As already seen in Chapter 6, Artificial Neural Networks have been proven to be a promising modelling approach when it comes to modelling semantic trajectories and predicting future locations. Recurrent network architectures show a particularly good performance. However, very little research has been done on the use of Convolutional Neural Networks (CNN) in connection with modelling human movement patterns. In this chapter, we introduce a CNN-based approach for representing semantic trajectories and predicting future locations. Furthermore, we consider and explore the use of an additional embedding layer with regard to a higher predictive performance. In order to evaluate our approach, we use the Feed-Forward (FFNN), the Recurrent (RNN) and the LSTM neural network of Chapter 6 as baselines to compare it with at two different semantic representation levels. It can be shown that CNNs are more than capable of handling semantic trajectories, while providing high prediction accuracies at the same time.

The content of this chapter is based on a publication at ICANN 2018 [KSB18a].

7.1 Introduction

Chapter 6 explores the use of Artificial Neural Networks (ANN) with respect to modelling semantic trajectories and predicting future semantic locations. After evaluating three different types of ANNs, a Feed Forward (FFNN), a Recurrent (RNN) and a Long Short-Term Memory (LSTM) neural network, it could be shown that ANNs are generally capable of providing high prediction accuracy scores with the recurrent and especially the LSTM network making the overall best impression. This outcome corresponds fully to the results of respective studies that apply neural networks for modelling time series and other similar sequences.

The use of recurrent neural network architectures for modelling 1-dimensional (1D) sequences has become a well established approach in recent years. As established as *Convolutional Neural Networks (CNN)* in the image processing domain are, for modelling 2-dimensional data. However, convolutional networks are not that popular in other domains such as in forecasting time series. And this, despite the fact that the moving filtering functionality of the latter could bring major benefits with it in the 1-dimensional use case as well.

In this chapter, we present and evaluate a Convolutional Neural Network architecture in a semantic location prediction scenario. In particular, we propose a CNN-based approach extended by an additional embedding layer for modelling semantic trajectories and providing an estimation about the next semantic location of a mobile user. We evaluate our approach against the FFNN, the RNN and the LSTM of Chapter 6 and it can be shown that the CNN is able to compete with them, especially when trajectories are described at a lower semantic level.

This chapter is structured as follows. First, in Section 7.2, we describe some related work that has been done in the realms of ANN-based semantic location prediction and CNNs. Next, in Section 7.3 we elaborate on the way in which CNNs work, by providing some relevant term definitions at the same time. In Section 7.4 we outline our own architecture together with some basic implementation details. Finally, in Sections 7.5 and 7.6, we discuss the outcomes of our evaluation and draw some final conclusions with regard to our findings.

7.2 Related Work

The most popular application area of Convolutional Neural Networks is the image classification and recognition [LKF10]. Lv et al. attempt to take advantage of the CNN's good performance in this field and explore in [LLW16] the possibility of using Convolutional Neural Networks (CNNs) to predict taxi trajectories. Their approach projects past trajectories upon a map and models them in turn as 2D images, on which the CNN is finally applied to estimate about future trajectories. By modelling trajectories as 2D images, they are able to make use of the inherent advantage of CNNs, namely their good performance in image analysis. This is also confirmed by their results. However, their approach is applied on raw, non-semantic GPS trajectories.

Beside image classification, CNNs can be applied on other areas as well, such as speech recognition and time series [LB+95]. However, to our knowledge, there is no work exploring the performance of CNNs on semantic trajectories. Moreover, it seems that there is no work using trajectories (semantic or non-semantic ones) in combination with CNNs directly, e.g., without transforming them in an intermediate step into 2-dimensional (2D) images as mentioned above, but handling them in their raw form instead, as 1-dimensional (1D) vectors. For this reason, the second half of the related work section highlights some work coming from a different domain, the sentiment analysis and the Natural Language Processing (NLP) domain, where, similar to our case, the data are also represented by 1D sequences and some significant work in combination with CNNs has already been done. Convolutional networks are often applied in the sentiment analysis domain. Santos et al. introduce in [SG14] a convolutional neural network sentiment analysis framework that uses a CNN to model text at three different levels at the same time: character-level, word-level and sentence-level. Their approach is able to outperform the respective state of the art and define new performance benchmarks for the Stanford Sentiment Treebank [SPW+13] and the Stanford Twitter Sentiment [GBH09] corpora. Kalchbrenner et al. present a convolutional network that uses a dynamic k-max pooling function capable of handling world relations of varying size [KGB14]. They evaluated their model with the same Twitter Sentiment corpus as Santos et al., among others, achieving a similar high performance.

Particularly interesting is the work of Collobert et al. [CWB+11], in which a CNN architecture is proposed for solving several NLP problems including named entity recognition and semantic role labelling. Their framework features an unsupervised training algorithm for learning internal representations, e.g., by using an embedding layer and learning low-dimensional feature vectors of given words through backpropagation, yielding a good performance both in terms of accuracy and speed. The benefit of using embeddings has been recently also bshown in connection with modelling human trajectories by Gao et al. in [GZZ+17] and by Yao et al. in [YZHB17].

7.3 Convolutional Neural Networks (CNN)

In this section, we provide a brief insight into the fundamental theory of a convolutional neural network and it's respective elements. As already mentioned in Section 7.2, the most popular domain for CNNs is the image classification and recognition domain, where data are 2-dimensional. A CNN example architecture concerning the image classification use case can be seen in Fig. 7.1.

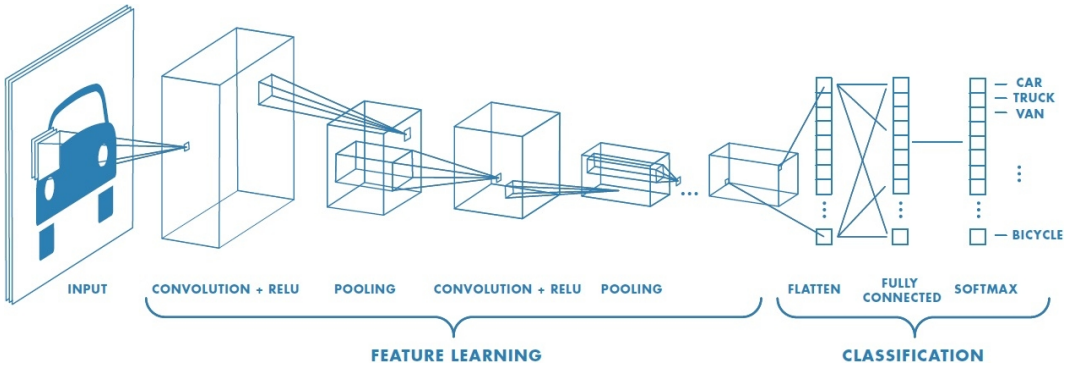

Figure 7.1: Typical CNN architecture for Image Classification (source: [Mat18]).

Here, the CNN first receives an image which is supposed to classify, as its input:

$$i = w_i \times h_i \times r_i, \tag{7.1}$$

with w_i and h_i the width and the height of the image (in pixels) and r_i the number of channels (e.g. in the case of RGB the number of channels is $r_i = 3$).

Next, a set of *convolution operations* takes place in order for the (first set of) features to be extracted. These operations are realized by k *filters* (or *kernels*) of fixed size $w_f \times h_f \times r_f$, containing adaptive, learnable weights, which are sled over the input image to "search" for certain features. The kernel size represents one of the most important hyper parameters of a CNN and depends primarily on the size and the content of the input images as well as the application itself. Usually, it varies from $2 \times 2 \times 3$ for small images and can go up to $5 \times 5 \times 3$ for large images.

Then, the convolved intermediate outcomes are passed over to an activation function (in this case a Rectified Linear Unit (ReLU) [HSM+00]). At the end, each convolution filter output results in a new layer that contains the findings of that filter in the input image. These layers are then further processed by a *pooling* operation set. Pooling operations combine multiple outputs from filter kernels in a feature layer into a single value (e.g. by taking the average (*average pooling*) or maximum value (*max pooling*) of the outputs in question). The resulting pooled layers can then be further processed by adding more *Convolution + Pooling* operations and thus building in this way a deeper model, as shown in Fig. 7.1, and as such, compound features of a higher level can be identified and extracted. The last pooled layer is flattened i.e. transformed into a single long vector containing all of its weights. These are then connected to a fully connected layer, which is further connected to the output of the network, which in this case is a Softmax layer [Bis06], containing a field for every classifiable object, and as such representing the classification estimation of the network for the given input.

7.4 CNNs on Semantic Trajectories

Our CNN model takes semantic trajectories as input, like the ones defined in 2.2. For this purpose, each semantic location is given a unique index. Each index value in the trajectory gets passed to a hash table, the *embedding layer*, which assigns each index, and as such each semantic location, a k-dimensional feature vector (*embedding vector*), whereby k represents a hyperparameter, which is set by us (see Section 7.5). At the very beginning, our feature vectors in the lookup table are randomly initialized. These vectors are then trained

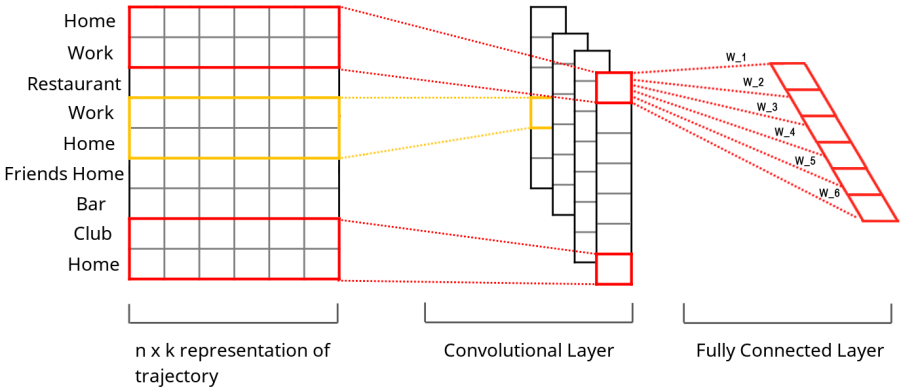

Figure 7.2: An abstracted view on the core layers of our CNN-based location prediction approach. A trajectory of size n with each location represented by an embedding vector of size $k < n$ is fed into the convolutional core of our model. (The image is based on [Kim14]).

using the available training data via backpropagation in order to become optimal task-specific representations. In tangible terms, for our case, this means that we give our model the freedom to find the optimal semantic location representation by itself. The resulting representations will be used as input for our core convolutional model. Using an embedding layer brings a twofold advantage with it. First, it leads to a dimension reduction in contrast to using one-hot encoding as in Chapter 6, since the length of the embedding vector k is lower than the size of a raw one-hot encoded input vector. Second, in contrast to the sparse one-hot encoded representation of semantic locations, an embedding layer builds dense vectors with *each* value contributing to the learning process, a fact that leads generally to a more "symmetrical" and thus more efficient training of machine learning models. A similar idea was proposed by Collobert et al. in [CWB+11] to learn feature vectors that represent words in a text corpus for solving NLP problems. After the embedding process, our semantic locations set that composes the input trajectory and which was initially represented by a $n \times 1$ vector, becomes a $n \times k$ matrix. This can be seen on the left of Fig. 7.2 and is referred to as *self.embedded_locs_expanded* in Listing 7.1.

In the next step, a set of *convolutional filters* is applied on the resulting matrix. These filters span along the entire feature vector dimension and across

multiple locations of the trajectory as can be seen in Fig. 7.2. The number of the applied filters itself is a hyper parameter that can be also set by the user. Like the size k of the embeddings dimension described above, it can also affect significantly the performance of the prediction.

The outputs of the filters are then concatenated and flattened (see Listing 7.1) to make up a fully connected layer. Finally, these are linked to a Softmax output layer, which provides the final estimation about the next semantic location to be visited by a user. We decided against using pooling layers on the filter outputs, since this led to the loss of significant feature information (e.g., locations in the later part of the trajectory being more important to location prediction as the older ones). Finally, the resulting output weights are flattened into one long vector. To prevent overfitting, we applied the dropout method [SHK+14] on this flattened vector as shown in Listing 7.1 in line 14.

In order to train our model, we used backpropagation in combination with the Adam optimizer [KB14]. The Adam optimizer maintains an individual learning rate for each network weight and adapts them separately. This proved to be especially effective because our dataset is quite sparse compared to other more typical problems addressed by CNNs such as image recognition. Python and the TensorFlow[1] library were used to implement our model.

Listing 7.1: Convolution output and flattened layer.

```
# Convolution Layer
self.conv1 = tf.layers.conv2d(
inputs=self.embedded_locs_expanded,
filters=num_filters,
kernel_size=[filter_size, embedding_size],
padding="VALID",
name="conv1")

# Combine all the features
filter_outputs_total = num_filters * ((sequence_length -
    filter_size) + 1)
self.h_pool_flat = tf.reshape(self.conv1, [-1,
    filter_outputs_total])

# Add dropout
```

[1] https://www.tensorflow.org

```
self.h_drop = tf.nn.dropout(self.h_pool_flat, self.
    dropout_keep_prob)
```

Listing 7.2 illustrates the implementation of the fully connected layer. W and b represent the weights and the offset respectively. Furthermore, we used TensorFlow's *nn.softmax_cross_entropy_with_logits* and *reduce_mean* functions to calculate the loss. The calculated loss is used by the Adam optimizer to adjust the weights of the TensorFlow graph, and as such to complete a single training step.

Listing 7.2: Fully connected layer and loss calculation.

```
# Final (unnormalized) scores and predictions
W = tf.get_variable(
"W",
shape=[filter_outputs_total, num_classes],
initializer=tf.contrib.layers.xavier_initializer())
b = tf.Variable(tf.constant(0.1, shape=[num_classes]), name="b")
self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores")
self.predictions = tf.argmax(self.scores, 1, name="predictions")

# Calculate mean cross-entropy loss
losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.
    scores, labels=self.input_y)
self.loss = tf.reduce_mean(losses)
```

7.5 Evaluation

In order to evaluate our approach, we used the same MIT Reality Mining dataset [EP06] of Chapter 5. We then extracted from the resulting dataset trajectories of a fixed length and considered the subsequent location to be the ground truth prediction label (see Fig. 7.3). We shuffled the resulting (trajectories, label) pairs and took 90% of them for training and 10% for testing. We trained and evaluated both the individual single-user models separately as well as a multi-user model trained with the trajectories of all users. In the case of the multi-user model, a single trajectory composed of all the available single-user trajectories was fed into the model as if it came from a single user.

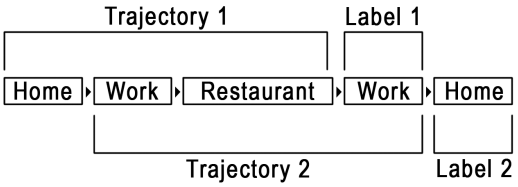

Figure 7.3: Fixed-length trajectory extraction with a trajectory length of 3.

In addition, since there is a timestamp present for every location visit in the Reality Mining dataset and similar to Chapter 6, we also tested the performance of our model for the case in which time is considered as an additional feature. For this purpose, we aggregated the available timestamps into hourly time slots. Finally, we evaluated a version of our model with the embedding layer missing, in order to define the role of embeddings in terms of predictive performance. We compared our approach with the Feed-Forward (FFNN), the recurrent (RNN) and the Long Short-Term Memory (LSTM) model from Chapter 6, which serve here as our baseline. All models were evaluated in terms of *Accuracy*, *Accuracy@k*, *Precision*, *Recall*, and *F-Score*. We tested several trajectory lengths (2, 5, 10 and 20) on different configurations of the following hyper parameters:

Filter Size: Width of the filter kernel, i.e. how many locations (or complete trajectories) it encompasses.
Number of Filters: The number of different filters the CNN learns.
Embedding Dimension: The dimension of the learned location representations.
Dropout Probability: The percentage of neurons in the fully connected layer that are dropped (used to minimize overfitting).

At the same time, we performed an exhaustive grid search to find the following optimal parameters: **Learning Rate**, **Number of training Epochs** and **Batch Size**. Both the results with respect to the trajectory length and the corresponding optimal parameter set can be found in Table 7.1. All fur-

Trajectory Length	Accuracy	Accuracy@4	Accuracy@10	Precision	Recall	F-Score
2	0.783	0.**976**	0.994	0.455	0.433	0.443
5	0.790	0.973	0.995	0.466	**0.439**	**0.451**
10	**0.792**	0.971	0.994	**0.467**	0.435	0.45
20	0.788	0.968	0.993	0.454	0.425	0.438

Table 7.1: Impact of trajectory length. Filter Size: 2, Embedding Dimension: 100, Number of Filters: 50, Dropout Probability: 0.4, Batch Size: 100, Learning Rate: 0.001, Number of Epochs: 10.

ther CNN results in this section base on this optimal parameter and hyper parameter set.

In general, it seems that the longer the trajectory the better our model performs with regard to almost all of our metrics, that is, accuracy, precision, recall and F-Score. However, if they get too long, e.g., > 10, the performance seems to drop again, especially in terms of recall and F-Score. This could be attributed to the fact that human movement is characterized, up to a certain length, by a long-term behaviour and thus raising the considered trajectory length in the model leads to an improved predictive performance. The overall trajectory length seems to be making little difference though, and a trajectory length of 2 is enough to predict the next location with an accuracy of approximately 78%.

In Fig. 7.4 we can see the results of our model, with and without an embedding layer. Both CNN models, with and without embedding layer, outperform the FFNN of Chapter 6 with regard to all of our metrics. Moreover, the Embedding Layer seems to be giving a slight performance boost to our model. Thus, the dense representation of location and trajectories seems indeed to positively support our model.

Fig. 7.5 contains the average outcome (over all users) of our model in the single-user model case in contrast to the FFNN, RNN and LSTM architecture for the low semantic representation level. Our CNN outperforms the other ANN architectures in terms of accuracy by 7-8%, but falls a bit short in terms of precision, recall and consequently F-Score. This could be interpreted as an indication that the CNN is worse at finding the relevant locations out of certain single-user datasets during the prediction phase due to being sparse (compared to the rest single-user datasets) and thus it is worse at predicting

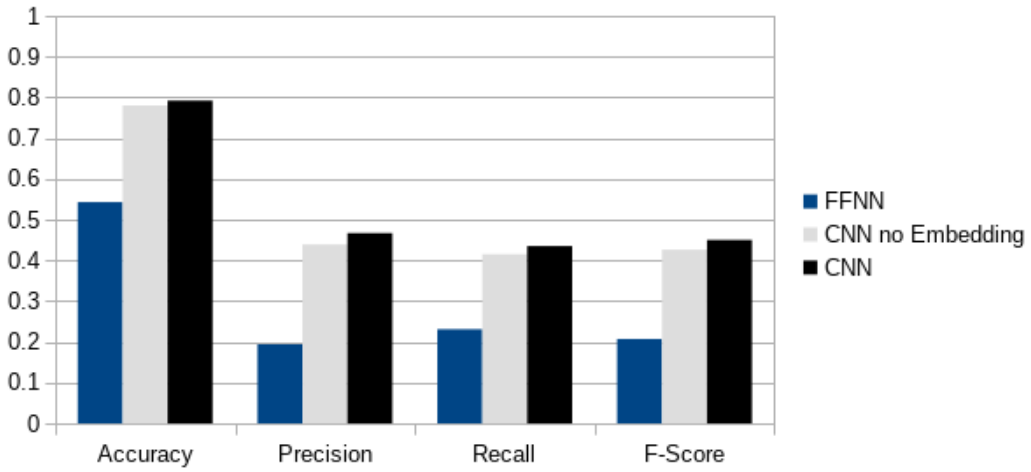

Figure 7.4: Comparison of evaluation results of our architecture with and without embedding layer vs. FFNN.

the respective location transitions compared to the other ANNs. This poor performance due to the few sparsely filled individual models seems to pull the average of our CNN model over all single user models down. Another possible explanation for this might be the fixed size of the filters that go over the single locations of the trajectories covering sometimes *only a part* and sometimes *more than one* daily trajectory. At the higher semantic level the

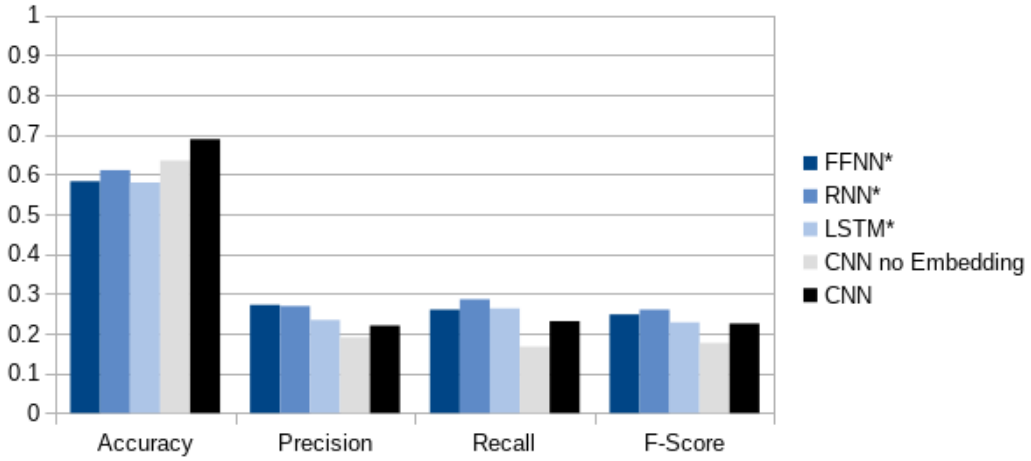

Figure 7.5: Evaluation results of CNN vs. FFNN, RNN and LSTM of Chapter 6 (average over all single user models at the lower semantic level).

accuracy discrepancy between the various models is similar to the low semantic version (see Fig. 7.6). However, in terms of precision, recall and F-Score the

CNN seems to perform much worse than at the lower semantic representation version. It seems to disregard high-level locations that occur relatively seldom in the dataset almost completely, which leads us to this result. In both versions of the dataset (low and high semantic level) the embedding layer seemed to make a small, but still significant positive difference.

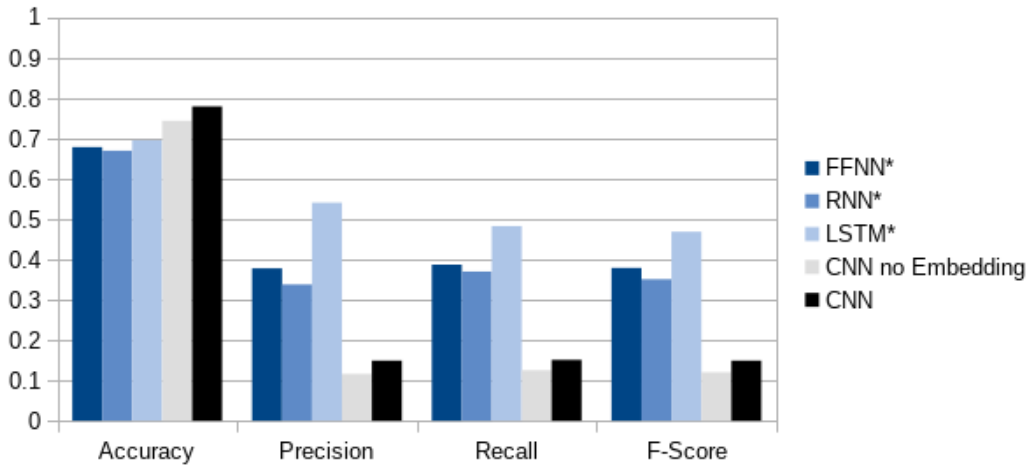

Figure 7.6: Evaluation results of CNN vs. FFNN, RNN and LSTM of Chapter 6 (average over all single user models at the higher semantic level).

Fig. 7.7 contains the comparison results between the average of the single-user models with the multi-user modelling method. While the multi-user evaluation achieves much lower accuracy scores (as expected, since it attempts to fit the pattern of many users simultaneously), it outperforms by far the average of the single-user models in terms of precision, recall and F-Score. This can be attributed to the fact that the additional user information in the multi-user model, that comes from training with data of many, fills the gap of missing locations and trajectories, which can be often found in the single-user models. This trade-off issue can often be found in the user modelling literature when it comes to comparing a generic model with a personalized one.

Finally, in Fig. 7.8, it can be seen how adding time as an additional training feature affects the behaviour of our models. Similar to the results in Chapter 6, time seems to be having a negative influence on the prediction performance of our CNN model, both in terms of accuracy and F-Score. Thus, this underpins again the fact that in semantic location prediction the order of visiting the various semantic location types is more important than the absolute time at

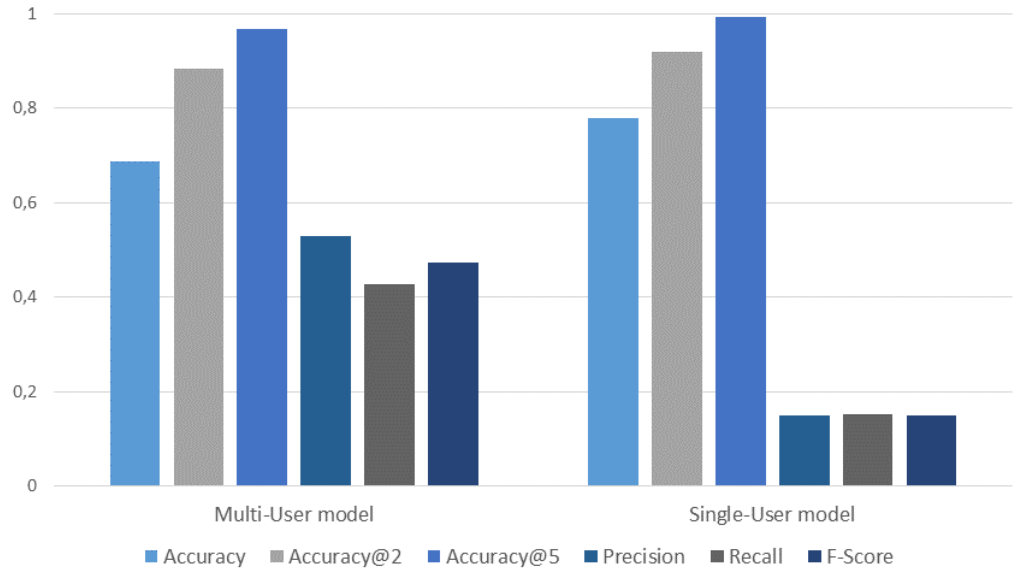

Figure 7.7: Comparison of the Multi-User model with the average performance of the Single-User CNN models.

which the respective locations were visited.

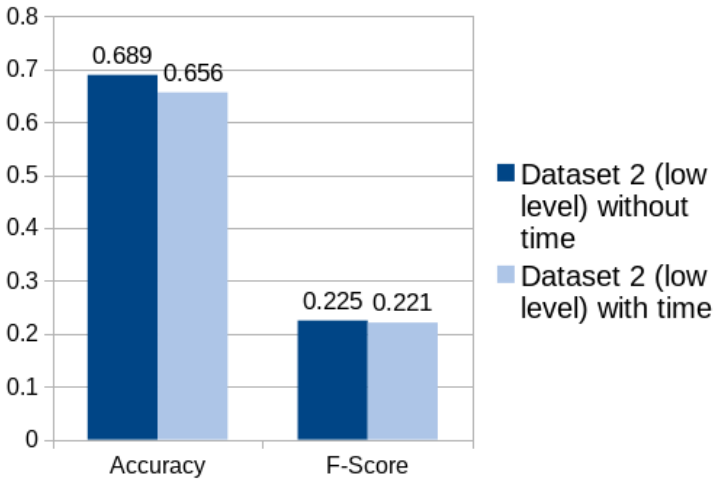

Figure 7.8: Impact of time in the case of the low-level semantic representation.

7.6 Conclusion

In this chapter, we investigate the performance of Convolutional Neural Networks and the use of an embedding layer with respect to modelling semantic

trajectories and predicting future locations in a semantic location prediction scenario. We evaluate our approach on a real-world dataset, using the FFNN, the RNN and the LSTM network of Chapter 6 as our baselines. It can be shown that the proposed CNN-based model outperforms all the above reference systems in terms of accuracy reaching up to approx. 8% higher scores and is thus capable of modelling semantic trajectories and predicting future human movement patterns. However, our approach seems to be sensitive to sparse data, a limitation represented by relative low recall and F-Score values. In addition, we show that, similar to the outcomes of Chapter 6, both the semantic representation level and the time as an additional feature, as well as the overall number of users considered for training the model can have a significant impact on the performance, especially with regard to precision and recall.

CHAPTER 8

APPLYING ARTIFICIAL NEURAL NETWORKS ON SEMANTIC TRAJECTORIES: SEQUENCE TO SEQUENCE (SEQ2SEQ) LEARNING

Abstract

Chapter 6 and 7 focus on training Artificial Neural Network (ANN) models with semantic trajectories for predicting a *single* future location at a certain time. It could be shown that ANN-based modelling can lead to overall high semantic location prediction scores. In this chapter, we explore a method that trains the core model to predict *whole sequences* instead of single semantic locations. After that, we chose the highest ranked location to be the final *next semantic location* estimation. For this purpose, we extend the Long Short-Term Memory network from Chapter 6 by applying *Sequence to Sequence (Seq2Seq) learning*, a method widely applied in the area of text mining, on human semantic trajectories. In particular, in this chapter, we explore whether and to what extent sequence to sequence learning in combination with neural networks can contribute to improving the accuracy in a location prediction scenario. We compare the performance of our Seq2Seq learning framework with the performance of a standard LSTM, a semantic trajectory tree-based approach and a probabilistic graph of first and higher order on two different real-world datasets. It can be shown that Sequence to Sequence learning may well be used to model semantic trajectories and predict future human movement patterns.

The content of this chapter is based on a publication at ACM SIGSPATIAL 2018 [KJB18].

8.1 Introduction

Several different models and approaches have been applied for modelling semantic trajectories so far. These include probabilistic graphs, decision trees and artificial neural networks (ANNs), with the latter performing overall best. Particularly recurrent topologies , such as Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) models, show a very good performance as also shown in Chapter 6. However, to our knowledge, none of the previous work has explored the use of *Sequence to Sequence learning* on modelling and predicting semantic trajectories.

Sequence to Sequence learning (Seq2Seq) is a widely applied method in the field of machine translation and question and answering (Q&A) systems. It was specially designed to learn and predict 1-dimensional sequences, such as complete sentences in a certain text or speech, or DNA sequences. Motivated by the fact that semantic trajectories represent 1-dimensional sequences as well, in this chapter, we investigate whether and to what degree Seq2Seq learning may be used for modelling human semantic trajectories. For this purpose, we propose an Attention- and LSTM-based Seq2Seq approach, which we evaluate on two different real-world datasets. In addition, we compare the performance of our method with the performance of a standard LSTM neural network as the one presented in Chapter 6, a probabilistic Markov Chain model of 1. and higher order as well as with a semantic trajectory mining and tree-based framework. It can be shown that Seq2Seq learning can lead to high accuracy scores and may well be used on semantic trajectories predicting the next location of users.

The rest of this chapter is structured as follows. In section 8.2, we reiterate briefly some of the most related ANN-based work in the field of location prediction from Chapter 6 and we go through a selected group of Seq2Seq papers from the text and speech mining domain. Section 8.3 describes the theory behind the two basic concepts of our approach: the sequence to sequence learning approach and the attention mechanism. Section 8.4 discusses in detail our architecture, while section 8.5 covers our evaluation and the respective results. Finally, in section 8.6 we summarize our conclusions and discuss potential improvements and future work.

8.2 Related Work

Chapters 6 and 7 address a number of semantic trajectory modelling and location prediction approaches based on different types of Artificial Neural Networks. In addition, further ANN-based related works are discussed in Sections 6.2 and 7.2, such as the work of Yao et al. in [YZHB17], where a Recurrent Neural Network architecture is combined with an *embedding layer*, similar to the one applied both in Chapter 7 (see Section 7.4) and in this one (as we shall see later), to enhance the performance of their model. Gao et al. use an embedding layer as well in order to overcome the training limitations that bring sparse one-hot encoded trajectories with them [GZZ+17]. Furthermore, recent work, like Song et al.'s work in [SKS16] demonstrate that Long Short-Term Memory (LSTM) models are more than capable of outperforming the competition in trajectory modelling.

The aforementioned work refers to models that are trained on trajectory data for predicting a *single* next location. In this chapter, we investigate whether training a model for providing estimations about *whole trajectory parts* instead of single locations may lead subsequently to a more accurate next semantic location prediction. For this purpose, we adopt the *Sequence to Sequence learning (Seq2Seq)* method (see 8.3.1) for our semantic trajectory use case.

Sequence to Sequence learning with neural networks was first introduced in 2014 by Sutskever et al. in [SVL14] as a solution for mapping sentences to sentences. In particular, Sutskever et al. applied successfully 2 LSTM models in an encoder-decoder architecture for translating automatically English sentences to French ones. Their approach could improve the performance of a sole LSTM model. Bahdanau et al. propose a similar machine translation model, which other than in [SVL14], it does not have to encode the source sentences into fixed-length vectors, achieving comparable results [BCB14].

Wang et al. use an attention-based Seq2Seq approach for modelling the online users' purchased or viewed item history and providing a likelihood for future item vectors within the scope of building a recommendation system [WC17]. Their approach is able to outperform state of the art recommenders based among others on Collaborative Filtering and Matrix Factorization.

Yao et al. propose an interesting time- and space-invariant way for clustering GPS trajectories in [YZZ+17]. Their approach reduces the GPS trajectories in sequences of features that depend neither on time nor on space. After that, Seq2Seq learning is used for learning fixed-length vector representations upon which is being finally clustered. Finally, an interesting work comes from Tang et al. in [TXMO16], where an attention-based Seq2Seq learning model is used for time series classification.

8.3 Background Theory

This section provides insight into the theory behind the two basic concepts described in the presented work, the *Seq2Seq learning* approach and the *attention* mechanism.

8.3.1 Sequence to Sequence Learning (Seq2Seq)

Sequence to Sequence learning (Seq2Seq) refers to a machine learning technique introduced by [SVL14] and found primarily in machine translation [SVL14, BCB14], automatic image captioning and question-answering systems. Seq2Seq models are designed and trained for mapping input sequences to output ones. A Seq2Seq model consists of two parts, the so called *encoder* and *decoder*, as shown in Fig. 8.1. These can be either simple single-layer RNNs or LSTMs, or multi-layer stacks of them. The encoder reads the input sequence and produces a matching internal representation of it as its output, e.g., the last hidden h_n and memory cell c_n states in the LSTM case. The decoder uses the produced internal state of the encoder to initialize its own internal state and to subsequently estimate step by step the correct output sequence in an iterative process. Often, the decoder part is designed as an autoregressive model, in which the prediction output of previous steps serve as input for the following ones (see Fig. 8.1).

A deeper insight into the operating principle of sequence to sequence learning (with respect to our use case) can be obtained in Section 8.4.

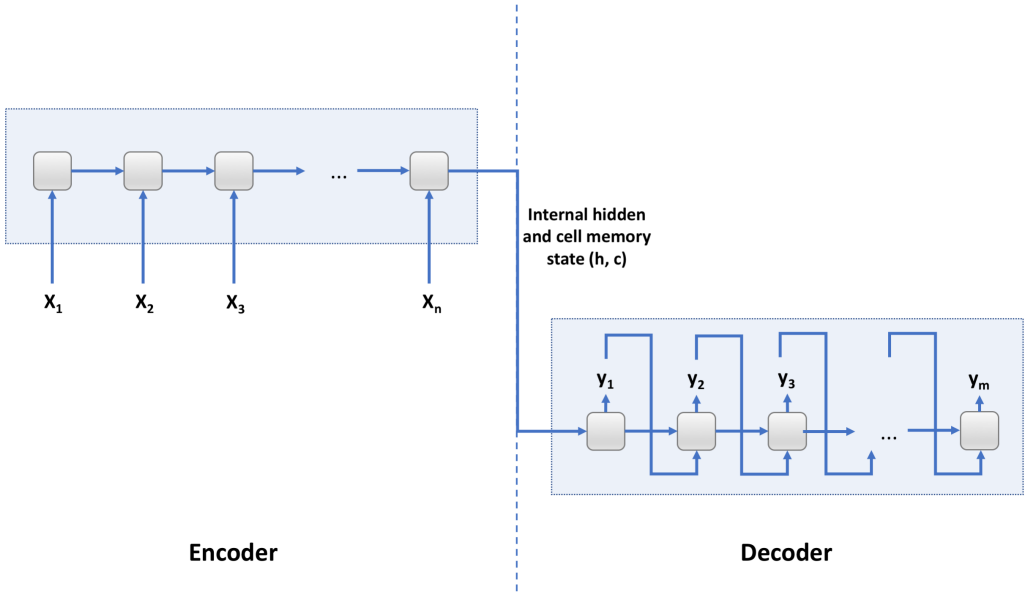

Figure 8.1: Unfolded RNN- or LSTM-based Seq2Seq model.

8.3.2 Attention

Long input sequences can lead the Seq2Seq model to a bottleneck situation in terms of modelling performance. This can be avoided by applying the *attention mechanism* proposed by Bahdanau et al. in [BCB14]. The attention method extends the Seq2Seq model by telling it at each time where exactly in the raw encoder sequence should pay attention to, "neglecting" the rest. In particular, the attention mechanism informs the decoder of the positions of the elements in the encoder's input sequence where the most significant information could be located at each time step. The more a certain piece of the encoder sequence contributes to the generated feature in the decoder side at a certain time, the more relevant and thus the more significant this is. This attributes to an improved modelling performance and a higher estimation accuracy. The attention mechanism can be found in other fields as well, such as in [XBK+15, YHG+16], where a so called visual attention mechanism is applied on images for improving an automatic image captioning system.

8.4 Seq2Seq Learning on Semantic Trajectories

The architecture of our Seq2Seq learning approach consists of two parts, a preprocessing and a training-prediction module (see Fig. 8.2). The preprocessing module serves the cleansing and the semantic annotation of the data collected by the users. It accepts both raw GPS data tuples in the form of ($user_{id}$, $timestamp$, gps_{long}, gps_{lat}), as well as pure semantic location data tuples in the form of ($user_{id}$, $timestamp$, $location_{id}$, $semanticLocation$). In a first step, the data are cleaned by filtering out all double, inconsistent and invalid entries. Next, locations that meet certain conditions (e.g., distance, or absurdly fast location changes within a temporal interval) are bundled into single locations. In the case of GPS data, locations are clustered based on the DBSCAN algorithm proposed in [EKS+96]. In our work we used $\epsilon = 0.001$ and $minPts = 1$ as values for the maximal distance between points in the same cluster and the minimum number of points per cluster respectively. After all the single locations are determined, the semantic enrichment or verification process takes place, in which the locations receive their final semantic annotation. This happens in a semi-supervised manner using the Google Places API [Inc] (distance-based, setting a radius of 5m) and filling the gaps when necessary (e.g., in case of private places like "home", "work", etc.) by taking the user's annotation into account. In a final step, a list of all semantic sequences (trajectories) is generated and forwarded to the training module. For this purpose each semantic location receives a unique ID and is formatted as a one-hot encoded vector.

The training and prediction module has the task to train a Seq2Seq model based on the provided semantic trajectories and subsequently to provide predictions about the users' future locations. The training module takes a list of semantic location sequences $S_i = \{s_1, s_2, ..., s_{|S|}\}$ of a length of $|S|$ and the corresponding target (to be learned) sequences $\{T_I = t_1, t_2, ..., t_{|T|}\}$ as its input. Both sequences are fed in batches $B_i = \{(S_1, T_1), (S_2, T_2), ..., (S_{|B|}, T_{|B|})\}$ of length $|B|$ into the module.

The core of our model adopts the attention-based approach applied for machine translation presented in [BCB14] and [LPM15]. It consists of the following components: *embedding layer*, *encoder*, *decoder*, *Attention layer* and

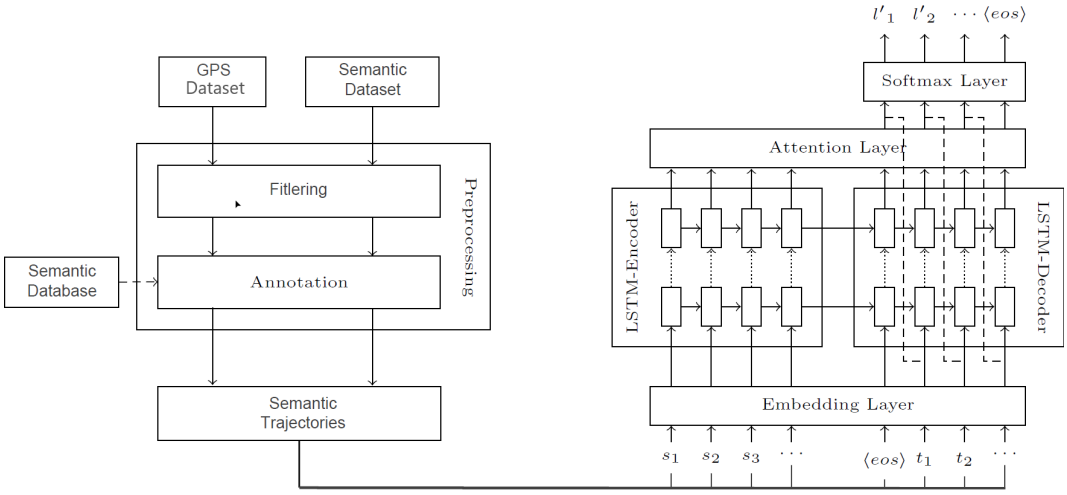

Figure 8.2: Attention-based Seq2Seq learning on semantic trajectories.

Softmax layer. Goal of the *embedding layer* is to produce dense representations out from the sparsely populated one-hot encoded vectors s_i, the so called embedding vectors v_e, in order to improve the overall performance of the model. The embedding vectors are initialized randomly and trained together with the rest of the model to map the input vectors.

The *encoder* is in our case a multi-layered LSTM neural network, which reads the input sequences step by step. The internal state of the LSTM cell at the end of the reading process $c_{|B|}$ represents the imported input sequence. This allows input sequences of variable length. Additionally, at each iteration k with $(1 \leq k \leq |S_i|)$, the LSTM's output $E_i = \{e_1, e_2, ..., e_{|Si|}\}$ is stored for the attention layer (see below).

The *decoder*, analogous to the encoder, consists of a multi-layered LSTM neural network as well. Its primary task is to generate the output sequence. At first, it is initialized with the encoder's last internal state $c_{|B|}$. An End-of-Sequence symbol $\langle eos \rangle$ initiates the decoding process. The *attention layer* takes both the decoder's and the encoder's output and builds a further fixed-length representation α, the so called *attention vector*. In contrast to $c_{|B|}$, which is taken only at the beginning of the output sequence generation process into account, the attention layer considers all the past input at each iteration step. This helps improve the prediction accuracy as already mentioned in Section 8.3.1. Moreover, it allows our model to learn complex interrelations between the location sequences in our training data. Apart from the attention

layer, our decoder is *auto-regressive*, which means that the output of previous states is used as input for the next ones. This helps further improve the prediction quality of our model.

Finally, the output of the attention layer is passed on to the *Softmax Layer*, that computes a probability distribution over the possible next sequences, of which we are interested only in their first element, the next locations $\hat{l}_I$. The distribution is computed based on the *Softmax* function found in Eq. 8.1.

$$\hat{l}_i = softmax(W_s \cdot a_i) \tag{8.1}$$

We chose the Log-Loss function from [SVL14] to be our loss function $Loss(\hat{L}, T)$ over the predicted $\hat{L} = \{\hat{l}_1, \hat{l}_2, ..., \hat{l}_{|\hat{L}|}\}$ and the target ltrajectories $\hat{T} = \{\hat{t}_1, \hat{t}_2, ..., \hat{tl}_{|\hat{T}|}\}$ during the training (see Eq. 8.2).

$$Loss(\hat{L}, T) = -\sum_i t_i \cdot log(\hat{l}_i) \tag{8.2}$$

An error signal based on the loss function value is backpropagated at each training step via an optmization method. Our module supports the *Stochastic Gradient Descent* method, as well as the *Momentum*, the *Adadelta* and the *Adam* optimizer.

During the prediction phase, the decoder applies the *Beam-Search* method described in [R+77] with $k = 2$ (instead of *greedy decoding* that is used during the training), in order to make our model more performant. Beam Search generates at each iteration k of the most likely locations based on the existing sequences so far. After going through all the iterations, we end up at the last iteration with a relative small group of trajectories that are worth considering. The trajectory with the highest overall probability gives the final outcome of Beam Search, whereas the first element represents our model's final estimation about the user's next location.

8.5 Evaluation

We used two different real-world datasets to evaluate our Seq2Seq approach, the open-source MIT Reality Mining dataset [EP06], which we already addressed in Chapter 5 and a dataset that we collected on our own and which we

	LSTM units	LSTM layers	sequence length	batch size	learning rate	embedding vector size	epochs	order	min. support	alphaS	alphaT
Value range	2, 4, 8, 16, 32, 64, 128	1-4	2, 3, 4, 10, 25, 50	16, 32, 64, 128	0.05, 0.075, 0.1	0, 2, 4, 5, 8, 10, 15	1, 5, 10, 20	1-5	0.005, 0.01, 0.05	0.1-0.8	0.1-0.8

Table 8.1: Range of investigated model parameter values.

	LSTM units	LSTM layers	sequence length	batch size	learning rate	embedding vector size	epochs	order	min. support	alphaS	alphaT
Seq2Seq ST	8	2	2	16	0.1	0	10	-	-	-	-
Seq2Seq ST (+emb)	32	3	2	32	0.1	8	10	-	-	-	-
Seq2Seq LT	8	2	10	128	0.075	10	10	-	-	-	-
LSTM ST	128	1	2	16	0.1	0	10	-	-	-	-
LSTM ST (+emb)	64	1	2	16	0.1	8	10	-	-	-	-
LSTM LT	128	1	25	16	0.1	15	10	-	-	-	-
Ying et al. [YLT13]	-	-	-	-	-	-	-	-	0.01	0.6	0.4
Markov	-	-	-	-	-	-	-	3.	-	-	-

Table 8.2: Optimal model parameter values for the multi-user models.

refer to as Sentient Destination Prediction (SDP) dataset and is described thoroughly in Chapter 9. The SDP dataset was generated by a 2-month long user study with 21 participants and includes beside locations and time, their activities as well as information about their personality and their emotional states. Both datasets were preprocessed in order to filter out illogical and inconsistent entries and annotations. For each dataset we developed and evaluated both a single multi-user model and the corresponding multiple single-user models separately, using a training/testing data ratio of 70/30 (%). Furthermore, we investigated our model's capability in modelling short and long semantic trajectories (*short-* and *long-term prediction*). A standard LSTM approach, a probabilistic Markov Chain model of 1. and higher order, as well as the semantic trajectory mining- and tree-based framework of Ying et al. in [YLT13] serve in this chapter as our baseline. Both for our Seq2Seq approach and for our baselines we conducted an exhaustive grid search to determine the optimal parameters of each model that lead to the best outcome. For this purpose and to avoid statistical outliers we ran each of the models with the same configuration a total of three times. The results listed in this section correspond the resulting average values. Table 8.1 contains the range of the tested parameters. The final best parameter values can be found in Table 8.2. Table 8.2 contains the multi-user model parameters. The individual single-user model parameter sets show similar values.

Fig. 8.3 shows the performance of the Seq2Seq model in comparison with the performance of a standard LSTM model, a Markov Model of 3. order and Ying's semantic trajectory based framework [YLT13] on the MIT Reality Mining dataset in terms of accuracy, precision, recall and F-score. In the

presented work, we used macro-average to preserve a class-dependent view on the overall performance. On average, the neural network models lead to the highest scores outperforming both the Markov Chain Model and Ying's tree-based approach. Our evaluation revealed that at the beginning, our LSTM-

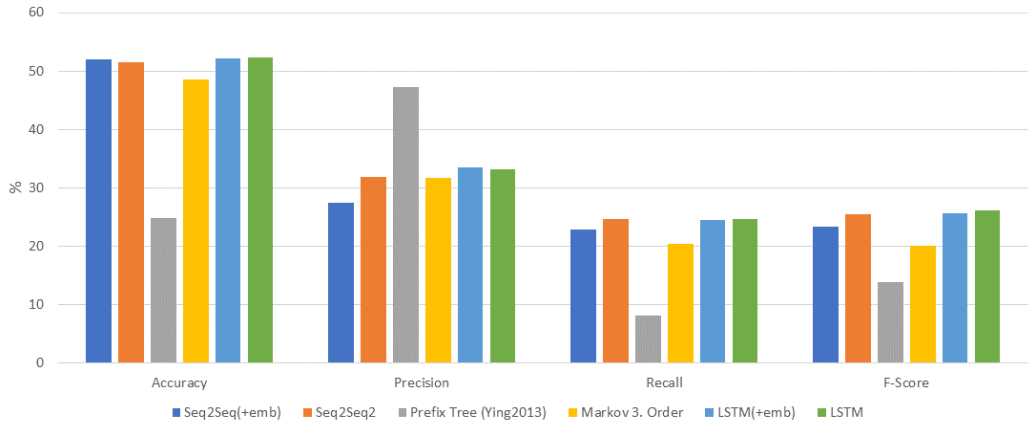

Figure 8.3: Multi-user model average prediction performance with and without embeddings (MIT dataset).

based Seq2Seq model was slightly better than the standard LSTM model with exact the same parameters. However, as we increased the number of the LSTM units, the standalone LSTM model became as good or even slightly better than the Seq2Seq model. Fig. 8.3 illustrates the best LSTM case with a LSTM unit number of 128 in contrast to the 8 units used in the Seq2Seq model. We can see that the LSTM reaches the same high accuracy and recall values as the Seq2Seq model, while outperforming it slightly with regard to precision. This results to a marginal increase of the overall F-score on the part of the LSTM. However the 128-unit LSTM required a higher amount of computation training time than the 8-unit Seq2Seq model due to its higher number of parameters. In addition, the slightly lower performance of the Seq2Seq model could be partly attributed to the fact that it is trained to learn primarily relations between location sequences. This fact by itself does not explain the low performance. But if we consider, that the number of trajectories and thus the number of the corresponding trajectory transitions in the available data is smaller than the single location transitions used by the LSTM to train itself, we could say that a Seq2Seq model needs far more data to bring out all its benefits. Although

the Reality Mining dataset is a relative big dataset, it is by far not so large like the ones found in the text mining field, where Seq2Seq models have long been established. At the same time, we investigated the impact of using a self-learned embedding layer for representing the locations in our datasets. As already mentioned in Section 8.4, embedding layers are usually used to help overcoming performance issues and improve the training process, a fact that we could also observe in our evaluation. But interestingly and unlike our findings in combination with the Convolutional Neural Networks (CNN) in Chapter 7, the use of embeddings (*Seq2Seq(+emb)* and *LSTM(+emb)*) seems to have a certain adverse effect on the neural network models, especially in the case of Seq2Seq learning. Solely in terms of accuracy show some kind of improvement in both models. Thus, it seems that the feed forward propagating CNNs are able to handle the dense embedded location representations better than the recurrent LSTM model. This could be partly explained by the relative small size of our datasets and the more "cumbersome" training of the latter due to its backpropagation through time (BPTT) learning algorithm. The probabilistic Markov model yields also relative high accuracy and precision values. With regard to recall though, it lies lower than both the Seq2Seq and the LSTM model. One of the most surprising facts were the extremely high precision figures given by Ying's approach. These match the findings in Chapter 4. This could in a way be understood as a very consistent predictive performance, in which most location transition estimations fall close together. However these figures come together with an extremely low recall value, a fact that complies with the precision-recall trade-off theory.

As mentioned above, we also implemented and evaluated separately multiple single-user models, that have been trained solely on the data of single users. Depending on the quantity and the quality of the data provided by each user, we could observe both very high scores, as well as very low ones, a fact that is documented in similar research works as well and can be mainly attributed to the annotating consistency of the respective users. Fig. 8.4 compares the top outcome of our four models for the best single-user case. What stands out in Fig. 8.4 is that both the Seq2Seq and the LSTM model, as well as the Markov approach achieve in the best single-user case a similar high accuracy, approx. 75.8%. In terms of precision, recall and the overall F-score, the Seq2Seq model

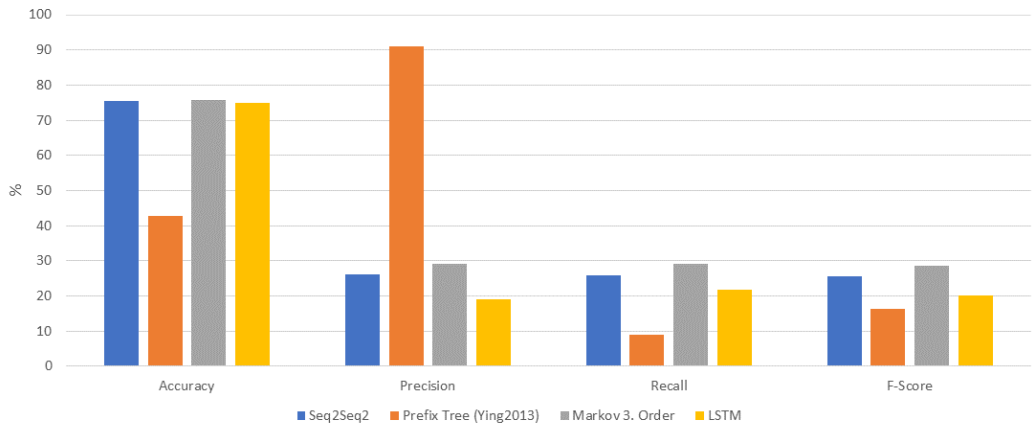

Figure 8.4: Top single-user model prediction performance (MIT dataset).

is better than the LSTM model, with the Markov model slightly outperforming both of them. Ying's approach shows a similar behaviour to the multi-user case shown in Fig. 8.3. Again, with respect to precision, it stands out from the rest, providing an extremely high precision of 91%, while showing very low recall figures at the same time.

We also investigated the long-term modelling behaviour of the models. For this purpose, we tested feeding and predicting upon very long, up to 50-semantic-locations long, semantic trajectories. A typical daily semantic trajectory consists in average of 3 (e.g., "home" → "work" → "home") to 6 (see Fig. 2.1) semantic locations. Both the Markov Chain model and Ying's framework showed a decreased performance the longer the sequence was. For this reason, we focused on the neural network models, which due to their memory-based architecture of the LSTM and unlike the simple RNN, are capable of handling long-term relationships. The most interesting findings can be found in Fig. 8.5 for the case of a 25-locations long considered history. The most surprising and striking observation to emerge from the results was that in respect of all metrics, the standard LSTM performed better than the more "sophisticated" Seq2Seq model. Apparently, the Seq2Seq model was not able to find long sequences to train and predict on evenly long sequences compared to the LSTM that is trained to predict a single future location. Recent work on machine translation based on Seq2Seq learning show that feeding the encoder with sequences of reversed order may lead to better results, especially when it comes

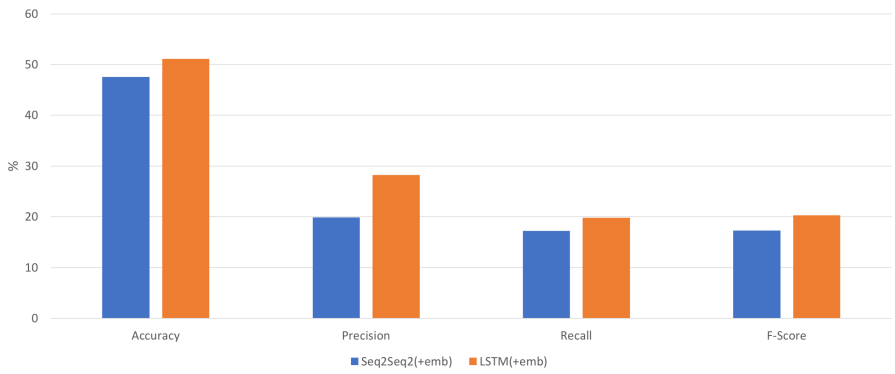

Figure 8.5: Long-Term multi-user model prediction performance (sequence length = 25, MIT dataset).

to modelling long sequences, because of introducing shorter-term dependencies between the input and the output [SVL14]. However, after experimenting with reverse trajectories, our investigation showed no improvement in contrast to keeping the location in the trajectories as they were. But at the same time, further analysis showed for the case of long-term prediction and in contrast to the short-term prediction findings, a significant improvement attributed to the embedding layer. The embedding layer could in the case of long trajectories show its strength by providing dense representations of the longer sparser trajectory vectors and consequently contribute to a more efficient training. Fig. 8.5 displays the long-term performance of the best LSTM model, a 128-unit model, in contrast to the best and more "light-weighted" Seq2Seq model with only 8 units.

Finally, Fig. 8.6 contains the multi-user modelling performance of our four models on the SDP dataset. In this case, it can be seen that all four models perform equally good when it comes to accuracy. Particularly interesting is the fact that Ying et al.'s tree-based approach outperforms this time both the Markov and the LSTM model. It seems that the SDP dataset, in contrast to the Reality Mining dataset, fulfills the right conditions for their trajectory mining based approach. The corresponding precision, recall and F-score figures though are extremely low. Solely Ying's framework remains consistent to the former evaluation results and features a very high precision. The particu-

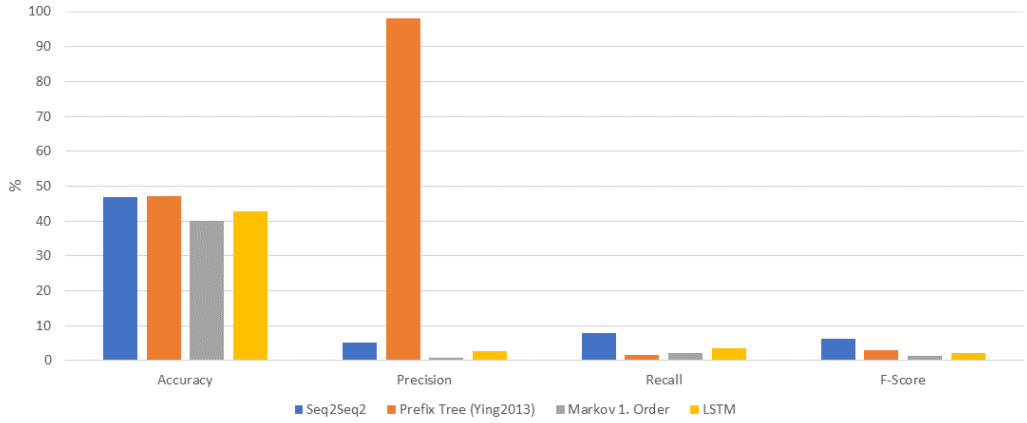

Figure 8.6: Multi-user model prediction performance (SDP dataset).

larly poor performance of the ANN-based models can be mainly attributed to the small size of the SDP dataset, since neural networks are highly dependent on the size of the available training data. We expected that the probabilistic Markov model would lead to higher results based on the findings in the previous chapters of this work that showed a certain dataset size robustness. However, surprisingly, the best Markov model of 1. order showed the poorest performance.

All in all, we can say that the Seq2Seq approach provides a solid basis for future investigations. It could be shown that Seq2Seq learning outperforms both the probabilistic and the tree-based baseline. We also saw that it may lead to similar or higher (in the single-user case) prediction accuracy scores as the ones of a standard LSTM with a lower computational effort. Especially in the single-user modelling case, Seq2Seq showed that with the right data, it can lead to a higher performance in terms of accuracy, precision, recall and F-score. This highlights once again the importance of the quantity and the quality of the training dataset and how this may significantly affect the models' outcome. However, we could also identify some limitations, especially in the long-term prediction scenario.

8.6 Conclusion

In this chapter, we investigate whether and how well a *Sequence to Sequence (Seq2Seq)* learning approach may be used for modelling semantic trajectories

and predicting the next semantic location. For this purpose, we designed and implemented an attention- and LSTM-based Seq2Seq location prediction model. We compared a multi-user as well as a single-user version of our approach on two different real-world semantically enriched datasets. A standard LSTM neural network, a tree-based framework and a probabilistic Markov Chain model of 1. and higher order served as our baselines. We could show that the Seq2Seq model generally outperforms both the tree-based and the probabilistic Markov approach. However, in the multi-user case, we saw no significant benefits compared to the LSTM model. Solely the slightly higher computational effort by the 128-unit LSTM model in contrast to the 8-unit equally good Seq2Seq model could be here mentioned. In the single-user modelling case things seemed to be changing for the better for the Seq2Seq learning approach, since it led to the best top results outperforming the rest of the models. In general, we could identify a higher degree of sensitivity to the quantity and the quality of the available training dataset on the part of the Seq2Seq method compared to the standard LSTM model, a fact that could be in part attributed to its training-on-whole-trajectories learning process. All in all, we believe that Seq2Seq learning is able to compete other semantic trajectory modelling techniques and this chapter lays a solid basis for further investigations.

CHAPTER 9

APPLYING ARTIFICIAL NEURAL NETWORKS ON SEMANTIC TRAJECTORIES: ENHANCING THE AFFECTIVE SENSITIVITY

Abstract

In addition to context awareness and proactive behaviour, personalization constitutes one of the most important factors to consider when building digital assistance systems. Personalization enables providers to offer services that meet the user's need in a tailored and flexible way. In the field of location prediction, personal information can lead to a significant improvement of the predictive performance of the respective models. However, most of the existing approaches handle this type of information separate from the actual location and movement data. This results in static user models that are not capable of taking the dynamics that lies underneath each situation when visiting a certain place into account. This chapter describes a dynamic ontological construct, which we call *Context-Specific Cognitive Frame (CSCF)*, in order to capture the entire experience of a user at a given moment. In particular, we extend the Purpose-of-Visit-dependent Frame (PoVDF) concept of Chapter 4 to associate situations when visiting a certain location with both the respective context information and the emotions as well as the personality of the user. We show that our method can be used to provide a flexible, more accurate and therefore more personalized user experience using the example of semantic location prediction.

The contents of this chapter is based on two publications, at KESW 2017 [KSB18a] and at Ubicomm 2017 [KB17], and on an article in the Journal of User Modeling ans User-Adapted Interaction, Special Issue Harnessing Personal Tracking Data for Personalization and Sense-Making (to come 2019).

9.1 Introduction

Much work has been done over the past years on modelling and predicting human behaviour, both by psychologists, as well as by computer scientists, like in [PL99], aiming at implementing personalized and more accurate software solutions. There is strong evidence that human behaviour prediction can be improved when psychological features, like the personality and the emotional state of a user, are taken additionally into consideration. This hypothesis is supported by a great variety of works that discuss the degree of interrelation between these kind of features and the situation in which a user finds himself as well as his social behaviour [Kas04, Fle01, Bus77, Jac74, Bor93]. Most of them focus on how and to what extent psychological features affect a human's general social behaviour. Some, such as Adali et al's. and Staiano et al.'s work, go the other way and attempt to infer these features from the behaviour of the user and his social network structure [AG14, SLA+12]. Recently, Bollen et al. were able to predict in [BMZ11] whether the Dow Jones index will close higher or lower than it started that day by analyzing the mood of large-scale Twitter feeds with standard tools, reaching an accuracy of 87.6%. Song et al. [SK18] show via Poisson regression that there is a strong relationship between certain personality factors and the location selection of humans. Their result matches with the outcome of Rentfrow et al., in which personalities are distributed unevenly, building personality clusters across the U.S.A. [RGJ+13]. While personality could apparently be used to establish or interpret time-invariant relations between users and locations, many human movements are of impulsive nature and they can be traced back to the users' mood and their overall emotional and mental state. This becomes very clear if we consider how often we have brief stopovers to buy a snack in case of sudden hunger or cancel at the last minute party invitations because we are tired and we don't feel like it. Karatzoglou et al. show in [KSB18b] that using certain mental and emotional states of the users together with information about their personality profile contributes significantly to a more accurate model of the human movement behaviour.

Thus, we believe that both the personality and the emotional state of users have a significant impact on the way they move from place to place in their ev-

eryday life. Therefore, it seems more than reasonable to consider these features, among other context information, when trying to model human movement patterns and predict future locations. For this reason, we built an ontology-based knowledge base, in order to aggregate the available context information into an object, which we refer to as *Context-Specific Cognitive Frame (CSCFs)*, or in our case ***Location**-Specific Cognitive Frame (LCSF)*, which in turn is used as input for our location prediction model, a Long Short-Term Memory (LSTM) neural network that proved to perform best in the previous chapters. Each LSCF instance represents an extension of the PoVDF approach described in Chapter 4 and aims at encapsulating the overall experience of a user at a certain location. It contains the semantic location type, her activity and the overall purpose of visit, the corresponding time, her personality, her current emotional and mental state, as well as information about whether she is alone and if not with whom (companion). In this chapter, we investigate the impact of modelling all this additional context information using LSCFs on the performance of the next semantic location prediction. We refer to our approach generally as *Sentient User Modelling (SUM)*.

In addition to evaluating the LSCF-based approach and motivated by the results in Chapter 4, this chapter explores the idea of feeding our LSTM network during the training, apart from the current input location, with semantically similar locations as well (e.g., *cinema* and *theatre*, or *fitness studio* and *boulder hall*). This could allow the network to learn the interrelations between locations and the respective transitions from one location to another much faster. Thus, by doing this, we are hoping to be able to speed up the training process and make it more efficient. We call this approach in this chapter *combined learning*.

There exists a number of datasets, which include beside semantic location, other context information as well, such as the activity of the user and biometric information[1]. However, we couldn't find a dataset that matches all of our criteria including the recording of the emotional state and the personality traits of the users. For this reason, we conducted a user study to create our own training and evaluation dataset.

[1] `http://ntcir-lifelog.computing.dcu.ie/` and `https://www.imageclef.org/2018/lifelog`

The rest of this chapter is structured as follows. In Section 9.2, we give a short overview of the existing techniques in the field of user modelling and reiterate some of the most relevant semantic location prediction approaches. Due to the interdisciplinary nature of our work, Section 9.3 goes through the basic concepts of each of our components: ontology, personality and emotional state models. Section 9.4 discusses in detail our user study design as well as the architecture of our approach. Our evaluation methodology and the respective results can be found in Section 9.5. Finally, in Section 9.6 we summarize our conclusions and discuss our future work.

9.2 Related Work

The previous chapters of this work discuss various models and architectures as possible solutions for modelling semantic trajectories. Most of the models handle solely semantic locations as their input, except of the approach in Chapter 4, which takes time, the purpose of visit and the activity of the users additionally into consideration. Samaan et al. in [SK05b] attempted as one of the first to add more context information about each user in order to improve the prediction in an indoor semantic localization scenario. In particular, they use information about their users' interests and schedule, among others, combined with evidential reasoning in order to make a prediction regarding to where a user will go next. Moreover, they apply their algorithm on so called spatial conceptual maps to determine the path taken by the user to reach her desired destination. Ridhawi et al. [ARAKA09] propose a similar location- and context-aware system architecture to predict the user's future location, where additional context information is used again to improve the accuracy. However, in their work, the additional context information is stored in an OWL-based ontology model [WMS04]. The context information includes the user's profile, context (such as comparative locations identified by similar floor numbers and building addresses) and location history. Interestingly, they explicitly identify and differentiate between undergraduate students who have a predictable tight schedule and graduate students who have more degrees of freedom and are more difficult to predict.

The aforementioned approaches have one thing in common, namely that

they do not consider psychological attributes. As already stated in our introductory section, there is strong evidence regarding psychological features and their role in location prediction. Kim et al. use the personality of users to predict their next location [KS14, KKS16, KKS17]. They conducted a survey over a period of 6 months to collect location and personality data using the Big Five personality trait model [Gol93]. After clustering the locations, they came up with three (Home, School, Other), respectively four [KSK12] (Home, School, Other, Mountain) semantic location categories. A Feed Forward Neural Network (FFNN) is applied upon the clustered locations in order to provide predictions about the users' next location. Their model takes the users' personalities and the current time as input, without laying much value on the current location or location history. In addition, Kim et al. analyze the general relationship between personality and locations using regression. Their research points out that personality traits have a different influence depending on the time of the day. For instance openness has its peak at the noon, while neuroticism at midnight. A more sophisticated approach is introduced by Kim et al. in [KS17], where a Deep Neural Network (DNN) and a Deep Belief Network (DBN) are used on the one hand for location prediction and on the other hand to analyse the relationship between personality and locations in general. Similar to their previous work, only three clustered semantic locations are considered for the in-depth analysis and the personality together with the temporal information is the only input to the DNN.

Chauhan et al. [CTT16, CKT17] chose a slightly different way and use Tweets as a data source for predicting locations within a given geospatial range (e.g., 300m, 1000m, 2000m). In particular, they attempt to extract the necessary information about the users' location and their personality, among others, by crawling and processing the content of their Tweets using the Latent Dirichlet Allocation (LDA) method [BNJ03]. Their approach reaches an accuracy of approximately 60%.

As we saw earlier, many researchers rely on additional knowledge about the users and their context aiming at improving their predictive performance. While some let this knowledge flow indirectly into their location prediction models, others make use of structured data or knowledge bases, such as ontologies, to store and manage it. This is especially beneficial in case one is

interested in building extensive and large-scale *user models*, where managing a vast amount of information in an efficient manner is of crucial importance. Usually, a user model includes personal information, such as demographic data (name, age, etc.), the users' interest and preferences, and their goals among others. Ideally, a user model may reflects the point of view of the users towards the world and how it is being perceived by them. However, depending on the application, not every information is of relevance. The resulting user models can be either static or dynamic. The existence of a user model provides the basis for providing timely and user-tailored services. In addition, it can contribute to the transparency and understandability of applications on the part of the users [CK94]. Kay depicted in [Kay94] as one of the first the importance of giving the users an understanding of their own models, introducing a cooperative user modelling toolkit at the same time. In her work, it is shown that the interaction with their user model and the respective viewing tools is both helpful for the users and the machine, as it leads to more accurate models.

There are many different ways of building user models. Pentland et al. model humans and their behaviour as a multi-state machines with a finite number of internal mental states and well-defined interstate transition probabilities [PL99]. In particular, they use dynamic Markov models to model human driving behaviour and to provide short-term predictions regarding the drivers' subsequent actions with a promising accuracy performance.

Heckmann et al. [HSB+05] introduced GUMO, a General User Model Ontology. The idea behind GUMO is to provide a common ontology to represent and store the users of semantic web applications and thus to facilitate user-specific data exchange between such applications. It represents one of the most extensive and well-designed user model ontologies. However, while for example a user in GUMO can be *interested* in things, GUMO does not try to list all *things* that exist in the world, it leaves this task to other ontologies. This is an example of its modularization and customizability. GUMO is capable of storing the emotional state, the personality according to different models such as Meyers-Briggs, Big Five or the Three Factor PEN model and the mental state of the user. Hainš et al. explore in their work [HLK07] the use and application of ontologies for representing specific multi-dimensional personality models as well, such as the Costa and the McCrae Big Five factor model.

Similar to Heckmann et al., they developed their ontology using the Protégé editor. Our ontology design in this chapter, including certain elements, such as the available emotions, are partly based on the work of Heckmann and Hainš et al.

As already mentioned, in this chapter, we extend the work in Chapter 4 by adding emotions and personality to the PoVDF objects and creating so called Location-Specific Cognitive Frames (LSCFs). In a next step, we feed the resulting LSCFs into a LSTM neural network in order to provide estimations about the future locations of the users. We chose LSTM to be our model, since it has been shown to perform best in comparison to other network architectures. In the following section, we provide a detailed insight into the theory behind the components of our approach.

9.3 Basic Concepts and Preliminaries

Our system consists of several modules, which in turn provide a number of interdisciplinary functionalities. This section describes first our custom ontology, which on the one hand, we use to semantically classify locations, and on the other hand, to store additional user and context data (e.g., activity, personality and mental state among others). An important feature of our predictor is the personalization through the use of psychological features, such as certain personality traits and emotional states of the users. Therefore, we present briefly some of the models that have been proved successful in the psychology domain. Lastly, we reiterate briefly the theory behind the Long-Short Term Memory (LSTM) networks, which we use for the actual prediction and were thoroughly discussed in Chapter 6.

9.3.1 Ontologies

Ontologies represent formal representations of domain knowledge in the form of concepts and things as well as their relation to each other. In some aspects they show similarities to object-oriented programming, e.g., there are classes and instances, there is inheritance and relations between classes, but there are also some key differences: Firstly, an instance is not created from a specific class, rather an instance can belong to many classes at once. Classes can be defined

as disjoint in order to prevent nonsensical model states. When building an ontology, either the *open-world* assumption, or the *closed-world* assumption must hold (most of the time the open-world assumption). These state that anything that has not been explicitly defined is by default true (*open*) or false (*closed*). The creator of an ontology defines a set of classes (entities), which can be subclasses of each other (*is-a* relationship), and a set of additional relations between them. Once a certain domain has been sufficiently defined, one can create instances (individuals) and assign certain properties to them, e.g., which classes they are members of and what relations they have with other individuals. Finally, *reasoners* can perform logical reasoning over the model, deducing new memberships or relations and checking the model for consistency.

For the approach presented in this chapter, we created an Ontology-based knowledge base in Protege [Pro] based on OWL [WMS04] to store all the relevant information we need for our prediction model: semantic location (type), time, high-level purpose of visit, activity, companionship, personality, emotional state and time. We oriented ourselves on the Foursquare venue taxonomy [Fou], as in Chapter 4, and the GUMO ontology [HSB+05] in order to build the semantic location and user modelling part in our ontology respectively. Fig. 9.1 shows a small sample of our ontology.

As already mentioned, in our approach we attempt to encapsulate the emotional state of the user together with the respective context information and his personality when visiting a certain location at a certain time (e.g., *Mary, an outgoing person, sits on a bench at the park (location) with her boyfriend (companion), reading a book (activity), feeling happy (emotional state)*). In tangible terms, we want to link each visit at a certain location to each and every available information stored in our ontology, which reflects so called *n-ary relations*. However, OWL allows only binary relations to be defined. For this purpose and similar to Chapter 4, we use an extra object in order to aggregate all the relevant information, which we call *Location-Specific Cognitive Frame* (LSCF). We refer to the particular LSCF-based user modelling approach as *Sentient User Modelling (SUM)* and to the corresponding location prediction approach as *Sentient Destination Prediction (SDP)*. Fig. 9.2 provides a simplified illustration of the LSCF concept.

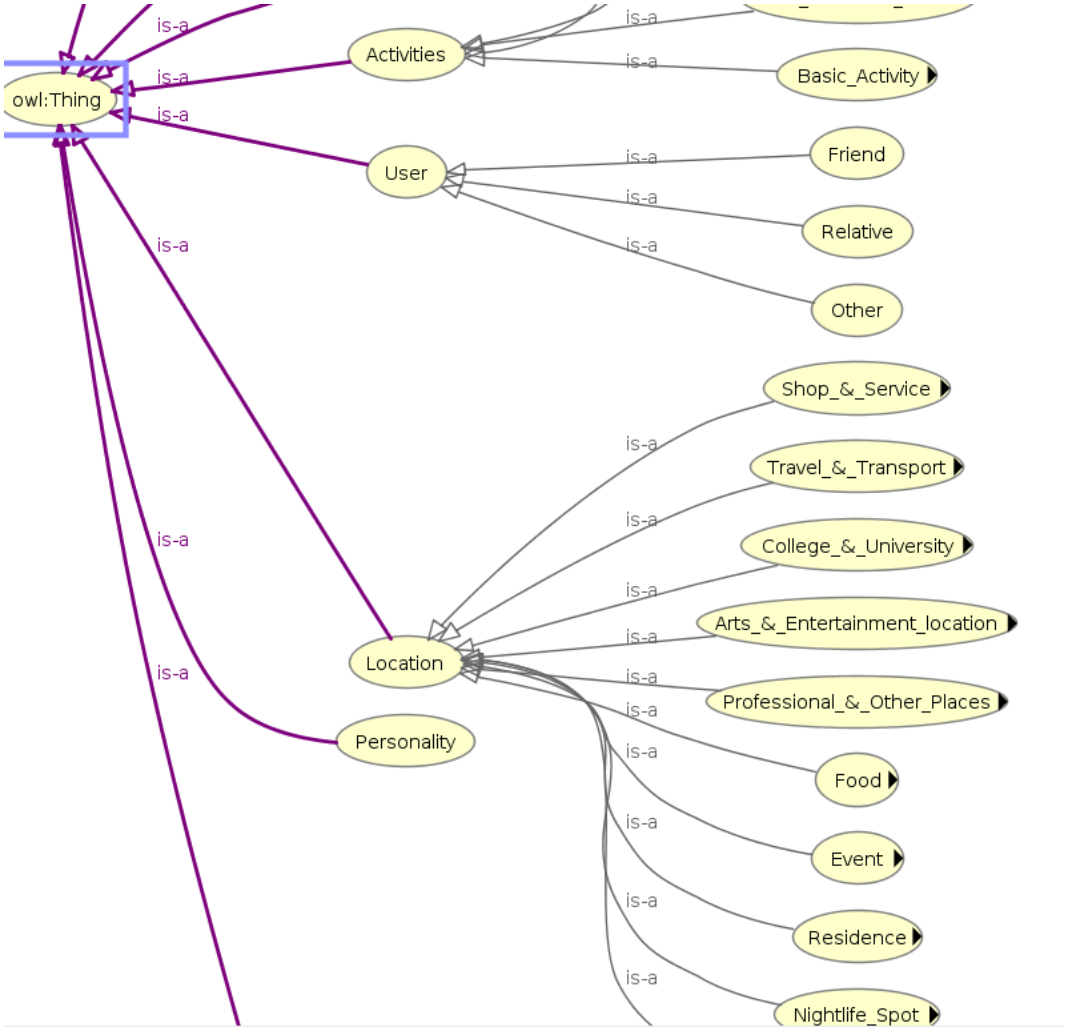

Figure 9.1: A snippet from our ontology used to store the LSCFs

The concept of defining Location-Based Cognitive Frames is introduced in this chapter as a means of dynamic modelling of users and their movement behaviour. We adapt in a certain way the initial *Frame System Theory* of Minsky in the 70's [Min75, Min74] in order to encapsulate the (overall) experience of users when visiting a certain location. In this way, correlated context information (e.g., *happy*, *with her boyfriend*, *reading a book* and *at the park*) can be aggregated into single object instances, which in turn can be further processed and fed into some context prediction model like in our case. Finally, the resulting LSCF instances are properly encoded (see section 9.4.3) and fed into our LSTM-based prediction model.

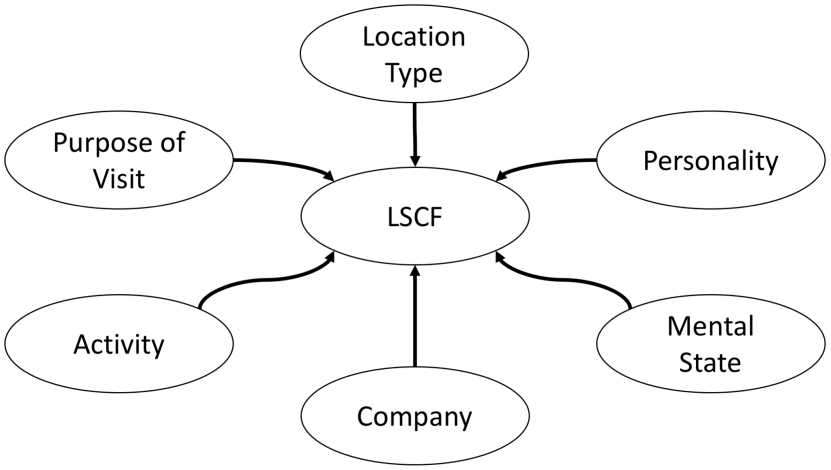

Figure 9.2: Location Specific Cognitive Frame (LSCF)

9.3.2 Personality and Emotional State Models

The most widely used and validated model for human personalities in psychology is the so called *Big Five model* [Gol93]. It represents a person's personality on a 5-dimensional scale by floating-point numbers from 1 to 5. The dimensions are *extroversion*, *agreeableness*, *openness*, *conscientiousness* and *neuroticism*. The higher the number, the stronger the corresponding trait is pronounced in the user. To measure the personality, psychologists use typically questionnaires. There exist many different questionnaires in the literature that vary in their length and comprehensiveness. For our approach, we chose the NEO-FFI-30 questionnaire [KGR$^+$08]. This relative short variant consists of 30 questions that are answered on a 5-point scale. Its reliability and validity has been empirically shown in former studies.

There also exist many different possible solutions for representing mental and emotional states. In the field of sentiment analysis, a common way to describe the mood is to use one-dimensional representations. This means that the mood of a user is represented as a number between -1 and 1, often also written as a feeling between *negative* and *positive*. We deemed this model unsuitable for our needs, as we wanted a more differentiable and multi-state model. A more detailed model is the Profile of Mood States model [MLD92]. It rates the current mood in 6 dimensions on a scale from 1 to 5. However, this requires the participants to fill out an adjective checklist comprised of 65

items, which would be very cumbersome for our study participants during the field study. There are also high-dimensional mental state representations for sentiment analysis using a continuous manifold [KLLE13]. However, similar to the aforementioned model, all these methods have one thing in common, namely that they are not suitable to capture the emotional states of a user in an everyday logging scenario, since too much time is required for the current mood to be acquired.

In order to limit the number and find the right set of mood states, we conducted an online pre-study. We asked 24 participants to describe their previous day in episodes, similar to the popular Day Reconstruction Method [KKS+04]. We recorded 213 episodes annotated with time, activity, location, companion and mood. We selected the 10 most frequent mental states for our user study participants to select from in the mobile application (see section 9.4.1).

9.4 Overview of Our Approach

In order to evaluate our approach in a real-world scenario, we designed and carried out a 8-week long user-study. In this section, first, we discuss the design of our user study and the respective decisions we made in our attempt to collect a consistent and qualitative dataset. In the second part, we explain the overall architecture of our prediction system, as well as the various variations of it together with the corresponding training methods.

9.4.1 User Study

Since we wanted our location prediction model to consider additional context information such as activity, mood and personality, we needed a semantic trajectory dataset that contained this kind of information. Unfortunately, due to the sensitive nature of this kind of data, the existing privacy laws and the difficulty to collect this information, there are very few trajectory datasets in this domain in the first place and, to our knowledge, none that fits our criteria. For this reason, we conducted a user study and collected our own dataset consisting of semantic trajectories, current activity, companionship, personalities of the users, as well as additional information about their mental

and emotional state at each location. We implemented a mobile application and had participants annotate their daily trajectories with the aforementioned additional context information.

As daily annotating is a cumbersome task, we tried several ideas to gather a maximum amount of data from our participants: Firstly, we designed the application in a way that minimized the time needed to use it to create an annotation. Each text field in the application had an auto-completion feature that stored past user input entries. We also made sure that everything was visible at a glance without having to scroll down the screen. In addition, as an extra incentive, each time the users annotated their data, they got a ticket for our lottery with a small monetary prize for the three winners. This means that users that annotated more often, had a higher chance of winning (whereby data quality was also considered as a criterion). Lastly, we offered participants to visualize their collected data at the end of the user study, e.g., in the form of maps showing visited locations and their overall trajectories as a life-log location and mood diary. This idea comes in the spirit of the recent "Quantified Self" movement that tracks everyday life in hard numbers. However, there was very little demand for the activity and mood diagrams from our participants. Solely the maps proved to be successful as incentive.

For the user interface (UI) of our application, we considered several alternatives. A screenshot of the main user interface of the application can be found in Fig. 9.3 as well as a translation of the interface in the respective caption, since the application was used in Germany. A crucial part here was the design of the section to select the mental state. We wanted the depictions to be as intuitive and neutral as possible, since these can affect the users emotions as well. The first approach were standard emoticons. The problem with these emoticons was the missing validation. Therefore we searched for scientifically validated input methods. We firstly considered the Geneva Emotion Wheel (GEW) [SSS12] consisting of a circle that has the two dimensions valence (negative to positive) and control (low to high). In the circle, different emotion families are aligned to be selected by the users. However, the GEW was not suitable for our application design because its alignment and a proper, readable size would make it impossible to fit in the mobile phone's screen together with the rest of the app's elements. Then, we considered the Self-Assessment Manikin

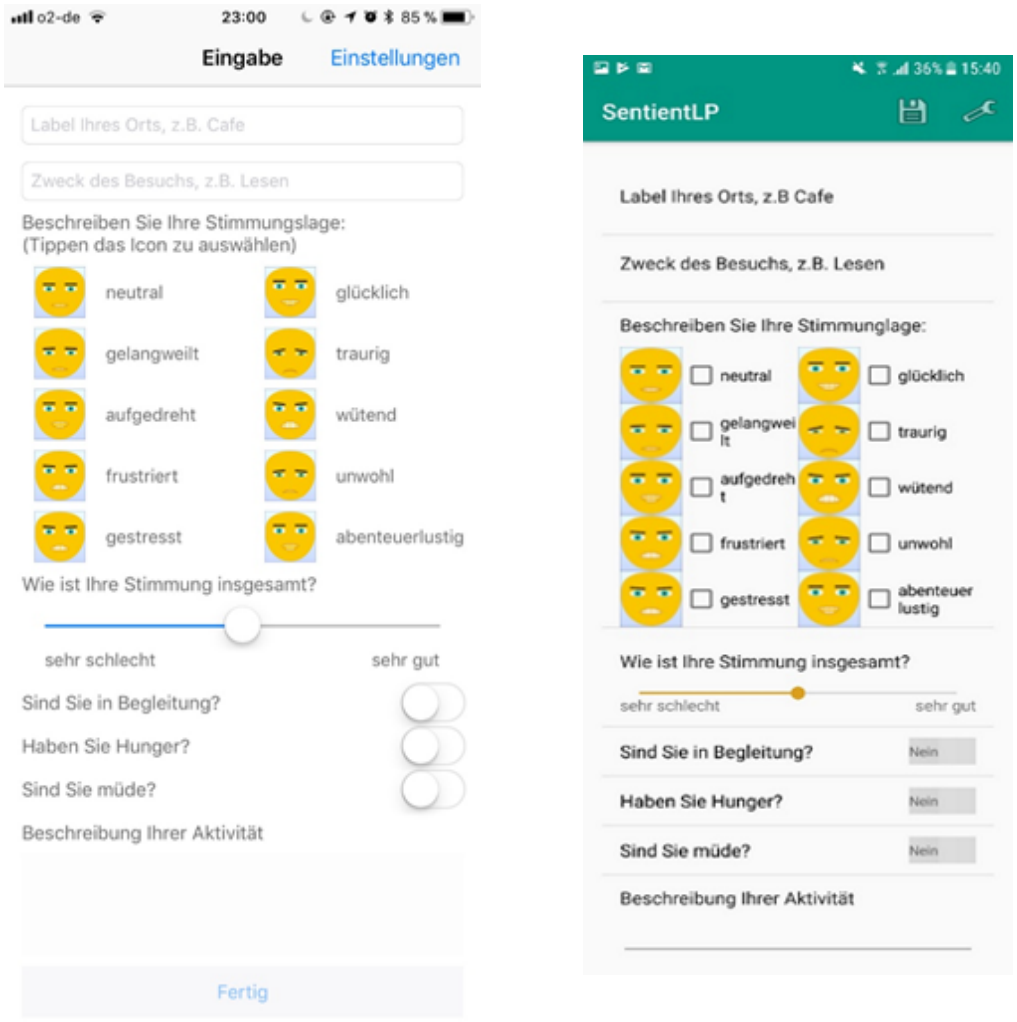

Figure 9.3: Screenshots from the iOS (left) and the Android (right) version of our app. The entered data are (from the top to the bottom): Label of the visited location, high-level purpose of visit, checkboxes for each mental state (depictions adopted from [BB09]), assessment of the mood in general (very bad - very good), do you have company?, are you hungry?, are you tired? and the description of the current activity.

(SAM) [BL94] that measures the affective dimensions of valence, arousal and dominance and depicts them in a form of three manikins. However, as shown by Siegert et al. [SBV+11] there are validation problems in terms of label reproducibility since the labellers have to be trained first to use it properly. Finally, we selected the AffectButton method [BB09]. The AffectButton has the advantage of being a validated method to enable users to give explicit

affective feedback.

In our attempt to reduce the dimensionality provided by most of the methods and come to a practical solution for our app, we tried to limit the number of possible emotions for the participants to select from. For this purpose and as already mentioned in Section 9.3.2, we conducted and evaluated a pre-study to find the most frequent annotated mental states. After designing and distributing our pre-study questionnaires, 24 participants handed in their 213 episode descriptions oft their daily life. We used the established Day Reconstruction Method (DRM) by Kahneman et al. [KKS+04] as a model to conduct this pre-study. Participants were asked to describe their day in episodes. Each episode refers to a single situation with one location and one activity. For each episode, participants annotated the time, the activity, the location, the companion (if any) and their mood. Afterwards we evaluated the pre-study and selected the 10 most frequently annotated mental states to be part of the app. We mapped the selected states to the AffectButton and inserted it into the application.

The participants used our application over a period of eight weeks and annotated their daily trajectories. They were instructed to use the application to store any location at which they spend more than five minutes. The application then automatically stored the user's current GPS location, the entered information and some additional information, such as the mode of transport as detected by the operating system and the software version among others. This information was later sent to our server as an encrypted message, where it was decrypted and stored. To identify the different participants, each installation was provided with a random App-ID, which was stored together with the data as well. Our participants also filled out a personality questionnaire from which we determined their Big Five personality scores [KGR+08]. To ensure anonymity, we pseudonymized all data using the App-ID, which we also used to correlate the personality results with the data from the application. Our application followed a classical client-server model, with the application installed on the users' phones being the clients and a server serving as a central storage node.

At the end of our study, we preprocessed the collected data by removing inconsistent and poor annotators whose location data weren't enough to obtain

meaningful results. Then, we propagated the filtered information into our ontology. Finally, we built the respective LSCF instances, which were used later as input for our LSTM network.

9.4.2 Architecture

In this section, we describe the architecture of our approach. Our semantic location prediction model consists of two main parts (see Fig. 9.4):

- *Ontology* The ontology is used to store and categorize the available context information. This includes the location semantic labels, purpose of visit, activity, mental state, personality and time. At the same time it serves as basis for aggregating the aforementioned information into LSCF objects.

- *LSTM* The LSTM neural network is used to learn the temporal relationship between succeeding LSCFs and provide predictions about future locations. Our LSTM network is a standard 1-hidden layer network with at most 128 neurons on it. Both the number of the layers and their neurons represent training hyper parameters and their optimal value was determined via conducting a grid search as we shall see later in this work. A deeper architecture (more than one hidden layers) did not bring any benefits and showed tendencies towards overfitting issues due to the small size of our dataset.

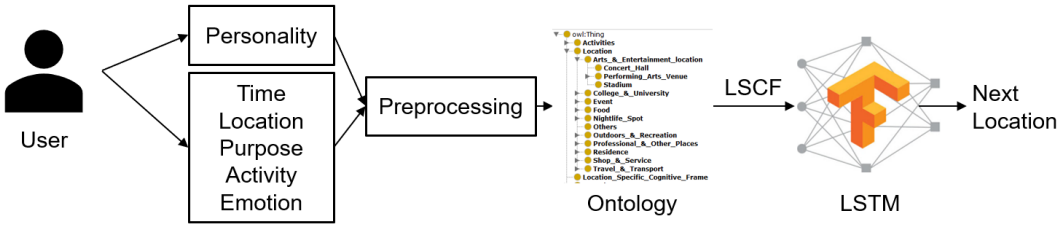

Figure 9.4: Sentient Destination Prediction architecture.

Our main contribution lies in the usage of the dynamic, comprehensive and highly personalized context information stored in each LSCF instance in our attempt to improve the modelling performance of the LSTM predictor in terms of accuracy.

In addition to the general usage of LSCFs, in this chapter we also explore the impact of feeding the LSTM with the most (semantically) similar location as an additional input (e.g., *cinema* and *theatre*). We hope in this way to raise the accuracy and reduce the time needed to train the LSTM, since usually, similar locations (or similar experiences in our case) are followed by similar locations as well. For instance, we assume that a person that usually goes to a restaurant after watching a movie with a friend at a cinema, will most likely visit a restaurant as well after watching a play at a theatre with the same friend. The idea behind this method is to let the LSTM to learn shortcuts with regard to the interrelations between locations and their transitions based on this additional semantic information in order for the latter to require a shorter training time for making an evenly good or better prediction as before. At the same time, by looking it from a different angle, this kind of *combined learning* may also bring certain benefits in cases where the size of the available training dataset is small (as in our case), since it leads to a faster learning of relations between locations. We evaluate this idea in Section 9.5.

9.4.3 Prediction Model Variations

We used three different versions to train our LSTM prediction model depending on the nature of the input training data:

1. *(Solely) Semantic Location Labels.* This version was used as a baseline to compare it to the other approaches, which use additional context information. We trained the LSTM on all six semantic levels in our location taxonomy of our ontology. The labels were one-hot encoded.

2. *LSCFs.* In this version, complete LSCFs were fed into the LSTM. Each partial context information (activity, emotions, etc.) in the LSCF was separately one-hot encoded and concatenated into a single input vector representing each LSCF.

3. *Personality & Emotions.* Here, we neglect the purpose of visit and the activity information, and we feed the LSTM only with the semantic location labels and the psychological features, namely the personality and the emotional state information.

4. *Combined Learning.* With *combined learning*, we aim at analyzing the effect of additional context information fed into the network in terms of similar locations as discussed at the end of the previous subsection. We do this by feeding the network both with the current location and the most similar location found in the training dataset (other than itself) at the same time. To calculate the similarity we explored several topological similarity scores on our ontology, which we describe below.

Similarity Scores

For calculating the location similarities, we consider the three following options from [LSSM08] and [MHG13]:

- *Shortest Path similarity*, which is simply the length of the shortest path between the locations in the ontology.

$$Sim_{path}(l_1, l_2) = 2 * deep_{max} - len(l_1, l_2) \tag{9.1}$$

  In this formula, $deep_{max}$ is the maximum depth of any node in the ontology and $len(l_1, l_2)$ is the length of the shortest path between the two locations l_1, l_2.

- *Wu & Palmer similarity*, which considers the depth of the lowest common subsumer, the "parent" of both locations, as well. If this is a low-level, that is, a very specific node, the two locations are highly similar.

$$Sim_{WP}(l_1, l_2) = \frac{2 * depth(lcs(l_1, l_2))}{len(l_1, l_2) + 2 * depth(lcs(l_1, l_2))} \tag{9.2}$$

  Here, $depth(v)$ is the depth of node v in the ontology, when viewing the class hierarchy as a simple tree, with *Thing* having a depth of 1. $lcs(u, v)$ is the *lowest common subsumer* of the nodes u and v, which is the node in the ontology that has both u and v as children with the highest depth.

- *Lin similarity*, which additionally considers the information content of each location, which in turn depends on the number of occurrences of

this location in the dataset. According to the information theory, rare locations have high information content.

$$Sim_{lin}(l_1, l_2) = \frac{2 * IC(lcs(l_1, l_2)))}{IC(l_1) + IC(l_2)} \tag{9.3}$$

$IC(v)$ denotes the information content of a node v: This is calculated by counting the occurences o_v of v and all children of v in the dataset. Then $IC(v) = \log(\frac{o_v}{n})$, with n denoting the size of the dataset.

In addition to the above standard similarity metrics, we also investigate a LSCF-based similarity analysis approach, similar to the one we applied in Chapter 4. Based on this approach, two locations are similar when the corresponding LSCFs, and thus the corresponding user experiences at those locations, are also similar. To calculate the most similar location to a given LSCF we define an assembled similarity score, based on the weighted average over all similarity scores over the components of the LSCF:

- For emotions and the purpose of visit (including the companion information) we use the Jaccard-Index as described in [LW71] and [Tve77]. For two items A and B, the similarity is given by:

$$sim_{Jaccard} = \frac{|A \cap B|}{|A| + |B|}, \tag{9.4}$$

with $|A|$ and $|B|$ describing the set of their attributes. If both sets are empty, we define the similarity as 1.0.

- For the personalities in the Big Five model, we calculate the sum of absolute differences in each dimension and normalize it to a number between 0.0 (polar opposites) and 1.0 (identical).

- For the semantic locations, we considered the three approaches from above and applied them on our location taxonomy. In our evaluation, the Lin similarity performed better than the other two similarity metrics and thus we used it in our LSCF similarity.

Single- and Multi-User Models

We made an additional distinction between a multi- and the corresponding single-user models. In the case of the single-user model, a separate LSTM network is trained for each user. In this case, the personality feature, as non-changing information for each user, was not included in the training of the individual network. As we already saw, having user-specific models can theoretically lead to better results. In practice however, it requires the existence of a large amount of data collected from the particular single user for such a single-user model to be feasible, which is usually not the case here. Despite these concerns, we investigated the usefulness of this approach.

In contrast to the single-user models, the multi-user model includes the additional context information and annotations of all participants. The single location trajectories are concatenated into one long trajectory sorted by user and time. This has the advantage of having a maximum amount of data that can be used for training, but represents an unpersonalized, one-size-fits-all solution.

Semantic Representation Level

The nature of the semantic location prediction task is similar to a multi-class classification problem. Our dataset comprised around 70 unique location labels. At the same time, our ontology models 6 different semantic levels (e.g., *Thing* $\rightarrow$ *Location* $\rightarrow$ *Food location* $\rightarrow$ *Restaurant* $\rightarrow$ *Fast food restaurant* $\rightarrow$ ...). Since the number of unique labels on level 4-6 ranges from 50 to 60, we further analyzed the performance on the third semantic level. To transform the dataset to a higher semantic level, we oriented on the Foursquare venue taxonomy and clustered all labels from the semantic levels 4-6 to their unique parent label in semantic level 3, resulting in around 10 unique labels. If a label was initially mapped to level 1-3 (for example due to a bad labelling by the users), we left it as it is. Fig. 9.5 supports this method. As can be seen, only a small fraction of LSCFs occur more than once, that is, in our data, similar experiences when visiting a location don't seem to have been occurred very often. This prevents the neural network from learning meaningful correlations at the lower semantic representation levels.

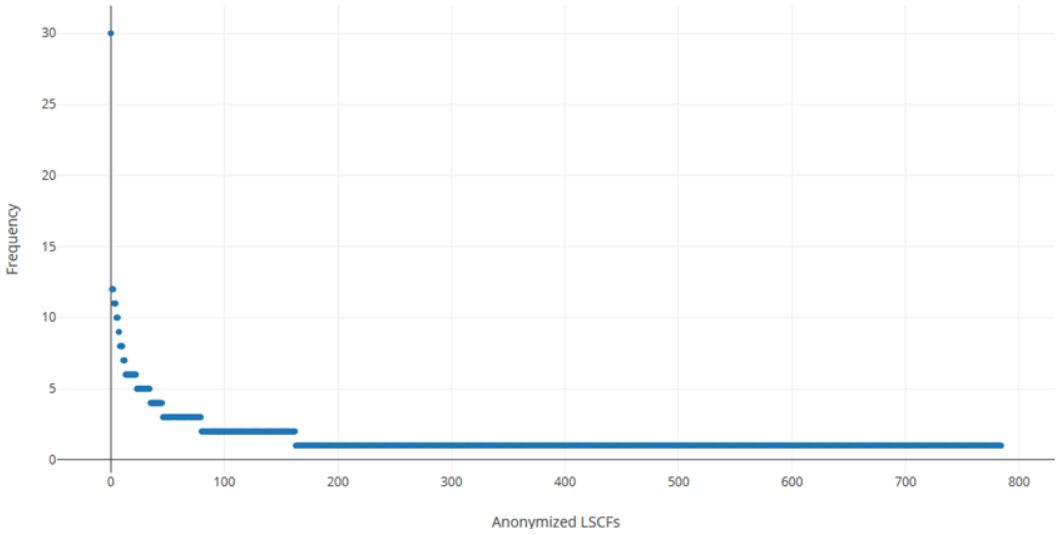

Figure 9.5: Number of occurences of each unique LSCF.

LSTM Parameters Search

To find the optimal parameters for the LSTM we conducted a grid search by iterating over all parameter combinations. The considered parameters included a batch size of (20, 24, 48), number of hidden units (32, 64, 128), input sequence length (number of locations fed into the network in a row) (1 and 2), learning rate (0.001 and 0.01), and number of epochs (1-10, 20, 50).

9.5 Evaluation and Results

In this section, we discuss the results of our user study (Section 9.5.1) and our LSCF-based prediction approach (Section 9.5.2). In section 9.5.3 we show the results of our *combined learning* approach based on similar locations, in which we feed the LSTM network with locations accompanied by the most similar location based on several similarity measures.

In the box plots, the boxes' bottom and top line represent the first and third quartile respectively and the line within the boxes is the median. The whiskers are determined in terms of the inter quartile range ($IQR = Q3 - Q1$). The upper whisker extends to the last datum less than $Q3 + 1.5 * IQR$, the lower whisker extends to the first datum greater than $Q1 - 1.5 * IQR$. Finally, the single points represent outliers. For the evaluation of our results we used

the weighted F-Score implemented in the scikit-learn module [PVG+11]. The weighted F-Score is used for evaluating multi-class classification tasks, which fits the semantic location prediction scenario. The F-Score is first calculated for each label. Afterwards, the average is weighted by support (number of samples/class) to account for class imbalance (see Section 2.3).

As described in Section 9.4.3 we concatenated all trajectories by user and time to train the multi-user model. We applied a 10-fold cross-validation, in which the trajectory was divided into 10 equally sized test data parts and was evaluated separately on each of it. The results displayed in Fig. 9.7, 9.8, 9.11 and 9.12 show the average evaluation findings over these 10 iterations. A spatial 1. order Markov Chain model trained on the semantic locations was used as an additional baseline.

9.5.1 User Study Results

All in all, 21 people participated in our user study. However, after preprocessing the collected data and filtering out all inconsistencies, we end up with the data of 13 participants, who annotated approximately 1200 data points and filled out the whole personality questionnaire. The full Foursquare venue taxonomy contains almost 900 location types. Around 70 of those can be found in our dataset. There were 53 different high-level purposes of visit and 30 activities. The frequency of the recorded emotions were happy (559), hungry (472), neutral (429), sleepy (302), energetic (81), frustrated (63), stressed (62), bored (60), adventurous (47), ill (23), sad (11), angry (9), shocked (0), companion (652).

The frequency of annotations varied heavily by participant. There was one very motivated participant annotating 419 LSCFs, 3 between 100 and 150 and the remaining participants less than 100 LSCFs during the period of 8 weeks (see Fig. 9.6). As already mentioned above, about three-quarters of the recorded visiting experiences captured in LSCFs were unique, that is, the respective experiences were entered only once (see Fig. 9.5). An *experience* refers here to a single LSCF instance, which in turn refers to a certain combination or co-occurrence of information, e.g., (location type: *cinema*, time: *evening*, emotion: *happy*, companion: *girlfriend*).

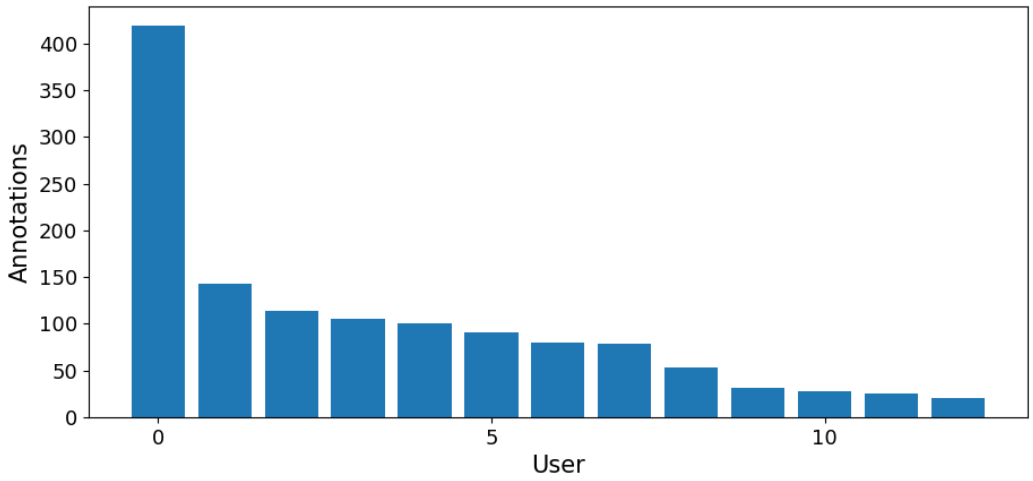

Figure 9.6: Total amount of annotations per user

9.5.2 Feeding LSCFs into the LSTM

As already stated in Section 9.4.3, we trained the LSTM network with several configurations at different semantic levels. Our findings with respect to F-Score for the deepest, that is, the lowest semantic location level in our ontology comprising around 70 unique location labels are shown in Fig. 9.7, while Table 9.1 lists the optimal parameters for each configuration. "Locations" refers to a LSTM trained solely on semantic locations as the one we had in Chapter 6, "PersonalityEmotions" refers to a LSTM trained on locations, personality and emotional state, and finally "LSCF" refers to a LSTM trained on the complete LSCFs, which contain locations, personality, emotional state, activity, companionship, purpose of visit and time. "Sim_Lin", "Sim_Wu&Pa", "Sim_Short_Path" and "Sim_LSCF" represent the models that apply *combined learning* based on Lin's, Wu and Palmer's, shortest path and LSCF similarity analysis metric respectively. Finally, "MM" refers to a 1. order spatial Markov Chain model. The highest F-Score score on the test data were achieved after 5 to 7 epochs. After that point, the network memorized the trajectories and started overfitting, degrading the prediction quality on the test data. In contrast to Ruder [Rud16], the learning rate showed no significant effect on the results.

In this section, we concentrate on the 3 box plots on the left. It can be seen that the LSCF-based approach performs better that the other two mod-

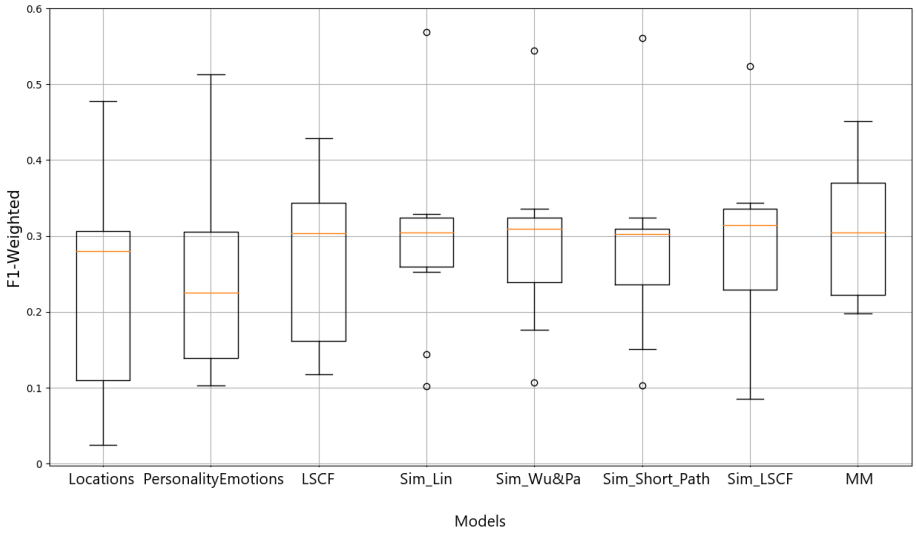

Figure 9.7: The F-Scores of the various tested models at the lower semantic representation level, i.e., the original labels from the dataset (after mapping them to our ontology)(70 semantic locations).

	Epochs	Batchsize	Learning Rate	Hidden Units	Seq-Length
Location	7	20	0.001	64	1
Personality & Emotions	5	20	0.01	128	1
LSCF	7	20	0.001	128	2

Table 9.1: The best configuration for each approach at the lower semantic representation level with 70 location labels.

els indicating in this way the added value of making use of comprehensive user context knowledge. However, although the LSCF-based model shows an overall better distribution towards higher F-Scores, the median value is only slightly better than the one of the Locations-model that uses just locations as input. Both models are skewed towards the higher F-Score values 0.28 and 0.31 respectively. The model that considers personality and emotional state (beside location) shows the poorest performance. This can be mainly attributed to the frequency distribution of the latter. As already mentioned in Section 9.5.1, the most common annotated moods were *happy*, *neutral* and *hungry*. The rest of the emotional states were recorded rather rarely. Furthermore, our analysis showed that while the *hungry* state correlates to a certain degree

with a number of location transitions related to visiting some type of food location, the other two states, *tired* and *companion*, show no correlation at all with any of the location types in our ontology. Thus, not only these could not contribute to a better location prediction, but it also seems to be having a negative effect on it, at least with our dataset. If we compare the three ANN-based models with the Markov model (MM) on the right, we can see that the "simple" Markov model outperforms the competition. We have seen this before, where in Fig. 6.8 the probabilistic Markov model is again better than the LSTM in the low-level case. Similar to the situation in Section 6.4, the evaluation is again based on a relative small dataset, a fact that generally favours probabilistic approaches such as Markov Chains.

Since the feature space of all possible visiting experience combinations in the lowest semantic level is very large (compared to the high-level case) and we only have around 1200 data points in our data set, we also investigated the effect of reducing the data space by jumping into a higher semantic level as described in Section 9.4.3. The results on the higher semantic level with all in all 10 semantic location labels (compared to 70 in the low-level case) were in general significantly better than in the lower levels (see Fig. 9.8), in part, due to the fact that the prediction problem becomes much easier. On the other hand, this can be additionally attributed to the fact that frequent fundamental, high-level human movement patterns can be captured more accurately by high-level semantic trajectories than with the low-level ones. However, reducing the dimensions of the input data while keeping the same training dataset led to the side effect of an earlier occurring overfitting. Table 9.2 shows the best parameter set for each configuration. All three configurations (again on the left

	Epochs	Batchsize	Learning Rate	Hidden Units	Seq-Length
Location	4	20	0.01	128	2
Personality & Emotions	3	20	0.01	128	1
LSCF	6	20	0.01	128	1

Table 9.2: The best configuration for each approach at the higher semantic representation level with 10 location labels.

of the plot) yield similar results. All results are higher than in the low-level case shown in Fig. 9.7. Moreover, the boxes of all models are much more compact, which means that the predictions in this case are more condense, that is, more

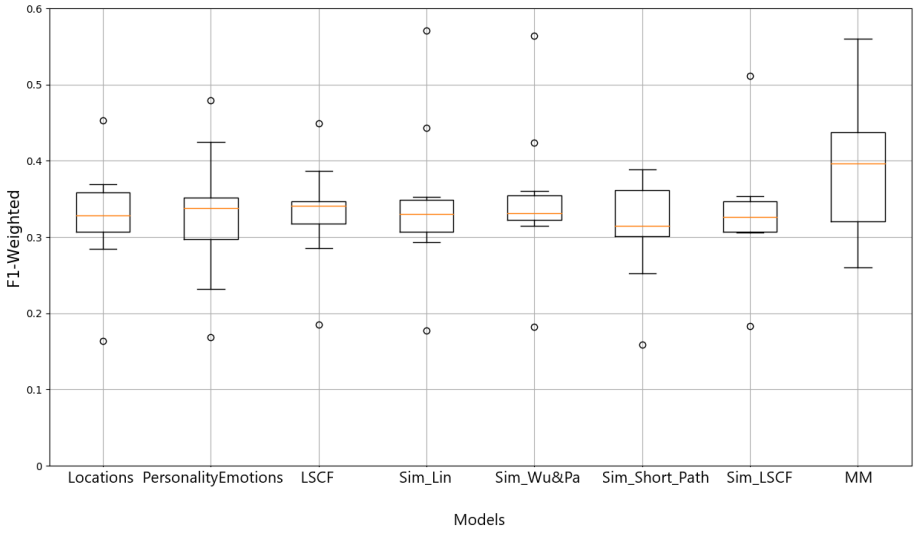

Figure 9.8: The F-Scores of the various tested models at the higher semantic representation level (10 semantic locations).

consistent. This can again be explained by the reduced number of classes, a fact that is typically associated with a reduced model uncertainty. In contrast, at the low level, the models have to fight against a more imbalanced class distribution. Once again, the LSCF-based model performs best, being both more condense and showing the highest median. The PersonalityEmotions-model seems again to be more uncertain showing the widest spread. The model that shows an even wider range of values is our baseline, the Markov model (MM). However, it also provides the highest scores, a fact that contradicts our findings in Section 6.4, where the LSTM was able to outperform the Markov model for the high-level case. As in Chapter 6.4, the better performance of the probabilistic model against the neural network approaches could be again attributed mainly to the small size of our training dataset and its imbalance.

In terms of accuracy, things don't look much different (see Fig. 9.9 and Fig. 9.10). The results match the box plots of Fig. 9.7 and Fig. 9.8 respectively. In general, the accuracy values are again much higher at the higher semantic representation level, with the simple and the LSCF-based LSTM performing better than the LSTM model that uses solely the personality and the emotions of the user. However, the Markov baseline model performs in both cases even

better.

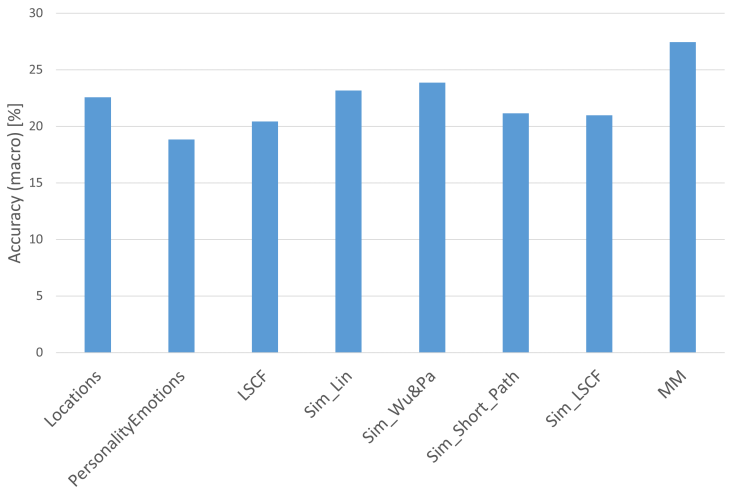

Figure 9.9: The (macro) accuracy scores of the various tested models at the lower semantic representation level, i.e., the original labels from the dataset (after mapping them to our ontology)(70 semantic locations).

It is apparent that the absolute scores of both accuracy and F-Score in this chapter are lower than the ones listed in the previous chapters of this thesis. This can be mainly attributed to the small-sized imbalanced dataset, our class-independent weighted macro averaging metrics and the fact that, in contrast to the rest of this work, we applied a "naive" shuffled 10-fold cross-validation to evaluate our models. In general, it could be observed that this 10-fold random splitting of the available dataset has led occasionally to near zero recall scores during the training process and thus to near zero F-Score values as well (see Fig. 9.11 and 9.12).

9.5.3 Combined Learning - Feeding Similar Locations Into the LSTM

The second approach that we tested consists of feeding the LSTM network with both the current location and the most (semantically) similar location, experimenting with several similarity scores. The applied similarity metrics are described in detail in Section 9.4.3. Table 9.3 and Table 9.4 lists the best configurations for each approach at the low and at the high semantic level respectively.

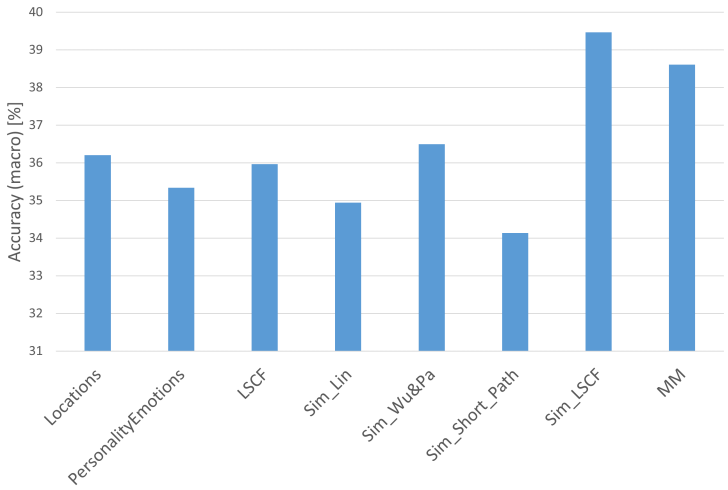

Figure 9.10: The (macro) accuracy scores of the various tested models at the higher semantic representation level (10 semantic locations).

	Epochs	Batchsize	Learning Rate	Hidden Units	Seq-Length
LSCF-Sim	10	20	0.001	128	1
Wu & Palmer	3	20	0.01	128	1
Shortest Path	3	20	0.01	64	1
Lin	4	20	0.01	128	1

Table 9.3: The best configuration for each combined learning approach at the lowest representation semantic level with 70 location labels.

The first thing that stands out at both representation levels as illustrated in Fig. 9.7 and Fig. 9.8 is that combined learning seems to provide much more robust results than the rest of the models. In both cases, both the majority of the F-Scores and the corresponding outlier values (whiskers of the boxes) of the Sim_Lin, the Sim_Wu&Pa, the Sim_Short_Path and the Sim_LSCF model scatter much less than the competition, that is, the box plots are compacter, a fact that reflects a higher precision. As before, the

	Epochs	Batchsize	Learning Rate	Hidden Units	Seq-Length
LSCF-Sim	6	20	0.01	128	1
Wu & Palmer	3	20	0.01	64	1
Shortest Path	7	20	0.01	128	1
Lin	9	20	0.01	64	1

Table 9.4: The best configuration for each combined learning approach at the higher semantic representation level with 10 location labels.

models are also far more compacter and symmetrical (from the median value point of view) when trajectories are represented at a higher semantic level. Nevertheless, apart from the aforementioned robustness at this level, they show a similar performance to the other models, remaining under the Markov Chain baseline, which interestingly shows the most varying behaviour. At the low level, the use of similar locations seems to be awarding the models with that little extra something and makes them perform much better. This can be attributed to the fact that the combined learning approach actually helps indirectly the network to cluster the low-level semantic locations into a few high-level ones. Thus, it provides the models with a similar performance as in the high-level case. An interesting fact that underpins the benefit of using semantic similarity if we consider that most of the classification tasks, especially in context-aware computing, consist of semantically similar classes as in our location prediction scenario. If we compare the four models with each other in terms of overall performance, the different similarity metrics are very similar, with the Lin, Wu-Palmer and our LSCF similarity performing slightly better than the other two approaches, as can be seen in Fig. 9.7 and Fig. 9.8, for the low and the high semantic representation level respectively.

When it comes to accuracy, it can be seen that combined learning is able to boost the predictive performance of the respective models. The Wu & Palmer based approach seems particularly promising, since it reaches higher values than the ones of the majority of the models at both the lower and the higher representation level. The Markov model constitutes again an exception. One thing that particularly stands out is the performance of the Sim_LSCF model at the higher semantic level, which provides the overall best results outperforming our baseline Markov Chain model. The fact that the combined learning models perform well could be explained by the fact that taking similar locations into account helps to a certain degree overcome feature dimensionality and sparseness issues. This reminds us the similar improved behaviour of the PoVDF-based model of Chapter 4.

9.5.4 Training Convergence

Fig. 9.11 presents how the prediction quality of each approach evolves with the growing number of epochs during the training in terms of F-Score. What

immediately draws attention is the fact that convergence discrepancies can be identified only in the left figure, that is, at the low semantic representation level. At the higher level, all three approaches converge similar fast and the differences are negligible. However, at the lower level, the rich in features models, that is, both the LSCF and the PersonalitEmotions models converge and therefore train much faster, achieving relative high values even after a single epoch. In comparison, the model that takes solely locations into account

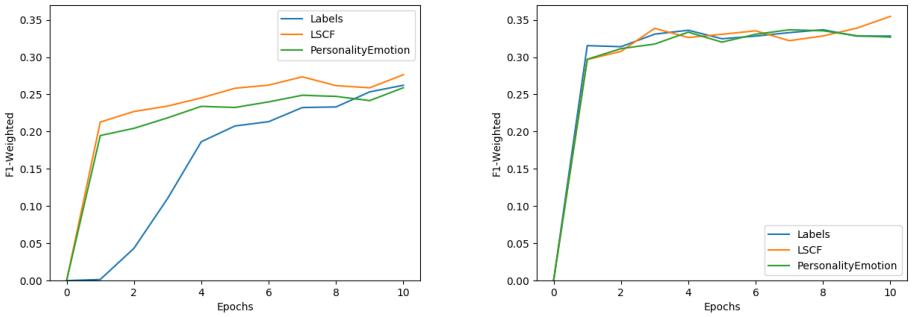

Figure 9.11: The weighted F-Scores by each method at the lower (left) and the higher semantic level (right) over the epoch number.

is much slower and reaches its best result after 7 epochs. Thus, in this case, additional context information seems to be leading in shorter training times. Moreover, the highly personal characteristics *personality* and *emotions* seem to influence the training process particularly positively, giving a good result after 4 epochs and peaking in epoch 5. The above results indicate that incorporating personal knowledge as well as additional context information such as the users' activity could be useful for cases where a large amount of data is available and a short training time is important. On the other hand, this approach can be additionally applied on cases where only few training examples are available, due to the similarity-based approach. Chapter 10 addresses this topic in detail.

Fig. 9.12 shows how fast each of the combined learning models reaches its optimum compared with the locations-only approach. It can be clearly seen that at the low level (on the left) all models are able to outscore the baseline model when it comes to giving a good result as fast as possible. However, the model with the more complex similarity metric, the LSCF similarity based model, although it is better than the baseline, it performs not so well compared

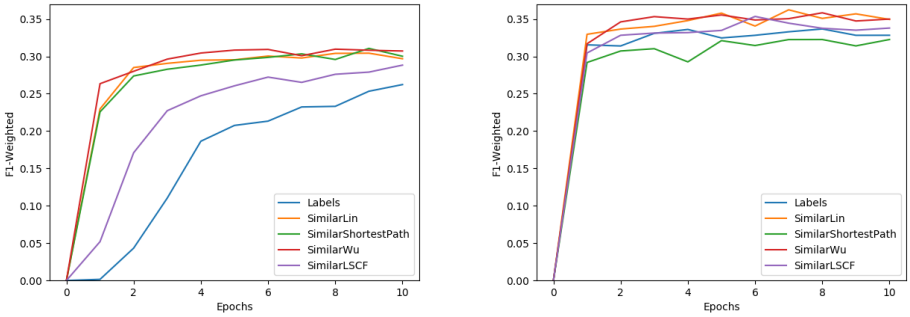

Figure 9.12: The weighted F-Scores by each similarity-based approach at the lower (left) and the higher semantic level (right) over the epoch number.

to the rest of the combined learning models. This can be attributed to the high dimensionality of the LSCF approach in combination with our small training dataset. It is much harder to find similar objects between high-dimensional LSCF instances than between instances that are characterized by a lower feature dimension. As previously, at the higher semantic representation level all models behave similar and show no significant differences.

9.5.5 Single-User Models

As already mentioned previously in this chapter, the participants of our study provided us with data of different size. Some users recorded and annotated more data than others. This section discusses the results of individual single-user models and indicates the importance of having a good dataset. Table 9.5 and 9.6 contain the results at the lower and the higher semantic representation level respectively.

User	Annotations	Location	PersonalityEmotion	LSCF	SimLin	SimWu	SimShortestPath	SimLSCF
0	80	54,42%	71,43%	77,55%	79,37%	34,29%	57,14%	54,42%
1	114	62,11%	30,26%	38,09%	53,55%	42,86%	38,10%	59,83%
2	105	43,39%	60,00%	54,81%	44,44%	36,77%	44,44%	62,96%
3	419	48,59%	46,23%	51,68%	43,61%	49,47%	45,26%	49,28%
4	143	33,88%	35,69%	37,04%	34,81%	33,33%	38,60%	34,57%

Table 9.5: Single-user model results (weighted F-Scores) for the 5 most frequently annotating users at the lower semantic representation level.

As expected, the outcomes of the individual and personalized models are

generally higher compared to the single multi-user model. However, they seem to depend significantly on the considered user, the semantic level and the model. So, although there is a general tendency towards high-level models performing better, surprisingly, the PersonalityEmotions model, as well as the LSCF- and the SimLin-approach work really well on user 0 on the lowest semantic level reaching an accuracy of up to 78%. Interestingly, in this special case, from all the combined learning models, the SimLin model is the only one that was able to lead by far to such high scores. Solely, the SimLSCF shows to a certain degree a comparably good performance for user 2.

From the figures at the high-level case, it can be seen that the LSCF approach yields in average the highest scores among the users. This provides again strong evidence that encapsulating personal characteristics, both dynamic (emotions) and static (personality), together with further context information, such as activity, may significantly help to improve the models' predictive performance. Interestingly, the overall top score comes from a combined learning model, namely the SimLin model, that shows a F-Score of 80%.

User	Annotations	Location	PersonalityEmotion	LSCF	SimLin	SimWu	SimShortestPath	SimLSCF
0	80	54,42%	61,22%	69,44%	80,00%	68,03%	64,76%	47,62%
1	114	79,19%	51,79%	64,94%	79,60%	69,96%	73,54%	61,54%
2	105	59,40%	65,33%	77,77%	65,08%	65,33%	65,74%	58,41%
3	419	61,71%	63,77%	65,93%	63,79%	63,22%	56,98%	63,68%
4	143	46,88%	45,89%	49,14%	47,00%	49,08%	48,61%	52,95%

Table 9.6: Single-user model results (weighted F-Scores) for the 5 most frequently annotating users at the higher semantic representation level.

All in all, single-user models work for some users well, while for others don't. It is not only the data size, but also the properties of the respective dataset that plays a role. Data-driven models such as ANNs can't handle low quality dataset very well, especially in cases of feature or class imbalance. Using a multi-user model helps to overcome this kind of issues. However, at the same time, a model trained with data of many probably won't be able to provide perfect scores.

9.6 Conclusion

In the presented work, we define and explore the ability of a dynamic user modelling construct, which we refer to as *Location-Specific Cognitive Frame (LSCF)*, in capturing the experience gained by a user when visiting a certain location at a certain time. Moreover, we investigate whether and to what extend user-specific psychological features, such as the personality and the emotional state, encapsulated in the LSCFs (in addition to further context information), affects the performance of LSTM neural networks when it comes to modelling and predicting human movement patterns. In order to evaluate our approach with a real-world dataset, a 8-week long user study with a total of 21 participants has been designed and carried out. It could be shown that the LSCF-based approach performs well both in terms of prediction accuracy and a reduced training time compared to the baseline models. Especially the shorter training time, makes it attractive for use cases with large data sets or low computational power. In addition, we introduced and investigated *combined learning*, a model training approach based on semantic similarity analysis. It can be shown that *combined learning* can contribute to further reducing the training time. Finally, we evaluated our user modelling approach on both a multi-user, as well as on separate single-user location models, with the latter reaching very high (comparably) accuracies provided that a big enough dataset is available.

CHAPTER 10

SEMANTIC-ENHANCED LEARNING (SEL) ON ARTIFICIAL NEURAL NETWORKS USING THE EXAMPLE OF SEMANTIC LOCATION PREDICTION

Abstract

Machine learning models, such as Artificial Neural Networks (ANNs), find nowadays a widespread use, whether in respect of data mining and forecasting or in the area of classification. However, real-world situations comprise complex, multi-class or multi-label estimation tasks that carry a certain *semantic load* and thus bring a certain degree of *fuzziness* with them. This is a fuzziness which humans, due to their common sense knowledge and their personal experience, can easily understand by linking, for instance, the underlying concepts together, while machines may from scratch not. This makes humans able to adapt themselves faster to the respective tasks. In contrast, a vast amount of both training data and time are necessary in order for a computational model to be capable of learning such kind of relations and adapting to new situations. In this chapter, we show that letting explicit semantic knowledge flow into a predictive model, such as a neural network, leads to an improved performance with regard to training time, accuracy and robustness. In particular, motivated by the promising results of using similar locations in Chapters 4 and 9, we propose adding an extra *semantic layer* to the network, whose role is to provide the network with information about the semantic interrelation of the treated classes *in advance*, saving in this way valuable training time and improving the quality of the model. We present, explore and evaluate several different variants of our approach and we illustrate their functionality in a semantic location prediction scenario using 2 different real-world datasets. The proposed method can be minimally adapted and used for optimizing other machine learning training algorithms as well.

The contents of this chapter is based on a paper under review at KDD 2019 [KB19].

10.1 Introduction

Artificial Neural Networks (ANNs) have long spread beyond the research community and are already being widely used for solving real-world tasks. In particular Deep Learning (DL) and Deep Artificial Neural Networks (DANNs) play an increasingly important role in the future of A.I. due to their recent very promising results in various areas of application, such as in the image and speech recognition domain as well as in diverse classification tasks.

As most of machine learning based approaches, ANNs are data-driven methods and demand a large amount of training data in order to perform their task properly. In addition, usually, real-world data carry a certain semantic load with them. For example let us say that a stock market forecasting model predicts that the share value of coffee will increase rapidly in the next couple of hours. But does it understand what its own prediction outcome means and how this relates ($\rightarrow$ *semantic relation*) to the rest of the world? And could this additional semantic information contribute to improving the model's future performance? Or let us consider the semantic location prediction scenario, which we use in this chapter to evaluate and underpin our argumentation. The particular scenario refers to both a sequence learning problem and a multi-class prediction task, that consists of predicting the future *semantic location* (location type), which is going to be visited next by some user based on his or her movement history. The movement history is in this case captured in *semantic trajectories*, which as already mentioned before reduce the available movement data to the essentials (see Section 2.2). So, let us now say that a respective prediction system outputs "burger joint", while the correct location is "pizza restaurant". Existing training algorithms would evaluate the particular estimation as wrong and would penalize the model accordingly (e.g., in the case of an ANN, by updating its weights). Despite the fact that "burger joint" and "pizza place" are from a semantic point of view not so different. They belong both to the high-level super class location "fast food restaurant", which in turn belongs to the higher-level class "restaurant" and so forth. A human would probably evaluate the same result in a softer manner, namely as *"pretty close, but not close enough"*. Thus, from a human's point of view, it seems intuitive to penalize the predictive model less severely. But how exactly could

this information be important in our location prediction use case?

In general, human movement patterns follow a certain logical order based on the semantics behind the visited locations. These semantics can be captured in hierarchical, graph-like constructs as described in Section 10.3. Depending on the semantic level, in which movement patterns are described, different conclusions can be drawn. The wider and deeper the perspective upon the semantic meaning of locations and their interrelation is, the more the corresponding semantic trajectories are able to cover the user's fundamental movement patterns. Moreover, it has already been shown in the previous chapters of this thesis that predictions at high-level semantic patterns can be more accurate than at lower-level ones. Thus, it seems reasonable to expect that "bringing meaning" into the learning process of machine learning models by *integrating high-level semantic information*, coming for instance from an external knowledge base, could lead to an improved performance. In tangible terms and with respect to our example, this would mean that the predictive model could perform better if it knew that the location "burger joint" is in some degree similar to the location "pizza restaurant", since the user's next semantic location (e.g., coffee shop) would probably be more related to the corresponding high-level semantic type "food location" than to the individual specific type.

In this chapter, we propose *adding an extra semantic layer* to the network in order to incorporate such kind of semantic knowledge into our model. We refer to it as *Semantic-Enhanced Learning (SEL)*. The role of this additional semantic layer is to provide the neural network *in advance* with knowledge concerning the (semantic) interrelation of the treated location classes. By doing this, we aim at making the training process more efficient in terms of a reduced training time and/or amount of training data needed, while at the same time making the model more confident about its estimations and thus more robust. This could be especially handy when only a small training dataset is available in which some test data classes are not included but which are semantically related to the ones that are (included). This could also be very helpful in the case of sequence learning like in our case, where the sequence order is highly related to the semantics behind the corresponding elements. We propose a total of 3 variants of our approach. Each variant differs from the other two mainly based on the position in the network at which the layer

is applied and the type of semantic knowledge it considers.

We can show using the semantic location prediction scenario that incorporating an additional semantic layer can lead to a better performance with regard to accuracy, training convergence and model robustness. Moreover, SEL could help overcome complex and challenging real-world multi-class or multi-label classification and pattern recognition tasks, such as the image segmentation and the road scene recognition tasks in the field of autonomous driving.

This chapter is structured as follows. First, we provide a brief insight into some of the most related work that served as a basis for our own approach. Next, in Sections 10.3 and 10.4, we discuss briefly about knowledge bases and reiterate the semantic similarity analysis measures from Chapter 9. In Section 10.5, we present in detail our approach, SEL. Finally, in Sections 10.6 and 10.7, we evaluate our approach and provide some concluding thoughts respectively.

10.2 Related Work

To the best of our knowledge, there exists no approach that integrates structured semantic knowledge into a neural network, in a similar way as we propose in this chapter. However, the general idea of injecting world knowledge into machine learning models has been partially explored.

A common approach for instance is to define formulations of general categorical constraints, e.g., in the form of model priors (e.g., Bayesian machine learning), regularization methods and user-defined kernel learning, in order to solve application-specific problems. The case of *user-defined kernel* and *multi-kernel learning* [Bis06, HAM17] is here particularly interesting due to the fact that their similarity analysis principle bears some resemblance to our work. In the matter of fact, the basic idea of our work lies partly on projecting the kernel learning approach onto the neural network training case, while adapting and extending it accordingly. *String kernel learning* approaches are especially interesting, since their scope covers natural language processing tasks, which in a way involve semantics as well. However, their usage is reduced mainly to text mining applications, like in [LSST+02], instead of dealing with semantically enriched real-world sensor data. Moreover, String Kernel learning

systems operate mostly in the Vector Space and use vector distance metrics for determining the similarity between two concepts (or their features). In contrast, in our chapter, we investigate several semantic similarity metrics applied on a structured (ontological) knowledge base that go beyond plain distance measurement. Beside term-to-term distance, our similarity metrics take depth (number and type of semantic levels), information content, topological and entity density and common properties (Jaccard Index) into account as well. Finally, although some common features (for instance from a statistical analysis point of view) can be found between kernel learners and neural networks, these still constitute two different ML methods (e.g., instance-based method vs parametric one) and here, we focus on the customization of the training algorithm of the latter.

A further algorithm that relies on similarity analysis as well, in order to raise the prediction accuracy while keeping the training time low is the *one-shot learning* algorithm. The one-shot learning method is used mainly in computer vision for classifying objects in image data [FFFP06]. Most machine learning based object classification algorithms are data-driven and require a great amount of training data in order to perform well. In contrast, one-shot learning is reliant on a very small number of training examples. In some cases, it just needs a single example [BU05]. In principle, it uses already learned knowledge for learning new visual object models rather than learning these from scratch. This is done by analysing the similarities between the new and the previously learned classes. Vinyals et al. illustrate one-shot learning on a deep neural network architecture [VBL$^+$16].

As already mentioned in the Introduction, our approach provides a way to incorporate the semantics and the involving *fuzziness* of the treated classes and concepts in a certain neural network's prediction task into its training mechanism. In general, the concept of fuzziness with respect to artificial intelligence was first introduced in the Fuzzy Set theory by Zadeh in 1965 ([Zad65]). The underlying idea and motivation was to provide so called fuzzy logic systems with the capability of capturing the real-world vagueness and expressing linguistic variables, such as "cold", "warm" and "slightly warm", through fuzzy values and fuzzy sets by applying a set of rules based on some linguistic multi-valued model [NR75]. Nguyen et al. [NVH14] investigate whether and how us-

ing both binary and auxiliary categorical label information (which reflect fuzzy descriptions, such as "strongly-disagree", etc.) could improve classification or regression (e.g., SVM/R). One could define these auxiliary labels as "semantic" knowledge. Their *soft-label* learning approach shows a similar smooth learning effect like in our case. However, our work differs by concentrating primarily on investigating the impact of using comprehensive structured knowledge representations on the training algorithm of artificial neural networks. In our case, the use of semantics and our semantic similarity analysis on an ontology-based world representation goes beyond using plain fuzzy-like categorical labels and thus, our work could be regarded as an extension of Nguyen et al.'s work in the special case of neural networks.

Our approach does not describe a knowledge distillation or transfer algorithm, where knowledge is transferred to a distilled model, such as in Hinton's work [HVD15]. However, knowledge transfer does indeed take place. In our case, structured knowledge is transferred from an ontological knowledge base into the neural network model at a certain position and is then forward- or backpropagated towards the out- or input through all its layer respectively. This leads to a similar effect of producing softer probability distributions over classes as we also see in Hinton et al.'s work.

10.3 Semantic Knowledge Bases And Ontologies

The term *Semantics* refers to the *analysis of word meanings and the relations between them* ([Oxf17a]). A (semantic) *knowledge base* refers in information science to a knowledge representation technology that stores knowledge, facts of the world or of a certain domain, in a structured model [HRWL84]. In the case of a structured, schema-oriented knowledge base, we speak of *Ontologies*. [Gru95] defines ontologies as follows:

> *An ontology is an explicit specification of a conceptualization.(...)*
> *with human-readable text describing what the names mean(...)*

In tangible terms, an ontology represents a part of the world and consists of a set of classes, subclasses, properties and their interrelations mapped in a directed acyclic graph-like structure. The simplest form that an ontology

can take is a so called *taxonomy*, which contains solely subsumption, that is, Superclass-Subclass relations.

In this chapter, we use ontologies for storing and organizing the semantic locations in our training and evaluation datasets into a specific hierarchy. We oriented ourselves on the Foursquare venue taxonomy[1]. Fig. 10.1 shows a part of our location ontology, which we used to train and evaluate our approach.

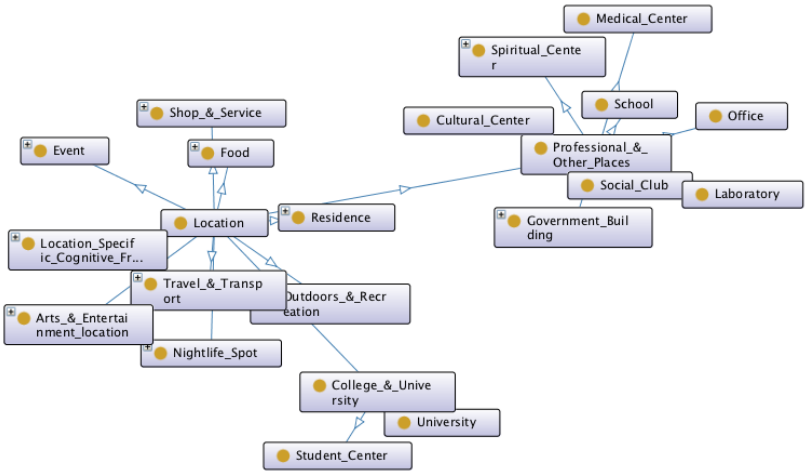

Figure 10.1: Part of our MIT Reality Mining Semantic Location Ontology.

10.4 Semantic Similarity

Semantic similarity defines whether and to what extend two or more concepts are similar to each other with regard to their semantic content. It is usually applied in the field of Computational Linguistics for analysing various text corpora. In our work, we use semantic similarity to determine the semantic relation between the treated classes by our neural network. Here, we reiterate ordered with a growing complexity briefly the equations of the three ontology-based semantic similarity measures that we apply in this chapter and which were already presented in Chapter 9.

Rada et al. [RMBB89] introduced a method based on measuring the distance of the *shortest path* between two concepts in an ontology. The shorter the distance, the greater their similarity. Their approach is rather one of the

[1]https://developer.foursquare.com/docs/resources/categories

simplest and it is further limited to taking just subsumption links into consideration. However, it provides a general impression of how close two concepts are. Eq. 10.1 defines Rada et al.'s semantic distance:

$$SemDist_{Rada}(C_i, C_j) = min(d(C_i, C_j)) = min(|edges_{C_i,C_j}|), \tag{10.1}$$

whereby C_1 and C_2 are two concepts contained in an ontology, and $|edges_{C_i,C_j}|$ the number of edges between them.

Wu and Palmer [WP94] defined a semantic similarity measure that considers two different path distances in the graph at the same time. Apart from the distance of the shortest path between the two concepts C_1 and C_2, they also take the distance of the shortest path between the root and the Least Common Superconcept (LCS) of C_1 and C_2, the so called *depth* of the LCS, into account:

$$Sim_{Wu,Pa}(C_i, C_j) = \frac{2 \cdot min(d(root, LCS(C_i, C_j)}{min(d(C_i, C_j)) + 2 \cdot min(d(root, LCSC_i, C_j))} \tag{10.2}$$

The depth in a semantic network, like in an ontology, reflects indirectly the level of Information Content (IC) of the corresponding concept. The deeper a concept is, the more information content it encapsulates. The information content is defined through the following equation (Eq. 10.3), with $P(C)$ referring to the concept C_i's probability of occurrence in the universe of discourse defined by the ontology, which in turn can be inferred from counting the occurrence frequency of the respective concept in the existing dataset. The information content depends on the structure, that is the density of the ontology.

$$IC(C_I) = -log(P(C_I)) \tag{10.3}$$

Finally, *Lin* [Lin98] introduced a similarity metric, which relies both on the semantic distance and the density of the ontology containing two concepts C_1 and C_2 . By calculating the information content of the concept and their Least Common Superconcept (LCS), Lin takes the semantic distance of each of them indirectly into account. Their metric is defined in Eq. 10.4:

$$Sim_{Lin} = \frac{2 \cdot IC(LCS(C_i, C_j))}{IC(C_i) + IC(C_j)} \tag{10.4}$$

In this work, we evaluate three different versions of our Semantic-Enhanced Learning algorithm based on the three aforementioned semantic similarity metrics.

10.5 Semantic-Enhanced Learning (SEL)

In this section, we present and describe 3 different variants of our Semantic-Enhanced Learning approach: *Semantic Similar Location layer (SemSimLoc)*, *Low-Level High-Level Location layer (LowHighLevelLoc)* and *Semantic-Enhanced Backpropagation (SEBP)*. Each of them describes the application of an additional semantic layer and differs from the other two based on the position of the neural network at which the layer is being applied and the type of semantic knowledge it considers. As already mentioned before, this extra semantic layer is responsible for providing the network with additional knowledge regarding the semantic interrelationship between the available location classes. However, in order for this knowledge to be incorporated into the mathematical neural network model, it has to have the right type. The semantic layer achieves that by feeding the result of a semantic similarity or relatedness analysis applied on some respective knowledge base (see Section 10.3) and based on the metrics introduced in Section 10.4 back into the network. Thus, the here introduced semantic layer comprises the following chain of actions: structured representation of the available semantic entities (locations), semantic similarity analysis over the locations, incorporation of the analysis' returned value, e.g., one-hot encoded similar location set and/or corresponding similarity score (see below) into the network.

10.5.1 Semantic Similar Location Layer (SemSimLoc)

This layer extends the input of the neural network by considering semantically similar locations as well and makes especially sense when dealing with semantically enriched sequences like in our case. The underlying idea behind this approach is that similar locations lead to similar future locations too. That is, people that usually visit a *restaurant* after watching a movie in the *cinema*, would probably also visit a *restaurant* after watching a play in a *theatre* or watching a live band at a *concert hall*. So, if we provide our network during

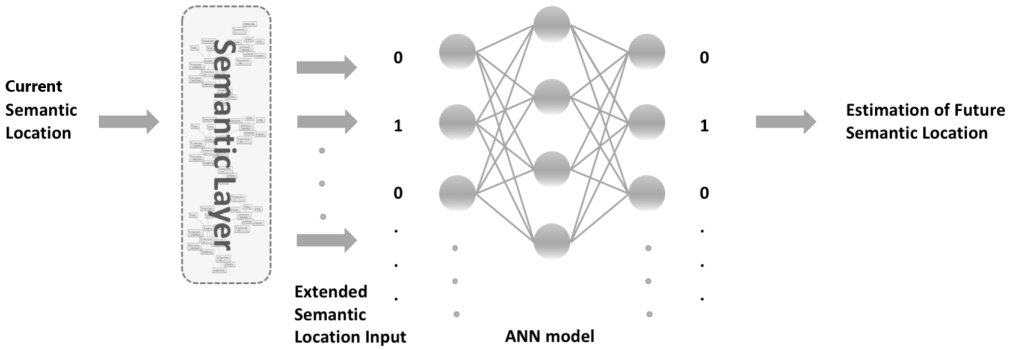

Figure 10.2: SemSimLoc(+) and LowHighLevelLoc(+) approach.

the training, apart from the actual (current) location, with similar locations as well, we produce a certain *shortcut* effect into the sequence learning process. This gives the network the chance to learn certain location transitions faster and easier.

Furthermore, extending the input by using similar locations brings an additional benefit. Often, having a sparse training dataset, sometimes means that certain classes in the test data set, or class transitions as in our case, are covered either partly or not at all. This means that a model trained with this data would have to take a random guess when meeting one of these classes during the prediction phase. Using similar locations as additional input can help overcome this issue.

The number of the similar locations n that are taken into consideration represents a hyper parameter and will also be investigated in the evaluation section (Section 10.6). *SemSimLoc**n*** refers to a semantic layer that extends the original location by adding n additional (most) similar locations. Fig. 10.2 illustrates how the additional semantic layer extends the input of the neural network. We propose 2 SemSimLoc layer versions. One that returns solely a set of similar locations (*SemSimLoc*) and one that returns the respective similarity scores (to the current location) as well which we will refer to as *SemSimLoc+*. The latter aims at providing the network with an additional quantitative training feature.

10.5.2 Low-Level High-Level Location Layer (LowHighLevelLoc)

Basically, this layer wraps up in a semantic manner the above described SemSimLoc layer. Like the SemSimLoc layer, it is also applied at the input of the neural network (see Fig. 10.2), but instead of extending the input by adding similar locations, this layer considers the locations' hierarchy and extends the input by taking the corresponding higher-level location type. In tangible terms, by doing so, we aggregate the group of similar locations that we had in the SemSimLoc layer into a single location class at a higher semantic representation level.

By semantically unfolding the input locations in this way, in addition to the desirable benefits of the SemSimLoc layer, we aim at taking advantage of the good performance that the high-level representation has shown previously in this work (e.g., see Chapters 3, 6 and 7).

10.5.3 Semantic-Enhanced Backpropagation (SEBP)

The principle idea of this variant lies in *penalizing or "rewarding" the network based on the semantic content of tis own predictions.* In order to achieve this, we extended the learning architecture as shown in Fig.10.3.

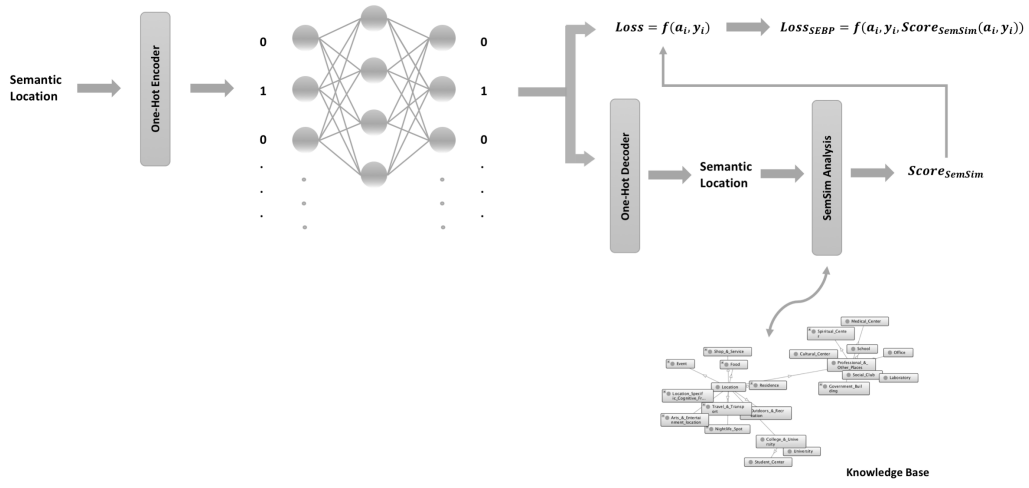

Figure 10.3: Semantic-Enhanced Backpropagation (SEBP).

In contrast to the two aforementioned variants of our approach, the semantic

layer is added at the output of the network. We further customized the original Backpropagation algorithm by appending an additional semantic-dependent factor c_{sem} to our Cross Entropy Loss function L illustrated in Eq. 10.5.

$$L = -\frac{1}{n}\sum_{X}\sum_{j}(y_j ln a_j + (1-y_j) ln(1-a_j)), \tag{10.5}$$

with n representing the total number of training examples, X the number of inputs, a_j the outputs of the network and y_j the desired outputs (*ground truth*). Compared to the more common *Sum Squared Error* function, the cross entropy loss function provides a faster learning curve (given a sigmoid activation function) and fits better to classification tasks, whereas SSE performs better in regression problems. In particular, we modified the L of Eq. 10.5 as follows:

$$L_{SEBP} = L \cdot c_{sem} = L \cdot (1 - Score_{SemSim}(a_i, y_i)), \tag{10.6}$$

whereby $Score_{SemSim}(a_i, y_i)$ refers to a normalized semantic similarity score like the ones described in Section 10.4. It takes values from 0 to 1, and describes the semantic similarity between the output of the network a_i and the desired value y_i (*ground truth*). That is, we bring semantic knowledge into the learning process *by providing the model information about the nature of its own estimation and its relation to the ground truth.* With respect to our special use case in this chapter, the semantic location prediction, the semantic knowledge comes in form of a location taxonomy, as the one described in Fig. 10.1. The choice of the knowledge base depends on the nature of the classification task.

This slight but significant modification of the standard loss function aims at making the network's learning process *softer*. According to our approach, predictions that don't match the ground truth values, but are semantically similar to them, will be less penalized. In other words, depending on the degree of the semantic similarity, the algorithm reacts harder or softer. Two completely different locations for a_i and y_i would result in our algorithm behaving the same as the original backpropagation algorithm. While in the case of these two being similar, SEBP would react softer. This provides our learning approach with a certain degree of fuzziness. One could say that a SEBP-based algorithm behaves more human-like.

10.6 Evaluation And Discussion

The semantic location predictions represents a real-world use case in which the use of (external) semantic knowledge can both offer a real added value and be easily integrated due to the existence of various open online gazetteers (e.g., OpenStreetMap[2], Google's Places API[3], ...). We applied and evaluated our SEL algorithms with regard to *training behaviour* and *test accuracy* using 2 real-world datasets: the SDP dataset collected through the user study described in Section 9.4.1 and the open MIT Reality Mining dataset [EP06], which was already discussed previously in this thesis. We again preprocessed the data by filtering out double entries, irrational jumps between consecutive locations, and inconsistent, sparse annotations ($<$ 5000 entries) ending up with a relatively consistent dataset containing almost 300.000 entries.

Due to its good performance in the previous chapters, we choose the multi-user Long Short-Term Memory neural network (LSTM) to be our semantic location prediction model. We kept a similar hyper parameter set, which is shown in Table 10.1 and the semantic locations were fed into the network encoded as one-hot vectors. In addition, we selected the *Rectified Linear Unit*

LSTM units	LSTM layers	Sequence length	Batch size	Learning rate	Activ. function	Optimizer
128	1	1	16	0.001	ReLu	Adam

Table 10.1: LSTM hyper parameter values.

(ReLU) to be our activation function, *Cross Entropy* and *Adam* as our loss and optimizing function respectively and a *Softmax* layer to be our final output layer. We split the dataset into a training and a test dataset with a ratio of 80% to 20%. Since our data consist of location trajectories, which are spatio-temporal sequences, keeping the temporal coherence is important, especially when it comes to artificial neural networks. Instead of using shuffled or stratified cross-validation and similar methods that split the data randomly (see Scetion 2.3.1), we applied a customized 3-fold *walk forward validation* method, where the data are split in a certain order with a moving observation window sliding upon them. In our case, we moved an 80%-of-the-data long

[2] `https://www.openstreetmap.org/`

[3] `https://developers.google.com/places/web-service/intro`

window a total of three times until we got back to the starting point. At the end, we calculated the average to determine the final training accuracy and loss value curves as well as the corresponding test accuracy scores.

Fig. 10.4 contains the training behaviour of the SemSimLoc layer in terms of training accuracy and how the loss value evolves over a period of 10 epochs in contrast to having just the original location layer (*OrigLoc* (black curve)) for the SDP dataset. Moreover, it illustrates the training performance of 5 SemSimLoc variants (SemSimLoc1, ..., SemSimLoc5), whereby each extends the input by 1, 2, ..., 5 semantic similar locations accordingly. Finally, Fig. 10.4 compares the additional layer based on the three semantic similarity metrics mentioned in Section 10.4. It is apparent from the figure that the additional semantic layer has indeed a significant impact on the training behaviour of our model. With regard to training accuracy, it seems that the SemSimLoc layer helps to achieve higher accuracy values while causing a certain smoothing effect at the same time. In particular, the more the considered similar locations n, the higher the accuracy and the smoother appears to be the training. The SemSimLoc5 (magenta curve) shows practically no zig-zag behaviour at all in all three plots, independently of the similarity metric. From all 3 metrics, we can see that Wu and Palmer's metric returns the smoothest training curves. This could be interpreted as an indication that taking the Least Common Super-Location type into consideration (see Section 10.4) can be helpful, a fact, which in turn supports our idea of a LowHighLevelLoc layer that we evaluate later in this section. With regard to the way the loss function evolves among the 10 epochs, what stands out in the figure is that the SemSimLoc layer results in all three cases in overall lower loss values. The most interesting finding here is that like before, the number of considered locations n seems again to be playing a major role. The larger the n, the faster the loss curve reaches lower values and the lower the respective values are. While a faster convergence reflects shorter training times, an overall lower loss curve can be interpreted as a more consistent, confident, and thus more robust model. That is, the SemSimLoc layer results in a model that is more confident about its correct predictions than a model without this additional layer. This supports our initial hypothesis that extending the input through semantically similar locations improves the quality of our model. Similar findings could also be

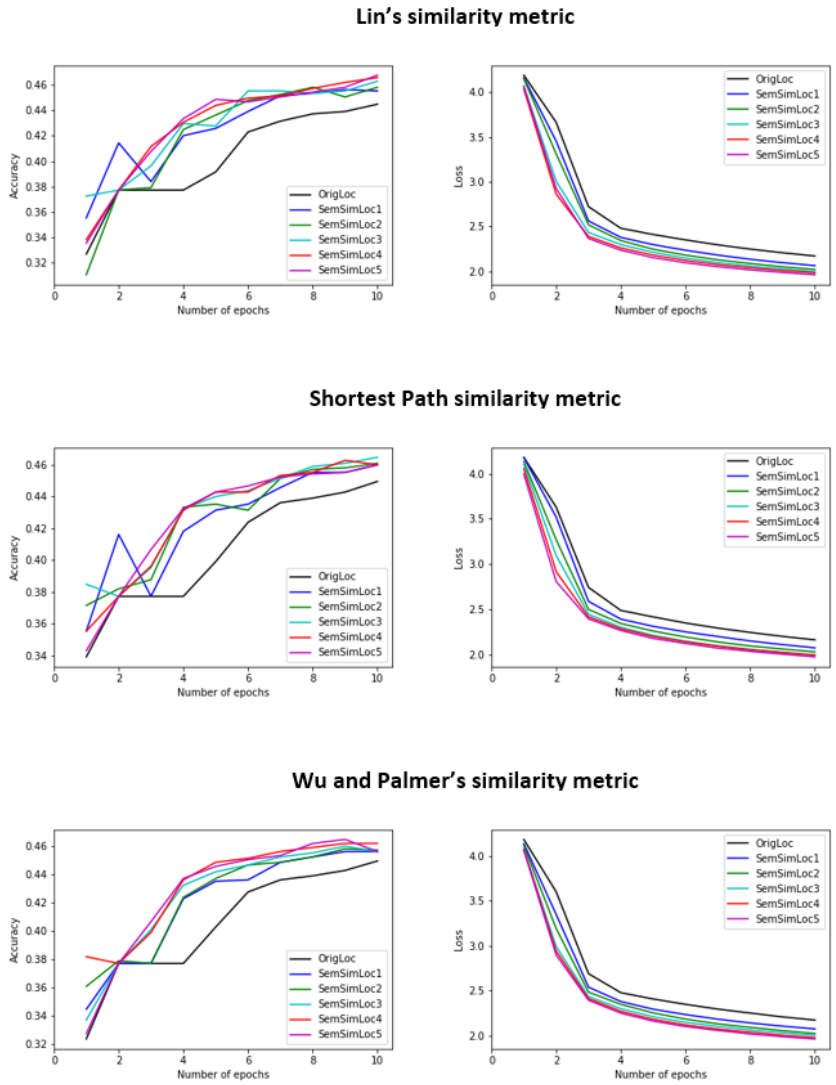

Figure 10.4: SemSimLoc (10 epochs, SDP).

observed considering a period of 100 epochs as shown in Fig. 10.5. Figure 10.6 contains the test accuracy scores for both the 10 and the 100 epochs case. It is apparent that the additional semantic layer improves the test accuracy scores as well. The benefits of the SemSimLoc layer approach become particularly obvious in the case in which the model was trained only for 10 epochs. The SemSimLoc layer helps the model to reach faster higher test accuracy scores in contrast to the baseline. What also stands out in the figure is that SemSimLoc5 shows together with the baseline (OrigLoc) one of the worst performances, both for the 10 and the 100 epoch case. Considering its training values, this could be mainly attributed to a certain overfitting effect caused by the use of too many

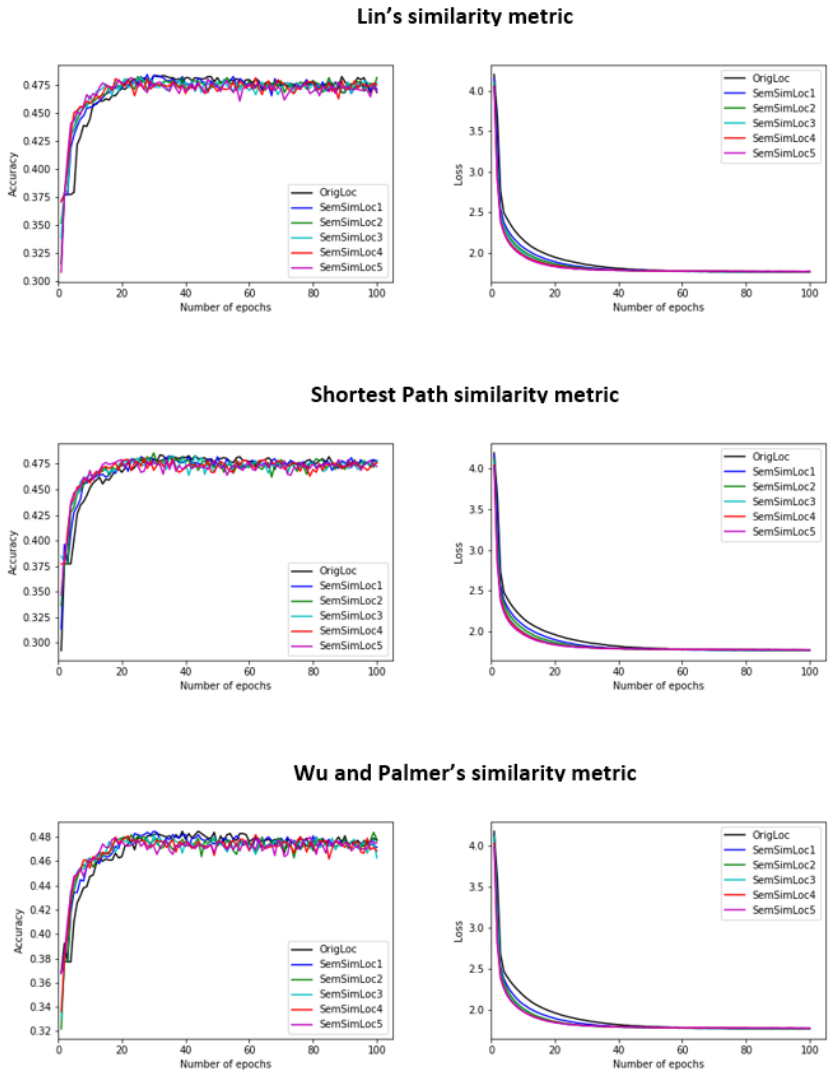

Figure 10.5: SemSimLoc (100 epochs, SDP).

similar locations as input. In Fig. 10.7, we evaluate the use of SemSimLoc+, a semantic layer, which takes beside similar locations, their respective similarity scores into account as well and feeds them also into the network. In general, it can be seen that feeding our model with the corresponding similarity scores amplifies the positive effect that we saw earlier. Both accuracy and loss curves are steeper, which means that the model converges and thus learns faster than the normal LSTM model. At the same time, compared to Fig. 10.4, all curves follow a much smoother course, even the ones of Lin and Shortest Path. What is also striking about the plots in this figure is the very good performance of the SemSimLoc+ variant based on the Shortest Path similarity score with respect

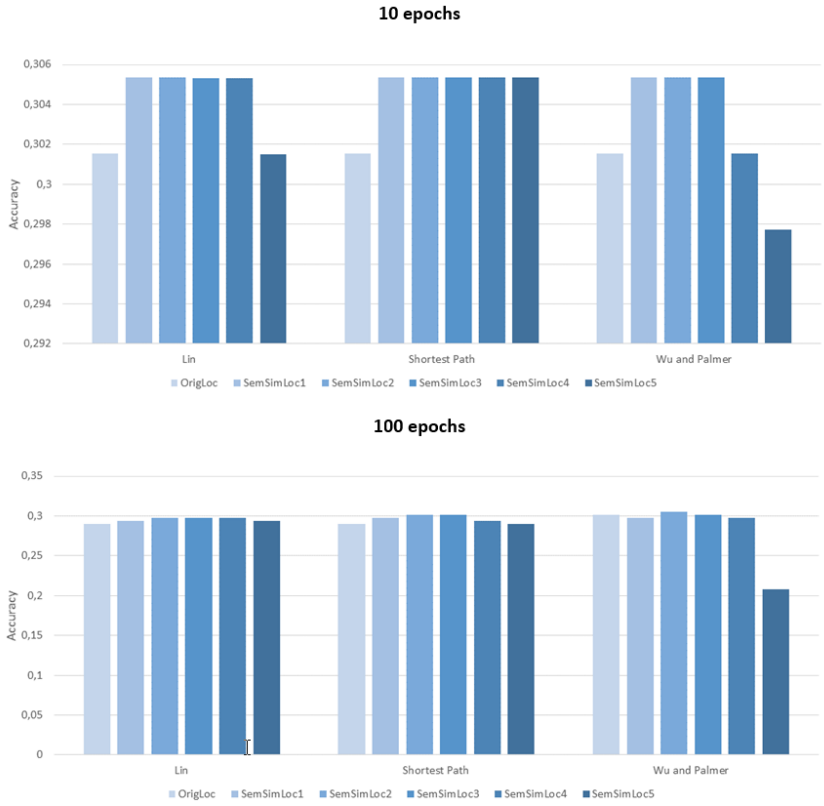

Figure 10.6: Test accuracy SDP (10 and 100 epochs).

to the convergence speed. Fig. 10.8 shows the respective SemSimLoc+ results for a training period of 100 epochs. The test accuracy results for SemSimLoc+ can be found in Fig. 10.9. We can again see that the added semantic layer brings benefits with it. If we compare Fig. 10.9 with Fig. 10.6, we can see that taking the similarity scores into account as well leads in some certain cases a significant performance boost. However, this can't be clearly pinpointed to a single attribute since sometimes it appears in SemSimLoc4+, sometimes in SemSimLoc1+, and SemSimLoc2+ as well as SemSimLoc3+ provide generally also high test accuracy scores.

Fig. 10.10 illustrate the training performance of both the SemSimLoc and the SemSimLoc+ approach for the MIT dataset. The regarded SemSimLoc layer relies in both cases on the Shortest Path similarity metric. The other two metrics show a very similar behaviour. Here, we see a slightly different training behaviour of the model. With respect to how the training accuracy evolves, both the SemSimLoc and the SemSimLoc+ layer result in overall higher scores,

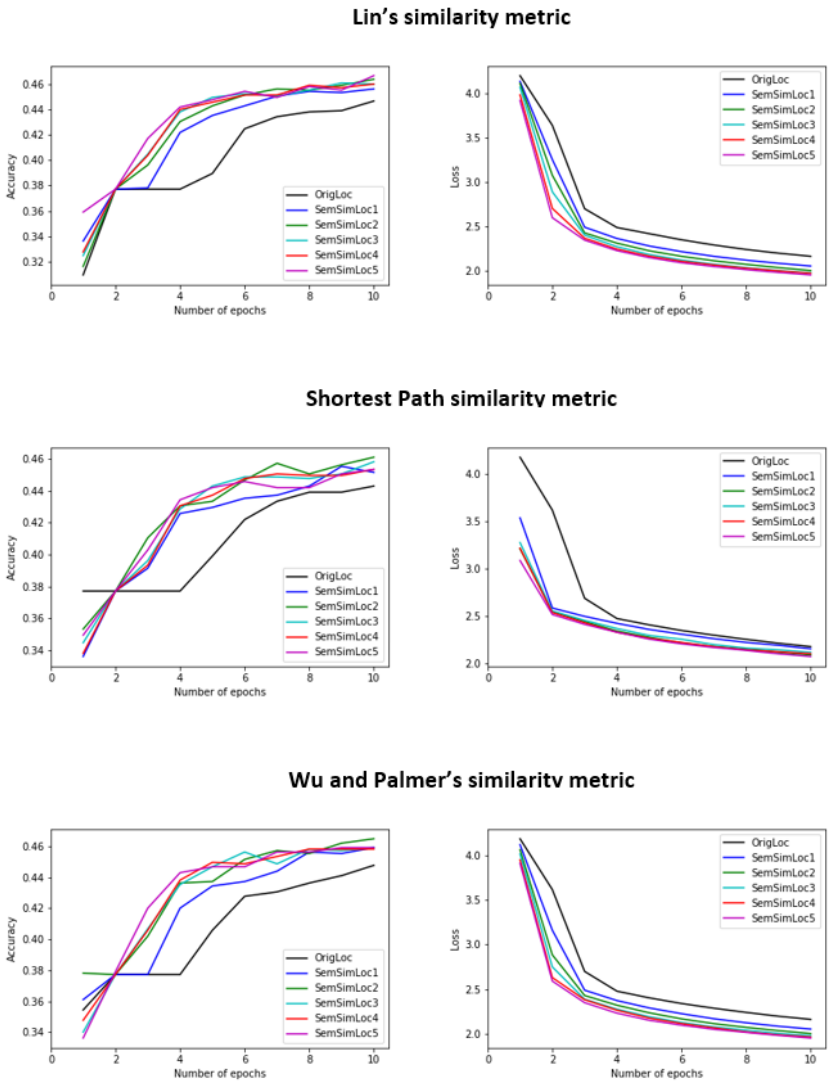

Figure 10.7: SemSimLoc+ (10 epochs, SDP).

however, these are achieved in a later point of time during the training than with the SDP dataset. With respect to how the loss value evolves, there are no differences to be recognized compared to our baseline model OrigLoc.

In terms of test accuracy, we could again observe that in certain cases the SemSimLoc as well as the SemSimLoc+ layer could lead to a performance boost, while in the rest of the cases either minor or no improvements could be observed (Fig. 10.11).

Fig. 10.12 and 10.13 illustrates the training performance of the model with an additional *LowHighLevelLoc* layer this time. The LowHighLevelLoc layer extends the input of the network by taking at each step the higher level location

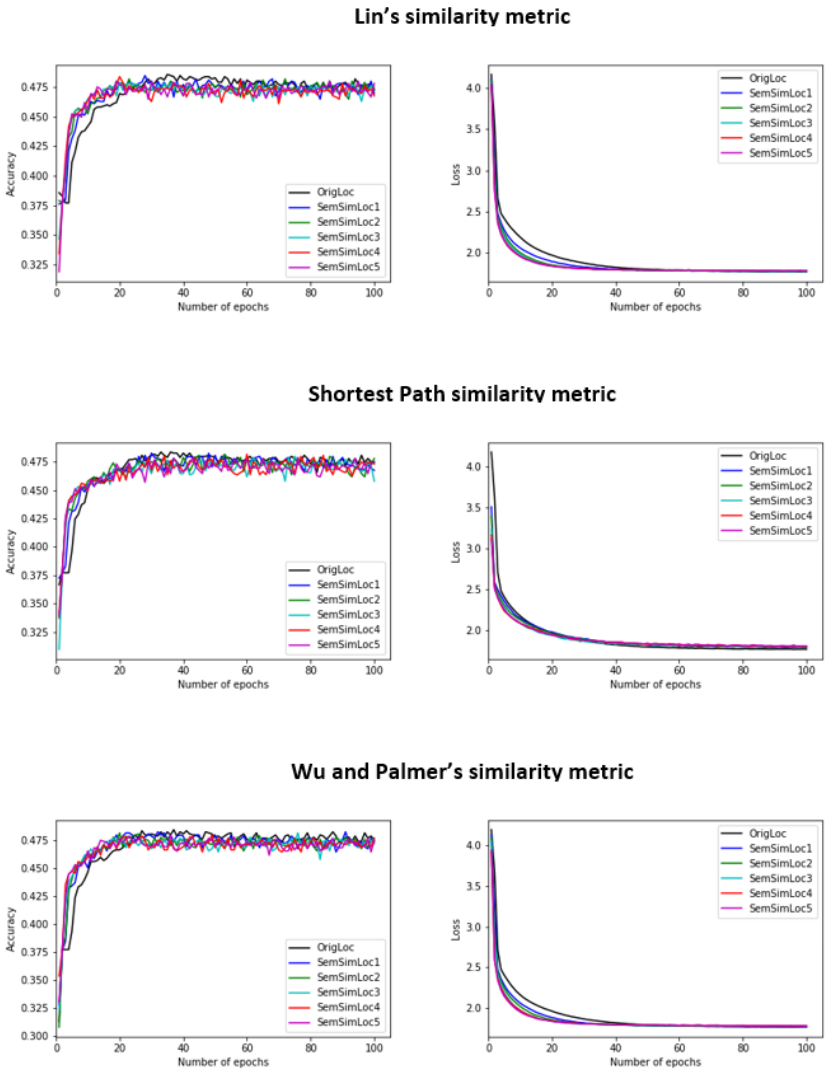

Figure 10.8: SemSimLoc+ (100 epochs, SDP).

type to which the current location belongs to additionally into consideration. In a sense, the LowHighLevelLoc layer envelopes the SemSimLoc layer approach. While for the SDP case the LowHighLevelLoc shows a similar trend to the SemSimLoc approach as expected based on the aforementioned results, it doesn't seem to have any positive (or negative) impact in the MIT dataset case. A possible explanation for this might be a certain inappropriate taxonomization of the locations in the MIT dataset from our side. For instance, a frequently occurring location type in the MIT dataset is the semantic location *Lab*, *Media Lab* and *ML*, all referring to the same location, a specific lab at the MIT campus. Knowing that the participants of the study were mainly

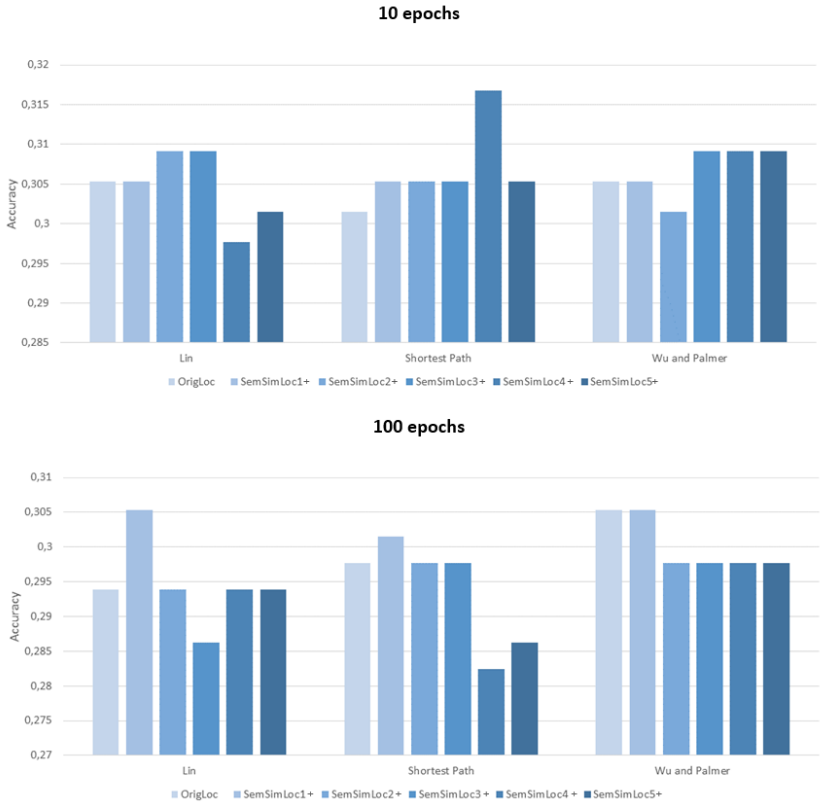

Figure 10.9: Test accuracy, SDP, SemSimLoc+ (10 and 100 epochs).

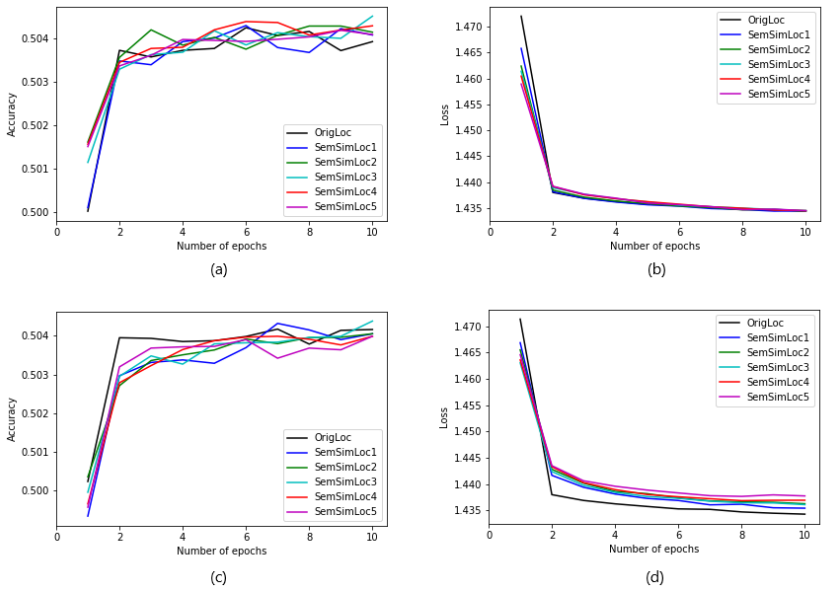

Figure 10.10: SemSimLoc (top) and SemSimLoc+ (bottom) (10 epochs, MIT, Shortest Path).

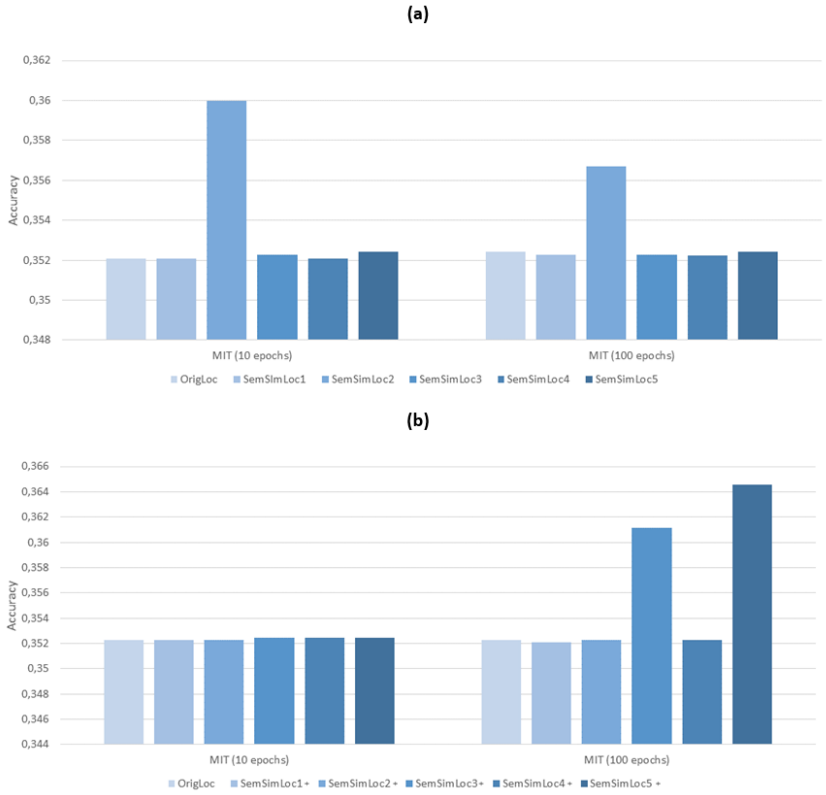

Figure 10.11: Test accuracy: SemSimLoc (a) and SemSimLoc+ (b)(10 and 100 epochs, MIT, Shortest Path).

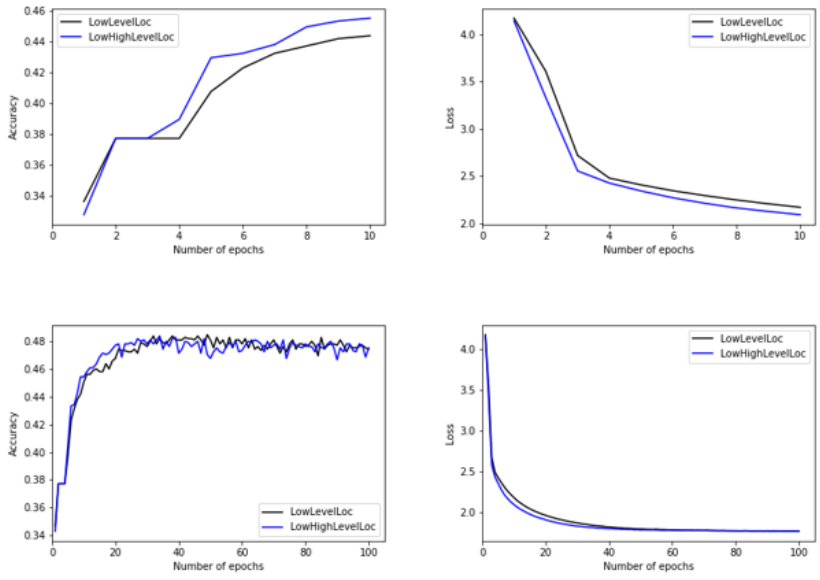

Figure 10.12: LowHighLevelLoc (10 (left) and 100 (right) epochs), SDP.

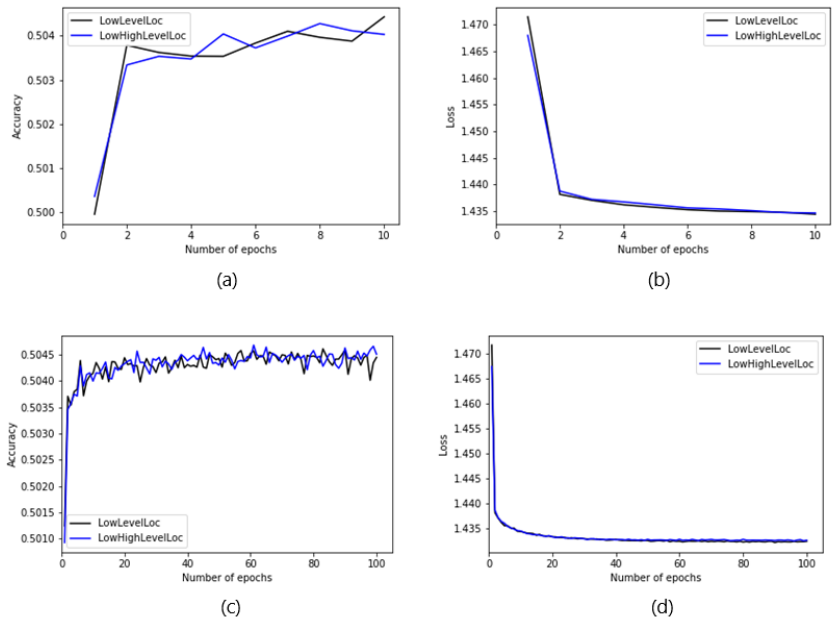

Figure 10.13: LowHighLevelLoc (10 (left) and 100 (right) epochs, MIT).

students, we assigned this particular location to the super class *university*. However, some of the participants were faculty in this lab. In their case, the more appropriate super class would be *office* or *work*, a location type that usually features different location transitions than the university. This and similar "false" assignments could have been misleading our network during the training. Fig. 10.14 contains the test accuracy scores for both the SDP and the MIT dataset. Like in the figures before, the LowHighLevelLoc layer shows

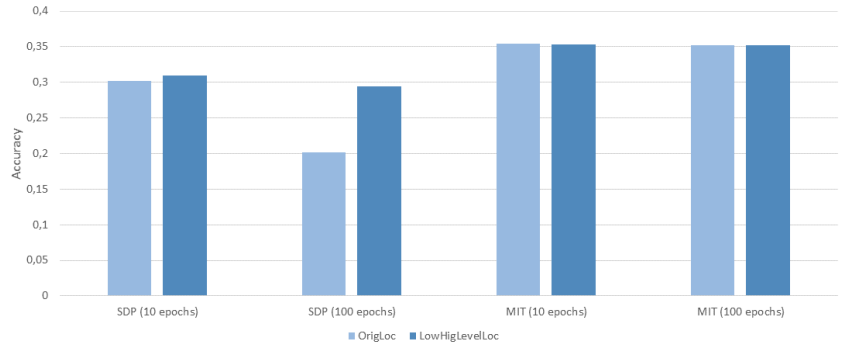

Figure 10.14: Test accuracy: LowHighLevelLoc (10 (left) and 100 (right) epochs, SDP and MIT).

its advantages in the SDP case more clearly than in the MIT case, especially

after a 100 epochs long training. The slight inconsistency between the SDP and the MIT results might be due to the fact that the MIT dataset, similar to other open datasets (e.g., Geolife [ZWZ+08], Gowalla[4], etc.), is a sparse dataset, meaning that very often, consecutive entries belong to different days or even weeks and months. This has as a result, that many of the location transitions and thus the sequences themselves show no logical order (from a common sense knowledge perspective) as one would expect from a daily trajectory and as the one featured in the SDP dataset. The consequence is that since both the SemSimLoc and LowHighLevelLoc approach relate not only one, but many (similar) locations with a certain future location at the same time, these are more sensible in such kind of illogical ordered data.

Finally, Fig. 10.15 and 10.16 shows the results of the SEBP approach for the SDP and the MIT case respectively. Since we apply batch training here, we

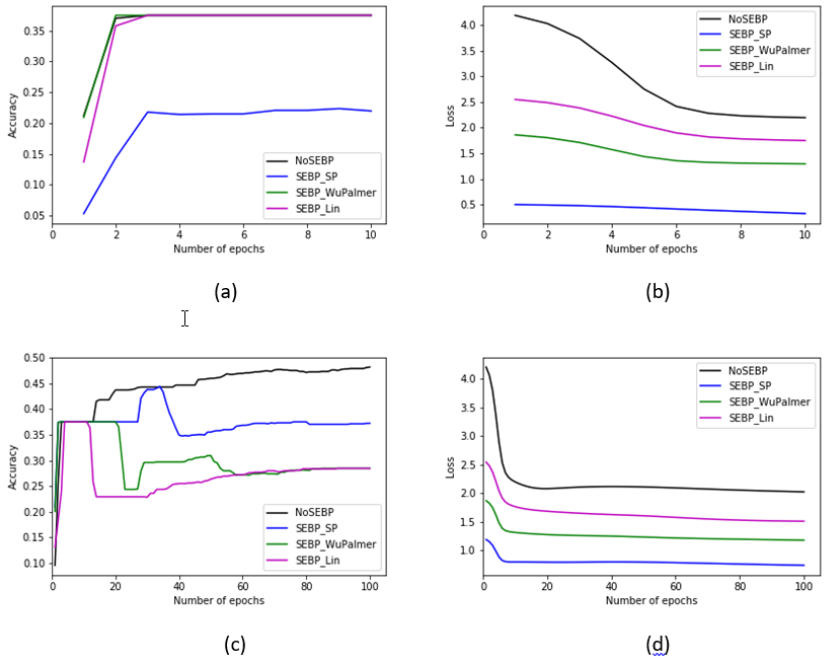

Figure 10.15: SEBP (10 (top) and 100 (bottomt) epochs, SDP).

tested several SEBP variants depending on the way the similarity score flows back into the error signal that is being backpropagated. The results presented in this section refer to the best setup, in which the loss values are being at each step first individually updated by the similarity score (as shown in Eq.

[4] https://snap.stanford.edu/data/loc-gowalla.html

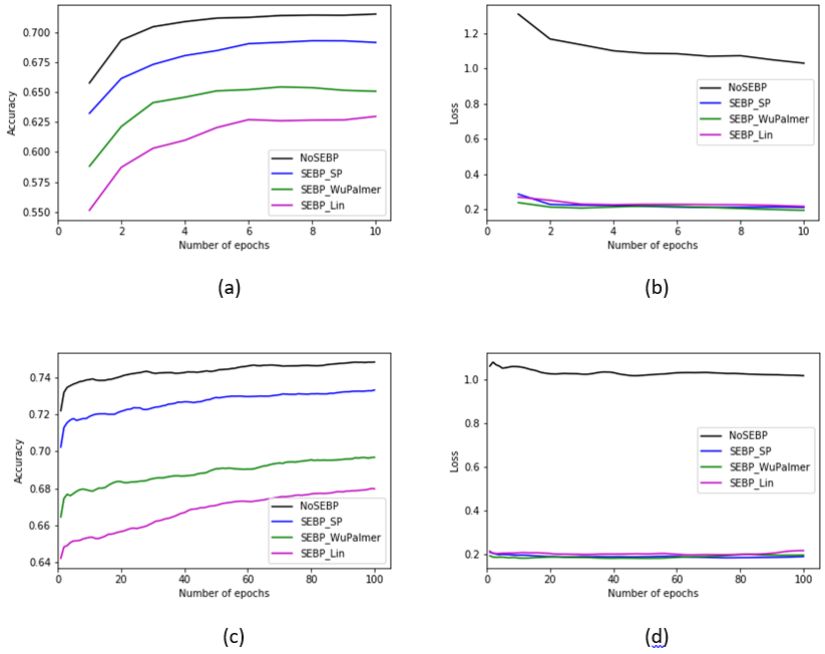

Figure 10.16: SEBP MIT (10 (top)) and 100 (bottom) epochs).

10.6) and then we calculate the overall average over the batch size. The value that is being backpropagated in the network relies on this average. The SEBP layer seems to be provoking an overfitting effect to the model. The training accuracy curves and the loss functions reach relatively fast their final values, while the test accuracy scores remain equal or even under the baseline (see Fig. 10.17). Thus, it seems that SEBP, especially given a small dataset like

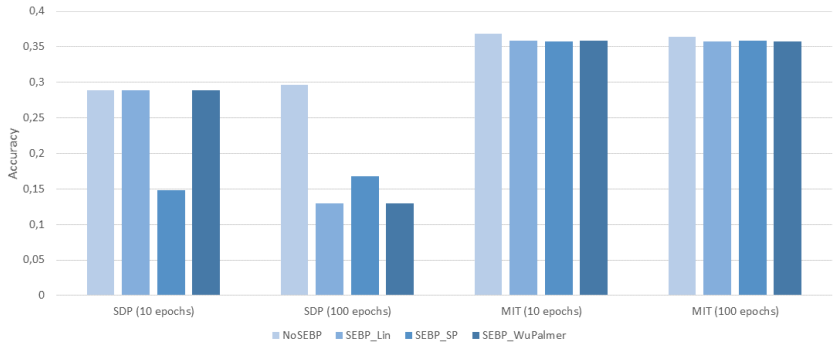

Figure 10.17: Test accuracy: SEBP (10 (top)) and 100 (bottom) epochs, SDP and MIT).

the SDP and at least in this initial form, leads models to fall relatively fast into

some local minima. In order to overcome this issue we experimented shortly by making our model simpler, that is, by reducing the number of LSTM units. This improved the training behaviour of the models significantly, but the test accuracy scores remained still low. Adding a momentum, a regularization term, as we plan to investigate in the future, might help solve this problem. Nevertheless, if we take a look at the loss curves, these are always clearly under the baseline. In the case of the SDP dataset, the Shortest Path similarity based SEBP is characterized by the lowest loss values, while achieving the best results for the 100 epoch case at the same time. In comparison, Lin and Wu and Palmer metrics show the best training and test accuracy for the 10 epochs by reaching the same performance as the baseline. In the MIT dataset case, all similarity measures show both a very similar and stable behaviour with regard to the loss values and a very similar test accuracy.

10.7 Conclusion

Real-world data used for classification and pattern recognition tasks carry a certain semantic load with them. A semantic load, which is not taken into account by most machine learning algorithms. We hypothesized that considering the semantics behind the data can help improve the prediction performance. For this purpose, we propose in this chapter to incorporate an additional semantic layer into the neural network whose role is to take these semantics explicitly into consideration. In particular, we presented 3 different variants of our approach with each differing from the other two based on the network's position at which the layer is applied and the type of semantic knowledge it considers. We evaluated our approach using two real-world datasets. All in all, we could show that our approach helps the models to converge faster to low loss values by keeping the same test performance or even improving it at the same time. A fact that reflects the need of less training examples and a shorter training time. However we could also identify some limitations. These refer mainly to generalization and overfitting issues and the need of regularization terms to overcome them. In the future we plan to further investigate the combination of semantics and machine learning. At the same time, we would like to test other similarity metrics and other ways of incorporating the

semantic knowledge into the network as well. Lastly, we plan to investigate our approach in further different use cases, such as in the image and object recognition domain.

CHAPTER 11

CONCLUSIONS AND OUTLOOK

The role of location awareness and Location Based Services (LBS) has become increasingly important in recent years. Moreover, LBS providers rely their services more and more often on location-prediction models in order to be capable of providing timely and forward-looking solutions to their users. The majority of such location prediction models base on data-driven and pattern-based statistical and machine learning algorithms. Due to their data-driven nature, these algorithms show some limitations, mostly caused by the lack of appropriate training datasets with respect to their size and quality. Recent research utilizes semantic knowledge in order to semantically enrich the available trajectory data aiming among others at overcoming this kind of limitations and thus, supporting in this way the modelling performance of their models. These enriched trajectories reduce the raw GPS trajectories into a number of few significant locations (*semantic locations*) and are referred to as *semantic trajectories*. Beyond supporting location prediction models, semantic trajectories help to obtain a deeper insight into human movement behaviour. Therefore, predicting future semantic locations can be of great importance in a big variety of cases such as for mobility analysis purposes, in the field of the aforementioned Location-Based Services (LBS), in the transport and telecommunication resource management domain, as well as in the field of recommendation systems, to name but a few.

However, there still exists very little research in the field of modelling semantic trajectories and predicting the user's future semantic locations. This can be mainly attributed to the fact that it represents a relative new research field. The aim of this thesis was to fill this gap by exploring a number of modelling methods for predicting the users' future semantic locations. The second aim of the presented thesis was to investigate whether and to what degree the semantic representation level of the trajectories as well as the degree of semantic enrichment may affect the predictive performance of the models. Finally, the presented thesis introduced and evaluated a number of ways for optimizing the training of neural networks by incorporating semantic knowledge using the example of semantic location prediction. The most important findings of the presented work are summarized in Section 11.1 below. Finally, Section 11.2 addresses some potential future work.

11.1 Conclusions

Chapter 3 evaluates the use of Markov Chains for modelling semantic trajectories at two different semantic representation levels using a total of three real-world datasets, a 12-week long single-user dataset, a 8-week long dataset of 21 participants and a 5-week long dataset of 10 participants. It compares the performance of a single-dimensional spatial Markov Chain model that uses only semantic locations as input with the performance of a multi-dimensional model that takes time, day and the activity of the user additionally into consideration. The multi-dimensional model is able to outperform our trajectory mining and prefix-tree based baseline in terms of accuracy, precision, recall and F-Score. However, the results of our study also show that the particular approach is more sensitive towards the size and the quality of the training dataset compared to the vanilla single-dimensional Markov model due to its multiple dimensions in combination with the small size of the available datasets. On the question of whether the semantic level has an impact on the model's performance, it can be shown that trajectories described at higher semantic levels lead generally to a higher accuracy. A possible explanation for this might be that humans' regular movement base mainly on high-level patterns and rules. A regular high-level behaviour could be for instance a visit to a *restaurant* after being at the *gym*, independent of the type of the restaurant (e.g., *Italian*, *Greek*, *Burger joint*). However, another possible explanation is the fact that high-level semantic trajectories contain less classes and thus, from a statistical point of view, it is easier for the Markov model to predict the correct one.

Chapter 4 discusses the use of a context-based semantic similarity analysis measure (*PoVDSSA*) in combination with the aforementioned multi-dimensional Markov Chain model of Chapter 3 in order to overcome the limitations of the latter due primarily to the sparseness of the available data. The method is evaluated using the same 5-week long real-world dataset from Chapter 3. The result is that the PoVDSSA-based approach allows indeed a better performance compared to both the standard Markov approach and the trajectory mining and prefix-tree based baseline of Chapter 3. However, it only shows a very slight improvement against sensitivity to sparseness compared to the Markov model that could again be primarily attributed to the lack of a bigger

dataset.

Chapter 5 aims at finding a way for coping the limitations caused by a sparse training dataset as well. In particular, it evaluates a user-specific Matrix Factorization approach as a means for filling the missing information in the Markov model's location transition matrix. The approach is evaluated using the 9-month long real-world Reality Mining dataset containing annotated data from 100 users and is compared to a standard Markov Chain, a standard matrix factorization and a factorized Markov Chain model. The study has found that our approach could clearly outperform all three baseline systems by obtaining in average 30% higher F-Score and Recall values and an at least 10% higher accuracy than the two latter models. However, despite the good performance of our model relative to our baselines, the absolute scores are relative low. This fact is likely to be related to the nature of our test dataset that lacks to a certain degree of regular and common movement patterns among the participants.

Chapter 6 addresses the use of Artificial Neural Networks on semantic trajectories for predicting future semantic locations. A Feed-Forward (FFNN), a Recurrent (RNN) and a Long Short-Term Memory (LSTM) Network are evaluated on two different real-world datasets, the 3-months long single-user dataset and the Reality Mining dataset mentioned earlier. Similar to Chapter 3, this chapter evaluates the performance of each model at two different semantic representation levels. A Markov Chain model serves as reference in our evaluation. The results show a good average performance of all network types, with the LSTM outperforming the other two at the higher semantic level achieving an average accuracy of 76% and being at the low level together with the simple RNN also at the top. Adding time as an additional feature seemed to be causing a negative effect. Furthermore, compared to the probabilistic Markov model, solely the LSTM could reach slightly higher accuracy values at the higher representation level. At the lower level, the Markov model performs almost 7% better. This reflects partly a higher dependency on the dataset and its properties on the part of the neural networks in contrast to the probabilistic method.

Chapter 7 investigates the performance of Convolutional Neural Networks (CNN) and the impact of embedding layers with regard to predicting future

locations upon semantic trajectories at again two different semantic representation levels. The CNN-based approach is evaluated using the Reality Mining dataset and the FFNN, the RNN and the LSTM network of Chapter 6 serve as baseline. The result of this investigation show that Convolutional networks are capable of outperforming the other network architectures in terms of accuracy by 7-8%, but show certain limitations when it comes to precision and recall, especially at the higher semantic level. This may partly explained due to the nature of CNNs and their application of fixed-size filters, while human daily trajectories are of varied size. The use of an embedding layer seems to be having a minor but still positive effect on the performance of the CNN model with regard to all of our evaluation metrics. Finally, the investigation has also shown the advantages of having user-specific models in contrast to a multi-user model in terms of accuracy. However, at the same time the multi-user model outperforms the average of all the single-user models with respect to recall, precision and F-Score. This inconsistency can be mainly attributed to the fact that a model trained with data coming from different users has a higher chance to finding the relevant locations out of the training dataset and thus is capable of showing a higher recall value.

Chapter 8 evaluates an attention-based Sequence to Sequence (Seq2Seq) learning approach using two different real-world data, the 8-week long dataset used in Chapter 3 and the Reality Mining dataset. At the same time, as in Chapter 7, Chapter 8 compares a multi-user model with the corresponding set of single-user models and evaluates the use of an embedding layer. A vanilla LSTM, a geographic-, semantic-, and temporal pattern mining framework and a Markov Chain model are used as reference. The evaluation showed that the Seq2Seq based approach could outperform both the pattern mining method and the Markov model. In the multi-user case though, while it yields the highest scores on the 8-week long dataset, no particular advantage could be observed against the vanilla LSTM model on the Reality Mining dataset. One could solely refer to the fact that the LSTM model, that achieved the same performance as the Seq2Seq approach, consisted of 128 units compared to the 8 units of the Seq2Seq model, a fact that reflects a higher number of parameters and thus a higher computational effort on part of the LSTM approach. In contrast to the multi-user case, the single-user Seq2Seq models outperform

the LSTM, especially in terms of recall and precision and consequently of F-Score. The use of an embedding layer showed analog to the CNN case a slight improvement with respect to accuracy but led to a minor decrease of the recall and the precision values. Finally, surprisingly, the LSTM could outperform the Seq2Seq model when it comes to predicting long-term dependencies, despite the use of feeding the encoder with sequences of reverse order, a common countermeasure for helping Seq2Seq models with the learning of long-term dependencies.

Chapter 9 builds on the findings of Chapter 4, addressing the degree of semantic enrichment, that is, the use of additional features and their impact on semantic location prediction. The approach presented here extends the dynamic clustering of locations based on contextual information described in Chapter 4 by taking a number of user-specific psychological features additionally into account, namely the user's personality and emotional state. For this purpose, a dynamic user clustering construct referred to as *Location-Specific Cognitive Frame (LSCF)* is introduced and fed subsequently as input into a LSTM-based prediction model. The method is evaluated with the above mentioned 8-week long dataset containing the data of a total of 21 mobile users at two different semantic representation levels as before. The results show a slight improvement (up to 3%) of the performance in terms of F-Score on the part of the LSCF-enhanced model compared to the vanilla LSTM at both semantic levels. Moreover, at the higher level, the LSCF-based model showed an approximately 20% higher precision. The benefit of the additional features with respect to a lower scattering of the predicted values could also be identified compared to a probabilistic Markov Chain model. However, the Markov model was still able to perform almost equally well or even better (at the higher semantic level) than both the LSTM and the LSCF-based approach, a fact that reflects most likely the sensitivity of neural networks to small training datasets. In addition, the additional features have shown to be leading to a shorter training convergence. In this context, Chapter 9 also investigates a semantic similarity analysis based model training approach. It could be shown that this kind of semantic-enhanced training can contribute to a further reduction of the training convergence time, which makes it attractive for the case of large datasets or low computational power.

Finally, motivated by the findings of Chapters 4 and 9, Chapter 10 explores the incorporation of an additional semantic layer into the neural network with the objective to improve both its training efficiency and predictive performance. In this thesis, this is referred to as *Semantic-Enhanced Learning (SEL)*. Three different semantic layer variants are discussed and evaluated using the example of semantic location prediction based on the same two datasets used in the previous chapters. The results confirm the findings in Chapter 9 and show that adding the proposed semantic layer can lead to a faster convergence by keeping or even improving the predictive performance at the same time. Furthermore, it seems that the additional layer contributes to raising the confidence of the model in terms of a much lower loss value. Both the faster convergence and the higher degree of confidence and thus robustness makes the use of an additional semantic layer for both big data and low computational power edge computing cases particular interesting.

11.2 Outlook

This thesis proposes and evaluates a number of approaches for modelling semantic trajectories and predicting upon them. Depending on the approach and due to the lack of appropriate open datasets, some individual results listed in this work rely on a relative small, unbalanced and not representative amount of data (e.g., participants were primarily students). This makes the respective findings less generalizable. A further, broader user study would help assess their representativity.

Chapters 3, 4 and 9 provide evidence that additional context- and user-specific information result in a higher predictive performance. Further research could usefully explore the impact of other, in this thesis ignored factors, such as additional user profile information (age, occupation, etc.), information about the weather and the transportation mode, on the performance of the models. However, the consideration of these additional information could be only examined provided the existence of a bigger dataset. Such a large dataset is necessary to compensate the complexity (number of trained parameters) of the neural network based models that is required to manage the extra input. Given the aforementioned lack of datasets, one possible solution could be the

use of synthetically generated data. These could enable the application of more sophisticated and deeper machine learning models.

Finally, the *Semantic-Enhanced Learning* approach of Chapter 10 shows promising results and lays a solid foundation for further investigation. Incorporating semantic knowledge into the training of neural network models by means of appending an additional semantic layer leads to a more efficient training while improving the predictive performance at the same time. A greater focus on the interplay between data- and knowledge-driven models, that is, on methods for integrating semantic information into the training of machine learning models, could produce interesting findings that account for a significantly more efficient training of the latter and enhance the quality of their estimations.

CHAPTER 12

PUBLICATION LIST OF THE AUTHOR

List of Publications

Antonios Karatzoglou, Michael Beigl: *Semantic-Enhanced Learning (SEL) On Artificial Neural Networks Using the Example of Semantic Location Prediction.* In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019 (under review).

Antonios Karatzoglou: *Evolutionary Optimization of Artificial Neural Networks for Predicting the User's Future Semantic Location.* In: Proceedings of the 20th International Conference on Engineering Applications of Neural Networks (EANN), 2019 (to come).

Antonios Karatzoglou, Jan Ebbing, Phil Ostheimer, Wenlan Hua, Michael Beigl: *Sentient User Modeling Using the Example of Location Prediction.* In: User Modeling and User-Adapted Interaction (UMUAI), 2019 (to come).

Antonios Karatzoglou, Yannick Menny, Michael Beigl: *Identification of Environment- and Context-Specific Key Factors Influencing The Users' Thermal Comfort.* In: Smart Cities, Green Technologies, and Intelligent Transport Systems, Series: Communications in Computer and Information Science, Springer, 2018 (to come).

Antonios Karatzoglou, Julia Anken, Florian Banscher, Lukas Diewald, Michael Beigl: *Searching for Temporal Dependencies in the Privacy Concerns of Location-Based Service Users.* In: Proceedings of the Twelfth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM), Athens, Greece, IARIA, 2018.

Antonios Karatzoglou, Adrian Jablonski, Michael Beigl: *A Seq2Seq Learning Approach for Modeling Semantic Trajectories and Predicting the Next Location.* In: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL), Seattle, USA, ACM, 2018.

Antonios Karatzoglou, Dominik Köhler, Michael Beigl: *Semantic-Enhanced*

Multi-Dimensional Markov Chains on Semantic Trajectories for Predicting Future Locations. In: Sensors 2018, 18, 3582, MDPI, 2018.

Antonios Karatzoglou, Markus Szarvas, Michael Beigl: *Towards an Affective Semantic Trajectory Generator (ASTG)*. **Best Student Paper Award**. In: Proceedings of the 14th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Limassol, Cyprus, October 15-17, 2018, S. 1-10.

Antonios Karatzoglou, Nikolai Schnell, Michael Beigl: *A convolutional Neural Network Approach for Modeling Semantic Trajectories and Predicting Future Locations*. In: Proceedings of the 27th International Conference on Artificial Neural Networks (ICANN), Rhodes, Greece, Springer International Publishing, 2018, S. 61-72.

Antonios Karatzoglou, Julian Janßen, Vethiga Srikanthan, Christof Urbaczek, Michael Beigl: *A Predictive Comfort- and Energy-aware MPC-driven Approach based on a Dynamic PMV Subjectification towards Personalization in an Indoor Climate Control Scenario*. **Nominated for the Best Paper Award**. In: Proceedings of the 7th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS) 2018, Madeira, Portugal: 89-100, 2018.

Antonios Karatzoglou, Dominik Koehler, Michael Beigl: *Purpose-of-Visit-Driven Semantic Similarity Analysis on Semantic Trajectories for Enhancing The Future Location Prediction*. In: Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom) 2018, Athens, Greece, IEEE, 2018: 100-106.

Antonios Karatzoglou, Michael Beigl: *Enhancing the Affective Sensitivity of Location Based Services Using Situation-Person-Dependent Semantic Similarity*. In: Proceedings of the Eleventh International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM) 2017, Barcelona, Spain, 2017, p. 95-100.

Antonios Karatzoglou, Michael Beigl: *Applying Situation-Person-Driven Semantic Similarity On Location-Specific Cognitive Frames For Improv-*

ing the Location Prediction. In: 8th International Conference on Knowledge Engineering and Semantic Web (KESW), Stettin, Poland, 2017, p. 4-5.

Antonios Karatzoglou, Stefan Christian Lamp, Michael Beigl: *Matrix Factorization on Semantic Trajectories for Predicting Future Semantic Locations.* In: Proceedings of the 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Rome, Italy, IEEE, 2017, S. 1-7.

Antonios Karatzoglou, Harun Sentürk, Adrian Jablonski, Michael Beigl: *Applying Artificial Neural Networks on Two-Layer Semantic Trajectories for Predicting the Next Semantic Location.* In: Proceedings of the 26th International Conference on Artificial Neural Networks (ICANN), Alghero, Italy, September 11-14, 2017, Proceedings, Part II, , p. 233-241.

Antonios Karatzoglou, Julian Janßen, Vethiga Srikanthan, Yong Ding, Michael Beigl: *Comfort-efficiency-equilibrium.* In: Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS), Porto, Portugal, 2017: 258-265.

Antonios Karatzoglou, Markus Scholz, Till Riedel, Michael Beigl: *A prototype of an in-situ radio sensing and visualization device.* In: Proceedings of the 9th International Conference on Networked Sensing, INSS 2012, Antwerp, Belgium, June 11-14, 2012.

List of Granted Patents

A Method and System for Identifying at Least One (Contextual) State of a Certain Area or Room (Verfahren und System zum Erfassen mindestens eines Zustands eines Raums). Deutschland, DE 10 2014 211 237.0.

Operating Procedure for Radio Transceivers and a System for Sensing a Certain Area (Betriebsverfahren für Funksendeempfangseinheiten und System zur Erfassung eines Rahmenbereichs), Deutschland, DE 10 2016 221 755.0.

BIBLIOGRAPHY

[AAB+11] Andrienko, Gennady ; Andrienko, Natalia ; Bak, Peter ; Keim, Daniel ; Kisilevich, Slava ; Wrobel, Stefan: A conceptual framework and taxonomy of techniques for analyzing movement. In: *Journal of Visual Languages & Computing* 22 (2011), Nr. 3, S. 213–232

[ABK+07] Alvares, Luis O. ; Bogorny, Vania ; Kuijpers, Bart ; Macedo, Jose Antonio F. ; Moelans, Bart ; Vaisman, Alejandro: A Model for Enriching Trajectories with Semantic Geographical Information. In: *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*. New York, NY, USA : ACM, 2007 (GIS '07). – ISBN 978–1–59593–914–2, 22:1–22:8

[AFGY02] Ayres, Jay ; Flannick, Jason ; Gehrke, Johannes ; Yiu, Tomi: Sequential PAttern Mining Using a Bitmap Representation. In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA : ACM, 2002 (KDD '02). – ISBN 1–58113–567–X, 429–435

[AG14] Adali, Sibel ; Golbeck, Jennifer: Predicting personality with social behavior: a comparative study. In: *Social Network Analysis and Mining* 4 (2014), Nr. 1, S. 159. – ISSN 1869–5469

[Alb10] Albert Au Yeung: *Matrix Factorization: A Simple Tutorial and Implementation in Python.* `http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/`. Version: September 2010

[Alt07] Altman, Rachel M.: Mixed hidden Markov models: an extension of the hidden Markov model to the longitudinal data setting. In: *Journal of the American Statistical Association* 102 (2007), Nr. 477, S. 201–210

[AMSS11] Asahara, Akinori ; Maruyama, Kishiko ; Sato, Akiko ; Seto, Kouichi: Pedestrian-movement prediction based on mixed Markov-chain model. In: *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems* ACM, 2011, S. 25–33

[ARAKA09] Al Ridhawi, I ; Aloqaily, M ; Karmouch, A ; Agoulmine, N: A location-aware user tracking and prediction system. In: *Information Infrastructure Symposium, 2009. GIIS'09. Global* IEEE, 2009, S. 1–8

[AS02] Ashbrook, Daniel ; Starner, Thad: Learning significant locations and predicting user movement with GPS. In: *Wearable Computers, 2002.(ISWC 2002). Proceedings. Sixth International Symposium on* IEEE, 2002, S. 101–108

[AS03] Ashbrook, Daniel ; Starner, Thad: Using GPS to learn significant locations and predict movement across multiple users. In: *Personal and Ubiquitous computing* 7 (2003), Nr. 5, S. 275–286

[BB09] Broekens, Joost ; Brinkman, Willem-Paul: Affectbutton: Towards a standard for dynamic affective user feedback. In: *Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009. 3rd International Conference on* IEEE, 2009, S. 1–8

[BC11] Belegundu, Ashok D. ; Chandrupatla, Tirupathi R.: *Optimization concepts and applications in engineering*. Cambridge University Press, 2011

[BCB14] Bahdanau, Dzmitry ; Cho, Kyunghyun ; Bengio, Yoshua: Neural machine translation by jointly learning to align and translate. In: *arXiv preprint arXiv:1409.0473* (2014)

[BCMR12] Boyd, Stephen ; Cortes, Corinna ; Mohri, Mehryar ; Radovanovic, Ana: Accuracy at the top. In: *Advances in neural information processing systems*, 2012, S. 953–961

[BEJ97] Biesterfeld, Jens ; Ennigrou, Elyes ; Jobmann, Klaus: Neural networks for location prediction in mobile networks. In: *Proc. Intern. Workshop on Applications of Neural Networks to Telecommunications*, 1997, S. 207–214

[Bis06] Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer, 2006 (Information Science and Statistics). `https://books.google.de/books?id=kTNoQgAACAAJ`. – ISBN 9780387310732

[BKV08] Bell, Robert M. ; Koren, Yehuda ; Volinsky, Chris: The Bellkor 2008 Solution to the Netflix Prize, 2008

[BL94] Bradley, Margaret M. ; Lang, Peter J.: Measuring emotion: the self-assessment manikin and the semantic differential. In: *Journal of behavior therapy and experimental psychiatry* 25 (1994), Nr. 1, S. 49–59

[BMZ11] Bollen, Johan ; Mao, Huina ; Zeng, Xiaojun: Twitter mood predicts the stock market. In: *Journal of computational science* 2 (2011), Nr. 1, S. 1–8

[BNJ03] Blei, David M. ; Ng, Andrew Y. ; Jordan, Michael I.: Latent dirichlet allocation. In: *Journal of machine Learning research* 3 (2003), Nr. Jan, S. 993–1022

[Bor93] In: Borkenau, Peter: *To Predict Some of the People More of the Time*. Boston, MA : Springer US, 1993. – ISBN 978–1–4899–2311–0, S. 237–249

[BU05] Bart, Evgeniy ; Ullman, Shimon: Cross-generalization: Learning novel classes from a single example by feature replacement. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* Bd. 1 IEEE, 2005, S. 672–679

[Bus77] Buss, Allan R.: The trait-situation controversy and the concept of interaction. In: *Personality and Social Psychology Bulletin* 3 (1977), Nr. 2, S. 196–201

[CBT+13] Csáji, Balázs Cs ; Browet, Arnaud ; Traag, Vincent A. ; Delvenne, Jean-Charles ; Huens, Etienne ; Van Dooren, Paul ; Smoreda, Zbigniew ; Blondel, Vincent D.: Exploring the mobility of mobile phone users. In: *Physica A: statistical mechanics and its applications* 392 (2013), Nr. 6, S. 1459–1473

[CK94] Cook, Ronny ; Kay, Judy: The justified user model: a viewable, explained user model. In: *In Proceedings of the Fourth International Conference on User Modeling* Citeseer, 1994

[CKT17] Chauhan, Arun ; Kummamuru, Krishna ; Toshniwal, Durga: Prediction of places of visit using tweets. In: *Knowledge and Information Systems* 50 (2017), Nr. 1, S. 145–166

[CTT16] Chauhan, A. ; Toshniwal, D. ; Tejwani, R.: Predicting future place of visit using user's personality profile. In: *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, 2016, S. 427–432

[CWB+11] Collobert, Ronan ; Weston, Jason ; Bottou, Léon ; Karlen, Michael ; Kavukcuoglu, Koray ; Kuksa, Pavel: Natural language processing (almost) from scratch. In: *Journal of Machine Learning Research* 12 (2011), Nr. Aug, S. 2493–2537

[CYLK13] Cheng, Chen ; Yang, Haiqin ; Lyu, Michael R. ; King, Irwin: Where You Like to Go Next: Successive Point-of-Interest Recommendation. In: *IJCAI* Bd. 13, 2013, S. 2605–2611

[Dey01] Dey, Anind K.: Understanding and using context. In: *Personal and ubiquitous computing* 5 (2001), Nr. 1, S. 4–7

[Dod16] Dodge, Somayeh: From observation to prediction: the trajectory of movement research in GIScience. In: *Advancing geographic information science: the past and next twenty years*. GSDI Association Press, 2016

[EKK12] Etter, Vincent ; Kafsi, Mohamed ; Kazemi, Ehsan: Been There, Done That: What Your Mobility Traces Reveal about Your Behavior. In: *Proc. MDC by Nokia Works. 10th PerCom*, 2012

[EKS+96] Ester, Martin ; Kriegel, Hans-Peter ; Sander, Jörg ; Xu, Xiaowei u. a.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Kdd* Bd. 96, 1996, S. 226–231

[Enc08] Encarnação, José Luis: Ambient Intelligence–The New Paradigm for Computer Science and for Information Technology (Ambient Intelligence–Das neue Paradigma der Informatik und der Informationstechnologie). In: *IT-Information Technology* 50 (2008), Nr. 1, S. 5–6

[EP06] Eagle, Nathan ; Pentland, Alex S.: Reality mining: sensing complex social systems. In: *Personal and ubiquitous computing* 10 (2006), Nr. 4, S. 255–268

[ESP06] Eagle, Nathan ; (Sandy) Pentland, Alex: Reality Mining: Sensing Complex Social Systems. In: *Personal Ubiquitous Comput.* 10 (2006), März, Nr. 4, 255–268. http://dx.doi.org/10.1007/s00779-005-0046-3. – DOI 10.1007/s00779–005–0046–3. – ISSN 1617–4909

[FFFP06] Fei-Fei, Li ; Fergus, Rob ; Perona, Pietro: One-shot learning of object categories. In: *IEEE transactions on pattern analysis and machine intelligence* 28 (2006), Nr. 4, S. 594–611

[Fle01] Fleeson, William: Toward a structure-and process-integrated view of personality: Traits as density distributions of states. In: *Journal of personality and social psychology* 80 (2001), Nr. 6, S. 1011

[Fou] Foursquare: *Foursquare Venue Categories*. `https://developer.foursquare.com/docs/resources/categories`, . – Accessed: 2018-04-06

[GBH09] Go, Alec ; Bhayani, Richa ; Huang, Lei: Twitter sentiment classification using distant supervision. In: *CS224N Project Report, Stanford* 1 (2009), Nr. 12

[GHB08] Gonzalez, Marta C. ; Hidalgo, Cesar A. ; Barabasi, Albert-Laszlo: Understanding individual human mobility patterns. In: *nature* 453 (2008), Nr. 7196, S. 779

[GKPC12] Gambs, Sébastien ; Killijian, Marc-Olivier ; Prado Cortez, Miguel N.: Next place prediction using mobility markov chains. In: *Proceedings of the First Workshop on Measurement, Privacy, and Mobility* ACM, 2012, S. 3

[Gol93] Goldberg, Lewis R.: The structure of phenotypic personality traits. In: *American psychologist* 48 (1993), Nr. 1, S. 26

[GP09] In: Gangemi, Aldo ; Presutti, Valentina: *Ontology Design Patterns*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – ISBN 978–3–540–92673–3, 221–243

[Gru95] Gruber, Thomas R.: Toward principles for the design of ontologies used for knowledge sharing. In: *International journal of human-computer studies* 43 (1995), Nr. 5-6, S. 907–928

[GTL12] Gao, Huiji ; Tang, Jiliang ; Liu, Huan: Mobile location prediction in spatio-temporal context. In: *Nokia mobile data challenge workshop* Bd. 41, 2012, S. 44

[GZZ+17] Gao, Qiang ; Zhou, Fan ; Zhang, Kunpeng ; Trajcevski, Goce ; Luo, Xucheng ; Zhang, Fengli: Identifying human mobility via trajectory embeddings. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence* AAAI Press, 2017, S. 1689–1695

[Häg70] Hägerstraand, Torsten: What about people in regional science? In: *Papers in regional science* 24 (1970), Nr. 1, S. 7–24

[HAM17] Hasan, Md Al M. ; Ahmad, Shamim ; Molla, Md Khademul I.: Protein subcellular localization prediction using multiple kernel learning based support vector machine. In: *Molecular BioSystems* 13 (2017), Nr. 4, S. 785–795

[HLK07] Hainš, Violeta V. ; Lovrenčić, Sandra ; Kirinić, Valentina: Personality Model Representation using Ontology. In: *DAAAM International Scientific Book 2007* 7 (2007), S. 423–434

[HNFB10] Heine, Bernd ; Narrog, Heiko ; Fillmore, Charles J. ; Baker, Collin: *A Frames Approach to Semantic Analysis*. //www.oxfordhandbooks.com/10.1093/oxfordhb/9780199544004.001.0001/oxfordhb-9780199544004-e-013. Version: 2010

[Hoc98] Hochreiter, Sepp: The vanishing gradient problem during learning recurrent neural nets and problem solutions. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6 (1998), Nr. 02, S. 107–116

[HRWL84] Hayes-Roth, F. ; Waterman, D. ; Lenat, D.: *Building expert systems*. Addison-Wesley,Reading, MA, 1984

[HS97a] Hochreiter, Sepp ; Schmidhuber, Jürgen: Long short-term memory. In: *Neural computation* 9 (1997), Nr. 8, S. 1735–1780

[HS97b] Hochreiter, Sepp ; Schmidhuber, Jürgen: LSTM can solve hard long time lag problems. In: *Advances in neural information processing systems*, 1997, S. 473–479

[HSB+05] Heckmann, Dominik ; Schwartz, Tim ; Brandherm, Boris ; Schmitz, Michael ; Wilamowitz-Moellendorff, Margeritta von: Gumo – The General User Model Ontology. In: Ardissono, Liliana (Hrsg.) ; Brna, Paul (Hrsg.) ; Mitrovic, Antonija (Hrsg.): *User Modeling 2005*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. – ISBN 978–3–540–31878–1, S. 428–432

[HSM+00] Hahnloser, Richard H. ; Sarpeshkar, Rahul ; Mahowald, Misha A. ; Douglas, Rodney J. ; Seung, H S.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. In: *Nature* 405 (2000), Nr. 6789, S. 947

[HVD15] Hinton, Geoffrey ; Vinyals, Oriol ; Dean, Jeff: Distilling the knowledge in a neural network. In: *arXiv preprint arXiv:1503.02531* (2015)

[Inc] Inc., Google: *Places API Web Service*. `https://developers.google.com/places/web-service/search`

[Jac12] Jaccard, Paul: The distribution of the flora in the alpine zone. 1. In: *New phytologist* 11 (1912), Nr. 2, S. 37–50

[Jac74] Jaccard, James J.: Predicting social behavior from personality traits. In: *Journal of Research in Personality* 7 (1974), Nr. 4, S. 358–367

[Kas04] Kassin, S.M.: *Psychology*. Pearson/Prentice Hall, 2004. – ISBN 9780130496416

[Kay94] Kay, Judy: The um toolkit for cooperative user modelling. In: *User Modeling and User-Adapted Interaction* 4 (1994), Nr. 3, S. 149–196

[KB14] Kingma, Diederik P. ; Ba, Jimmy: Adam: A method for stochastic optimization. In: *arXiv preprint arXiv:1412.6980* (2014)

[KB17] Karatzoglou, Antonios ; Beigl, Michael: Enhancing the Affective Sensitivity of Location Based Services Using Situation-Person-Dependent Semantic Similarity. In: *The Eleventh International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2017)* IARIA, 2017, S. 95–100

[KB19] Karatzoglou, Antonios ; Beigl, Michael: Semantic-Enhanced Learning (SEL) on Artificial Neural Networks Using the Example of Semantic Location Prediction. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* ACM, 2019, S. 01

[KBV09] Koren, Yehuda ; Bell, Robert ; Volinsky, Chris: Matrix Factorization Techniques for Recommender Systems. In: *Computer* 42 (2009), August, Nr. 8, 30–37. http://dx.doi.org/10.1109/MC.2009.263. – DOI 10.1109/MC.2009.263. – ISSN 0018–9162

[KGB14] Kalchbrenner, Nal ; Grefenstette, Edward ; Blunsom, Phil: A convolutional neural network for modelling sentences. In: *arXiv preprint arXiv:1404.2188* (2014)

[KGR+08] Körner, Annett ; Geyer, Michael ; Roth, Marcus ; Drapeau, Martin ; Schmutzer, Gabriele ; Albani, Cornelia ; Schumann, Siegfried ; Brähler, Elmar: Persönlichkeitsdiagnostik mit dem NEO-Fünf-Faktoren-Inventar: Die 30-Item-Kurzversion (NEO-FFI-30). In: *PPmP-Psychotherapie· Psychosomatik· Medizinische Psychologie* 58 (2008), Nr. 06, S. 238–245

[Kim14] Kim, Yoon: Convolutional neural networks for sentence classification. In: *arXiv preprint arXiv:1408.5882* (2014)

[KJB18] Karatzoglou, Antonios ; Jablonski, Adrian ; Beigl, Michael: A Seq2seq Learning Approach for Modeling Semantic

Trajectories and Predicting the Next Location. In: *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* ACM, 2018, S. –

[KKB18a] Karatzoglou, Antonios ; Köhler, Dominik ; Beigl, Michael: Semantic-Enhanced Multi-Dimensional Markov Chains on Semantic Trajectories for Predicting Future Locations. In: *Sensors* 18 (2018), Nr. 10. http://dx.doi.org/10.3390/s18103582. – DOI 10.3390/s18103582. – ISSN 1424–8220

[KKB18b] Karatzoglou, Antonios ; Köhler, Dominik ; Beigl, Michael: Purpose-of-Visit-Driven Semantic Similarity Analysis on Semantic Trajectories for Enhancing The Future Location Prediction. In: *International Conference on Pervasive Computing and Communications Workshops (PerCom)* IEEE, 2018

[KKS+04] Kahneman, Daniel ; Krueger, Alan B. ; Schkade, David A. ; Schwarz, Norbert ; Stone, Arthur A.: A survey method for characterizing daily life experience: The day reconstruction method. In: *Science* 306 (2004), Nr. 5702, S. 1776–1780

[KKS16] Kim, Seung Y. ; Koo, Hoon J. ; Song, Ha Y.: A study on influence of human personality to location selection. In: *Journal of Ambient Intelligence and Humanized Computing* 7 (2016), Nr. 2, S. 267–285

[KKS17] Kim, Seung Y. ; Koo, Hoon J. ; Song, Ha Y.: A study on estimation of human personality from location visiting preference. In: *Journal of Ambient Intelligence and Humanized Computing* (2017), S. 1–14

[KKST15] In: Krishnamurthy, Revathy ; Kapanipathi, Pavan ; Sheth, Amit P. ; Thirunarayan, Krishnaprasad: *Knowledge Enabled Approach to Predict the Location of Twitter Users*. Cham : Springer International Publishing, 2015. – ISBN 978–3–319–18818–8, S. 187–201

[KLB17] Karatzoglou, Antonios ; Lamp, Stefan C. ; Beigl, Michael: Matrix factorization on semantic trajectories for predicting future semantic locations. In: *Wireless and Mobile Computing, Networking and Communications (WiMob),* IEEE, 2017, S. 1–7

[KLLE13] Kim, Seungyeon ; Li, Fuxin ; Lebanon, Guy ; Essa, Irfan: Beyond sentiment: The manifold of human emotions. In: *Artificial Intelligence and Statistics*, 2013, S. 360–369

[KS14] Kim, Seung Y. ; Song, Ha Y.: Predicting human location based on human personality. In: *International Conference on Next Generation Wired/Wireless Networking* Springer, 2014, S. 70–81

[KS17] Kim, Dong Y. ; Song, Ha Y.: Method of predicting human mobility patterns using deep learning. In: *Neurocomputing* (2017)

[KSB18a] Karatzoglou, Antonios ; Schnell, Nikolai ; Beigl, Michael: A Convolutional Neural Network Approach for Modeling Semantic Trajectories and Predicting Future Locations. In: Kůrková, Věra (Hrsg.) ; Manolopoulos, Yannis (Hrsg.) ; Hammer, Barbara (Hrsg.) ; Iliadis, Lazaros (Hrsg.) ; Maglogiannis, Ilias (Hrsg.): *Artificial Neural Networks and Machine Learning – ICANN 2018*. Cham : Springer International Publishing, 2018. – ISBN 978–3–030–01418–6, S. 61–72

[KSB18b] Karatzoglou, Antonios ; Szarvas, Markus ; Beigl, Michael: Towards an Affective Semantic Trajectory Generator (ASTG). In: *14th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob 2018, Limassol, Cyprus, October 15-17, 2018*, 2018, S. 1–10

[KSJB17] Karatzoglou, Antonios ; Sentürk, Harun ; Jablonski, Adrian ; Beigl, Michael: Applying Artificial Neural Networks on Two-Layer Semantic Trajectories for Predicting the Next Semantic Location. In: *International Conference on Artificial Neural Networks* Springer, 2017, S. 233–241

[KSK12] Kim, Seungyeon ; Song, Ha Y. ; Koo, Hoon J.: Probabilistically predicting location of human with psychological factors. In: *Proc. ECCS* (2012)

[LB+95] LeCun, Yann ; Bengio, Yoshua u. a.: Convolutional networks for images, speech, and time series. In: *The handbook of brain theory and neural networks* 3361 (1995), Nr. 10, S. 1995

[LGPA+12] Laurila, J. K. ; Gatica-Perez, Daniel ; Aad, I. ; J., Blom ; Bornet, Olivier ; Do, Trinh-Minh-Tri ; Dousse, O. ; Eberle, J. ; Miettinen, M.: The Mobile Data Challenge: Big Data for Mobile Computing Research. In: *Pervasive Computing*, 2012

[Lin98] Lin, Dekang: An information-theoretic definition of similarity. In: *Icml* Bd. 98 Citeseer, 1998, S. 296–304

[LJJ12] Long, Xuelian ; Jin, Lei ; Joshi, James: Exploring Trajectory-driven Local Geographic Topics in Foursquare. In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. New York, NY, USA : ACM, 2012 (UbiComp '12). – ISBN 978–1–4503–1224–0, S. 927–934

[LKF10] LeCun, Yann ; Kavukcuoglu, Koray ; Farabet, Clément: Convolutional networks and applications in vision. In: *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on* IEEE, 2010, S. 253–256

[LLW16] Lv, Jianming ; Li, Qing ; Wang, Xintong: T-CONV: A Convolutional Neural Network For Multi-scale Taxi Trajectory Prediction. In: *arXiv preprint arXiv:1611.07635* (2016)

[LOC15] Likavec, Silvia ; Osborne, Francesco ; Cena, Federica: Property-based Semantic Similarity and Relatedness for Improving Recommendation Accuracy and Diversity. In: *Int. J. Semant. Web Inf. Syst.* 11 (2015), Oktober, Nr. 4, S. 1–40. – ISSN 1552–6283

[LPM15] Luong, Minh-Thang ; Pham, Hieu ; Manning, Christopher D.: Effective Approaches to Attention-based Neural Machine Translation. In: *CoRR* abs/1508.04025 (2015). http://arxiv.org/abs/1508.04025

[LSSM08] Lee, Wei-Nchih ; Shah, Nigam ; Sundlass, Karanjot ; Musen, Mark: Comparison of Ontology-based Semantic-Similarity Measures. 2008 (2008), 02, S. 384–8

[LSST+02] Lodhi, Huma ; Saunders, Craig ; Shawe-Taylor, John ; Cristianini, Nello ; Watkins, Chris: Text classification using string kernels. In: *Journal of Machine Learning Research* 2 (2002), Nr. Feb, S. 419–444

[LW71] Levandowsky, Michael ; Winter, David: Distance between sets [5]. In: *Nature* 234 (1971), 12, Nr. 5323, S. 34–35. http://dx.doi.org/10.1038/234034a0. – DOI 10.1038/234034a0. – ISSN 0028–0836

[Mat18] Mathworks: *Convolutional Neural Network*. https://www.mathworks.com/discovery/convolutional-neural-network.html, 2018. – [Online; accessed 19-February-2018]

[ME09] Mabroukeh, Nizar R. ; Ezeife, Christie I.: Using domain ontology for semantic web usage mining and next page prediction. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, 2009, S. 1677–1680

[MHG13] Meng, L ; Huang, R ; Gu, J: A Review of Semantic Similarity Measures in WordNet. In: *International Journal of Hybrid Information Technology* 6 (2013)

[Min74] Minsky, Marvin: A Framework for Representing Knowledge. Cambridge, MA, USA : Massachusetts Institute of Technology, 1974. – Forschungsbericht

[Min75] Minsky's Frame System Theory. In: *Proceedings of the 1975 Workshop on Theoretical Issues in Natural Language Processing.*

Stroudsburg, PA, USA : Association for Computational Linguistics, 1975 (TINLAP '75), 104–116

[MLD92] McNair, Douglas M. ; Lorr, Maurice ; Droppleman, Leo F.: *Profile of mood states: manual*. Edits, 1992

[MRM12] Mathew, Wesley ; Raposo, Ruben ; Martins, Bruno: Predicting future locations with hidden Markov models. In: *Proceedings of the 2012 ACM conference on ubiquitous computing* ACM, 2012, S. 911–918

[MWBV12] Malki, Jamal ; Wannous, Rouaa ; Bouju, Alain ; Vincent, Cécile: Temporal reasoning in trajectories using an ontological modelling approach. In: *Control and Cybernetics* 41 (2012)

[Nas07] Nasrabadi, Nasser M.: Pattern recognition and machine learning. In: *Journal of electronic imaging* 16 (2007), Nr. 4, S. 049901

[NGR+08] Nathan, Ran ; Getz, Wayne M. ; Revilla, Eloy ; Holyoak, Marcel ; Kadmon, Ronen ; Saltz, David ; Smouse, Peter E.: A movement ecology paradigm for unifying organismal movement research. In: *Proceedings of the National Academy of Sciences* 105 (2008), Nr. 49, S. 19052–19059

[NR75] Negoita, C.V. ; Ralescu, D.A.: *Applications of Fuzzy Sets to Systems Analysis*. Springer, 1975

[NVH14] Nguyen, Quang ; Valizadegan, Hamed ; Hauskrecht, Milos: Learning classification models with soft-label information. In: *Journal of the American Medical Informatics Association* 21 (2014), Nr. 3, S. 501–508

[oxf16] *Oxford English Dictionary*. `https://en.oxforddictionaries.com/definition/knowledge`. Version: 2016

[Oxf17a] *Oxford Dictionary. Definition of "Semantics"*. 2017. – `https://en.oxforddictionaries.com/definition/semantics`

[oxf17b] *Oxford English Dictionary*. https://en.oxforddictionaries.com/definition/similar. Version: 2017

[PBTU06] Petzold, Jan ; Bagci, Faruk ; Trumler, Wolfgang ; Ungerer, Theo: Comparison of Different Methods for Next Location Prediction. In: *Proc.12th International Conference on Parallel Processing*. Berlin, Heidelberg : Springer-Verlag, 2006 (Euro-Par'06). – ISBN 3–540–37783–2, 978–3–540–37783–2, 909–918

[PL99] Pentland, Alex ; Liu, Andrew: Modeling and prediction of human behavior. In: *Neural computation* 11 (1999), Nr. 1, S. 229–242

[Plu00] Plummer, E: Time series forecasting with feed-forward neural networks: Guidelines and limitations. In: *Neural Networks* 1 (2000), S. 1

[Pro] Protege: *Protege Ontology Editor*. https://protege.stanford.edu/, . – Accessed: 2018-04-06

[PSR+15] Pappalardo, Luca ; Simini, Filippo ; Rinzivillo, Salvatore ; Pedreschi, Dino ; Giannotti, Fosca ; Barabási, Albert-László: Returners and explorers dichotomy in human mobility. In: *Nature communications* 6 (2015), S. 8166

[PVG+11] Pedregosa, F. ; Varoquaux, G. ; Gramfort, A. ; Michel, V. ; Thirion, B. ; Grisel, O. ; Blondel, M. ; Prettenhofer, P. ; Weiss, R. ; Dubourg, V. ; Vanderplas, J. ; Passos, A. ; Cournapeau, D. ; Brucher, M. ; Perrot, M. ; Duchesnay, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[R+77] Reddy, D R. u. a.: *Speech Understanding Systems: A Summary of Results of the Five-Year Research Effort. Department of Computer Science*. 1977

[RAKA09] Ridhawi, I. A. ; Aloqaily, M. ; Karmouch, A. ; Agoulmine, N.: A location-aware user tracking and prediction system. In: *2009 Global Information Infrastructure Symposium*, 2009. – ISSN 2150–3281, S. 1–8

[Rel76] Relph, EC: Place and placelessness/[by] E. Relph. In: *Pion, London* 10 (1976), S. 156

[RFGST09] Rendle, Steffen ; Freudenthaler, Christoph ; Gantner, Zeno ; Schmidt-Thieme, Lars: BPR: Bayesian personalized ranking from implicit feedback. In: *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence* AUAI Press, 2009, S. 452–461

[RFST10] Rendle, Steffen ; Freudenthaler, Christoph ; Schmidt-Thieme, Lars: Factorizing Personalized Markov Chains for Next-basket Recommendation. In: *Proceedings of the 19th International Conference on World Wide Web*. New York, NY, USA : ACM, 2010 (WWW '10). – ISBN 978–1–60558–799–8, 811–820

[RGJ+13] Rentfrow, Peter J. ; Gosling, Samuel D. ; Jokela, Markus ; Stillwell, David J. ; Kosinski, Michal ; Potter, Jeff: Divided we stand: Three psychological regions of the United States and their political, economic, social, and health correlates. In: *Journal of personality and social psychology* 105 (2013), Nr. 6, S. 996

[RHW85] Rumelhart, David E. ; Hinton, Geoffrey E. ; Williams, Ronald J.: Learning internal representations by error propagation / California Univ San Diego La Jolla Inst for Cognitive Science. 1985. – Forschungsbericht

[RMBB89] Rada, Roy ; Mili, Hafedh ; Bicknell, Ellen ; Blettner, Maria: Development and application of a metric on semantic nets. In: *IEEE transactions on systems, man, and cybernetics* 19 (1989), Nr. 1, S. 17–30

[RRKN11] Ridhawi, Y. A. ; Ridhawi, I. A. ; Karmouch, A. ; Nayak, A.: A context-aware and location prediction framework for dynamic environments. In: *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2011. – ISSN 2160–4886, S. 172–179

[Rud16] Ruder, Sebastian: An overview of gradient descent optimization algorithms. In: *arXiv preprint arXiv:1609.04747* (2016)

[San19] *San Francisco photo from above.* 2019. – `https://commons.wikimedia.org/wiki/File:San_Francisco_downtown.jpg`

[SBKK05] Samaan, N. ; Benmammar, B. ; Krief, F. ; Karmouch, A.: Prediction-based advanced resource reservation in mobile environments. In: *Canadian Conference on Electrical and Computer Engineering, 2005.*, 2005. – ISSN 0840–7789, S. 1411–1414

[SBV+11] Siegert, Ingo ; Böck, Ronald ; Vlasenko, Bogdan ; Philippou-Hübner, David ; Wendemuth, Andreas: Appropriate emotional labelling of non-acted speech using basic emotions, geneva emotion wheel and self assessment manikins. In: *Multimedia and Expo (ICME), 2011 IEEE International Conference on* IEEE, 2011, S. 1–6

[SG14] Santos, Cicero dos ; Gatti, Maira: Deep convolutional neural networks for sentiment analysis of short texts. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, S. 69–78

[Sha92] Shafer, Glenn: Dempster-shafer theory. In: *Encyclopedia of artificial intelligence* (1992), S. 330–331

[SHK+14] Srivastava, Nitish ; Hinton, Geoffrey ; Krizhevsky, Alex ; Sutskever, Ilya ; Salakhutdinov, Ruslan: Dropout: a simple way to prevent neural networks from overfitting. In: *The Journal of Machine Learning Research* 15 (2014), Nr. 1, S. 1929–1958

[SK05a] Samaan, Nancy ; Karmouch, Ahmed: A Mobility Prediction Architecture Based on Contextual Knowledge and Spatial Conceptual Maps. In: *IEEE Transactions on Mobile Computing* 4 (2005), November, Nr. 6, S. 537–551. – ISSN 1536–1233

[SK05b] Samaan, Nancy ; Karmouch, Ahmed: A Mobility Prediction Architecture Based on Contextual Knowledge and Spatial Conceptual Maps. In: *IEEE Transactions on Mobile Computing* 4 (2005), November, Nr. 6, 537–551. http://dx.doi.org/10.1109/TMC.2005.74. – DOI 10.1109/TMC.2005.74. – ISSN 1536–1233

[SK18] Song, Ha Y. ; Kang, Hwa B.: Analysis of Relationship Between Personality and Favorite Places with Poisson Regression Analysis. In: *ITM Web of Conferences* Bd. 16 EDP Sciences, 2018, S. 02001

[SKJH04] Song, Libo ; Kotz, David ; Jain, Ravi ; He, Xiaoning: Evaluating location predictors with extensive Wi-Fi mobility data. In: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies* Bd. 2 IEEE, 2004, S. 1414–1424

[SKK04] Samaan, N. ; Karmouch, A. ; Kheddouci, H.: Mobility prediction based service location and delivery. In: *Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No.04CH37513)* Bd. 4, 2004. – ISSN 0840–7789, S. 2307–2310 Vol.4

[SKS16] Song, Xuan ; Kanasugi, Hiroshi ; Shibasaki, Ryosuke: Deeptransport: Prediction and Simulation of Human Mobility and Transportation Mode at a Citywide Level. In: *Proc. 25th International Joint Conference on Artificial Intelligence*, 2016. – ISBN 978–1–57735–770–4, S. 2618–2624

[SL09] Sokolova, Marina ; Lapalme, Guy: A systematic analysis of performance measures for classification tasks. In: *Information Processing & Management* 45 (2009), Nr. 4, S. 427–437

[SLA+12] Staiano, Jacopo ; Lepri, Bruno ; Aharony, Nadav ; Pianesi, Fabio ; Sebe, Nicu ; Pentland, Alex: Friends Don'T Lie: Inferring Personality Traits from Social Network Structure. In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. New York, NY, USA : ACM, 2012 (UbiComp '12). – ISBN 978–1–4503–1224–0, S. 321–330

[SPD+08] Spaccapietra, Stefano ; Parent, Christine ; Damiani, Maria L. ; Macedo, Jose A. ; Porto, Fabio ; Vangenot, Christelle: A Conceptual View on Trajectories. In: *Data Knowl. Eng.* 65 (2008), April, Nr. 1, 126–146. http://dx.doi.org/10.1016/j.datak.2007.10.008. – DOI 10.1016/j.datak.2007.10.008. – ISSN 0169–023X

[SPW+13] Socher, Richard ; Perelygin, Alex ; Wu, Jean ; Chuang, Jason ; Manning, Christopher D. ; Ng, Andrew ; Potts, Christopher: Recursive deep models for semantic compositionality over a sentiment treebank. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, S. 1631–1642

[SQBB10] Song, Chaoming ; Qu, Zehui ; Blumm, Nicholas ; Barabási, Albert-László: Limits of predictability in human mobility. In: *Science* 327 (2010), Nr. 5968, S. 1018–1021

[SSS12] Sacharin, Vera ; Schlegel, Katja ; Scherer, Klaus R.: Geneva emotion wheel rating study. (2012)

[SVL14] Sutskever, Ilya ; Vinyals, Oriol ; Le, Quoc V.: Sequence to sequence learning with neural networks. In: *Advances in neural information processing systems*, 2014, S. 3104–3112

[Tve77] Tversky, Amos: Features of similarity. In: *Psychological review* 84 (1977), Nr. 4, S. 327

[TXMO16] Tang, Yujin ; Xu, Jianfeng ; Matsumoto, Kazunori ; Ono, Chihiro: Sequence-to-sequence model with attention for time se-

ries classification. In: *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on* IEEE, 2016, S. 503–510

[VBL+16] Vinyals, Oriol ; Blundell, Charles ; Lillicrap, Tim ; Wierstra, Daan u. a.: Matching networks for one shot learning. In: *Advances in Neural Information Processing Systems*, 2016, S. 3630–3638

[VGPU04] Vintan, Lucian ; Gellert, Arpad ; Petzold, Jan ; Ungerer, Theo: Person movement prediction using neural nets. In: *1st Works. on Mod. and Retr. of Context* Bd. 114, 2004, S. 1–12

[VJ15] Vukovic, Marin ; Jevtic, Dragan: Agent-based Movement Analysis and Location Prediction in Cellular Networks. In: *Procedia Computer Science* 60 (2015), S. 517–526

[W3C18] W3C: *Defining N-ary Relations on the Semantic Web*. `http://www.w3.org/TR/swbp-n-aryRelations/`. Version: 2018

[WC17] Wang, Tian ; Cho, Kyunghyun: Attention-based Mixture Density Recurrent Networks for History-based Recommendation. In: *arXiv preprint arXiv:1709.07545* (2017)

[Wei91] Weiser, Mark: The Computer for the 21 st Century. In: *Scientific american* 265 (1991), Nr. 3, S. 94–105

[WLB11] Wong, Wilson ; Liu, Wei ; Bennamoun, Mohammed: Ontology Learning from Text: A Look Back and into the Future. In: *ACM Computing Surveys - CSUR* 44 (2011), 01, S. 1–36. `http://dx.doi.org/10.1145/2333112.2333115`. – DOI 10.1145/2333112.2333115

[WMBV13a] Wannous, Rouaa ; Malki, Jamal ; Bouju, Alain ; Vincent, Cécile: Modelling mobile object activities based on trajectory ontology rules considering spatial relationship rules. In: *Modeling approaches and algorithms for advanced computer applications*. Springer, 2013, S. 249–258

[WMBV13b] Wannous, Rouaa ; Malki, Jamal ; Bouju, Alain ; Vincent, Cécile: Time integration in semantic trajectories using an ontological modelling approach. In: *New Trends in Databases and Information Systems*. Springer, 2013, S. 187–198

[WMBV16] Wannous, Rouaa ; Malki, Jamal ; Bouju, Alain ; Vincent, Cécile: Trajectory ontology inference considering domain and temporal dimensions?Application to marine mammals. In: *Future Generation Computer Systems* (2016)

[WMS04] Welty, Chris ; McGuinness, Deborah L. ; Smith, Michael K.: Owl web ontology language guide. In: *W3C recommendation, W3C (February 2004) http://www. w3. org/TR/2004/REC-owl-guide-20040210* (2004)

[WP94] Wu, Zhibiao ; Palmer, Martha: Verbs semantics and lexical selection. In: *Proceedings of the 32nd annual meeting on Association for Computational Linguistics* Association for Computational Linguistics, 1994, S. 133–138

[WVMB15] Wannous, Rouaa ; Vincent, Cécile ; Malki, Jamal ; Bouju, Alain: An Ontology-Based Approach for Handling Explicit and Implicit Knowledge over Trajectories. In: *East European Conference on Advances in Databases and Information Systems* Springer, 2015, S. 403–413

[WXC10] Wong, Wai K. ; Xia, Min ; Chu, WC: Adaptive neural network model for time-series forecasting. In: *European Journal of Operational Research* 207 (2010), Nr. 2, S. 807–816

[XBK+15] Xu, Kelvin ; Ba, Jimmy ; Kiros, Ryan ; Cho, Kyunghyun ; Courville, Aaron ; Salakhudinov, Ruslan ; Zemel, Rich ; Bengio, Yoshua: Show, attend and tell: Neural image caption generation with visual attention. In: *International conference on machine learning*, 2015, S. 2048–2057

[YCP+13] Yan, Zhixian ; Chakraborty, Dipanjan ; Parent, Christine ; Spaccapietra, Stefano ; Aberer, Karl: Semantic

Trajectories: Mobility Data Computation and Annotation. In: *ACM Trans. Intell. Syst. Technol.* 4 (2013), Juli, Nr. 3, 49:1–49:38. http://dx.doi.org/10.1145/2483669.2483682. – DOI 10.1145/2483669.2483682. – ISSN 2157–6904

[YHG+16] Yang, Zichao ; He, Xiaodong ; Gao, Jianfeng ; Deng, Li ; Smola, Alex: Stacked attention networks for image question answering. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, S. 21–29

[YLT13] Ying, Josh Jia-Ching ; Lee, Wang-Chien ; Tseng, Vincent S.: Mining geographic-temporal-semantic patterns in trajectories for location prediction. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 5 (2013), Nr. 1, S. 2

[YLT14] Ying, Josh Jia-Ching ; Lee, Wang-Chien ; Tseng, Vincent S.: Mining Geographic-temporal-semantic Patterns in Trajectories for Location Prediction. In: *ACM Trans. Intell. Syst. Technol.* 5 (2014), Januar, Nr. 1, S. 2:1–2:33. – ISSN 2157–6904

[YLWT11a] Ying, Josh Jia-Ching ; Lee, Wang-Chien ; Weng, Tz-Chiao ; Tseng, Vincent S.: Semantic Trajectory Mining for Location Prediction. In: *Proc.19th ACM SIGSPATIAL*, 2011 (GIS '11). – ISBN 978–1–4503–1031–4, S. 34–43

[YLWT11b] Ying, Josh Jia-Ching ; Lee, Wang-Chien ; Weng, Tz-Chiao ; Tseng, Vincent S.: Semantic Trajectory Mining for Location Prediction. In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. New York, NY, USA : ACM, 2011 (GIS '11). – ISBN 978–1–4503–1031–4, S. 34–43

[YZC13] Ye, Jihang ; Zhu, Zhe ; Cheng, Hong: What's your next move: User activity prediction in location-based social networks. In: *Proceedings of the 2013 SIAM International Conference on Data Mining* SIAM, 2013, S. 171–179

[YZHB17] Yao, Di ; Zhang, Chao ; Huang, Jianhui ; Bi, Jingping: SERM: A Recurrent Model for Next Location Prediction in Semantic Trajectories. In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* ACM, 2017, S. 2411–2414

[YZZ+17] Yao, Di ; Zhang, Chao ; Zhu, Zhihua ; Huang, Jianhui ; Bi, Jingping: Trajectory clustering via deep representation learning. In: *Neural Networks (IJCNN), 2017 International Joint Conference on* IEEE, 2017, S. 3880–3887

[Zad65] Zadeh, Lofti A.: Fuzzy Sets. In: *Information and Control* Bd. 8, 1965

[ZHL+06] Zhao, Qiankun ; Hoi, Steven C. ; Liu, Tie-Yan ; Bhowmick, Sourav S. ; Lyu, Michael R. ; Ma, Wei-Ying: Time-dependent semantic similarity measure of queries using historical click-through data. In: *Proceedings of the 15th international conference on World Wide Web* ACM, 2006, S. 543–552

[ZWZ+08] Zheng, Yu ; Wang, Longhao ; Zhang, Ruochi ; Xie, Xing ; Ma, Wei-Ying: GeoLife: Managing and Understanding Your Past Life over Maps. In: *Proceedings of the The Ninth International Conference on Mobile Data Management*. Washington, DC, USA : IEEE Computer Society, 2008 (MDM '08). – ISBN 978–0–7695–3154–0, 211–212