UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

REGIS PIRES MAGALHÃES

SPEED PREDICTION APPLIED TO DYNAMIC TRAFFIC SENSORS AND ROAD NETWORKS

FORTALEZA

2018

REGIS PIRES MAGALHÃES

SPEED PREDICTION APPLIED TO DYNAMIC TRAFFIC SENSORS AND ROAD NETWORKS

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Banco de Dados

Orientador: Prof. Dr. José Antônio Macêdo

FORTALEZA

2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M168s Magalhães, Regis Pires.
Speed Prediction Applied to Dynamic Traffic Sensors and Road Networks / Regis Pires Magalhães. – 2018.
77 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação , Fortaleza, 2018.
Orientação: Prof. Dr. José Antônio Macêdo.

1. Speed Prediction. 2. Dynamic Traffic Sensor Networks. 3. Road Networks. 4. Urban Mobility. I. Título.

CDD 005

REGIS PIRES MAGALHÃES

SPEED PREDICTION APPLIED TO DYNAMIC TRAFFIC SENSORS AND ROAD NETWORKS

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Ciência da Computação. Área de Concentração: Banco de Dados

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. José Antônio Macêdo (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)

Dr. Francesco Lettich
Universidade Federal do Ceará (UFC)

Dr[a]. Chiara Renso
Consiglio Nazionale delle Ricerche (CNR-Italy)

Dr. Franco Maria Nardini
Consiglio Nazionale delle Ricerche (CNR-Italy)

I dedicate this work to my wife Evelane, my parents Flávio and Linda, and my dear children Renato and Luana Stanz, who have always encouraged me to be a better person every day.

ACKNOWLEDGEMENTS

I thank God and my parents for my existence and for being able to learn and share what I learn every day.

I also thank the constant support and encouragement of my wife Evelane, my parents Flávio and Linda, who have always supported me and taught me the importance of education at all times in my life. Thanks also to my dear brothers, Mario and Emanuelle.

I am very grateful to my wonderful children Renato and Luana Stanz for understanding all times that we could not be together, due to research activities developed during this work.

To Prof. José Macedo who inspires my steps every day with his dedication to research and to all another kind of work that he embraces. Thank you also for your guidance, as well as for all the support received throughout this research and for the constant encouragement over more than eight years of partnerships until this year of 2018. I hope that many more years of mutual learning can come.

To Prof. Vânia Vidal for being my advisor in the Master and Doctoral degree, and also for always encouraging me to grow as a person. Thank you for your understanding concerning the research paths I chose to pursue during my Ph.D.

To João Paulo Pordeus, Chiara, Franco, Francesco, and José Macedo, the members of this Thesis defense committee, for the readings, comments, and contributions for the improvement of this work.

To all friends from the HPC Lab in ISTI-CNR-Pisa, especially Chiara Renso, Franco Maria Nardini, Raffaele Perego, and Roberto Trani for the wonderful reception that I had during my stay in Italy in 2015 (Seek Project) and 2016, and for being always very attentive and accurate in their observations, which were essential for the accomplishment of this work. Fortunately, we were able to chat and use a lot of remote Hangouts sessions in HCP Lab and through countless other moments. To Vinícius Monteiro and Lívia Ruback, my good brazilian friends in Italy. I am also grateful to all friends and other people who helped me at CNR-Pisa: Salvatori Trani, Matteo Catena, Cristina Muntean, Luigi Caiazza, Patrizio Dazzi, Emanuele Carlini, Claudio Lucchese, Massimo Coppola, Salvatore Orlando, Nicola Tonellotto, Daniel Valcaccio, Fabrizio Fabbrini, Claudio Montani, Brunella Falchi, ...

To the esteemed researcher Francesco Lettich for all his numerous contributions and conversations for the constant improvement of this work. Francesco is a living example of

dedication to the achievement of high-quality research.

To the great friends and eternal partners of Master and Ph.D. degrees, work, research, and study, both in the Campus of Quixadá and in the Insight Lab: Lívia Almada and Ticiana Linhares. You inspire me every day. Thanks for everything. Thanks also to Lívia Almada and Erick Lima for the valuable discussions and contributions to the experiments of this work.

To all who have contributed someway to the development and improvement of the Graphast framework: José Macedo (the real father), Camila Costa, Gustavo Coutinho, Mario Nascimento, Lívia Almada, Samara Martins, Mirla Braga, Matheus Barrio, Lucas Vasconcelos, Erick Lima, Lucas Peres, David Araújo, Igo Brilhante, Josué Machado, Ticiana Linhares, Francesco Lettich, Janaína Oleinik, among others. I have learned a lot with you.

To Lucas Peres, my partner of jokes, quotations, philosophical thoughts, and all kind of fun, that helped a lot to keep me happy, even in moments of difficulties.

To the great friends of the Insight Lab, who contributed along this long journey in moments of study, work, discussions, and jokes: David Araújo (our great and esteemed project manager), Igo Brilhante, Gustavo Coutinho, Leopoldo Melo, Emanuel Oliveira, Tércio Jorge, Samara Martins, Tales Matos, Lívio Freire, Luís César, Ricardo Ávila, Minho, Guilherme, Nicksson, João Holanda, Victor, Darley, Victories, Felipes, Fujita, Hinessa, Andreza, Dayana, Matheus, Arina, Lucas Vasconcelos, Abelardo, among others. Thank you for the excellent moments of interaction and learning that we could share along this hard and, at the same time, a pleasant walk, because of your presence.

To all friends of the UFC Campus in Quixadá for the contributions to the education and development of this city, which today has a reserved place in our heart.

To all members of the ARIDA research group and the Masters and Doctoral Program in Computer Science at UFC, who contribute with advances that positively affect people's quality of life.

To all support received by the Federal University of Ceará (UFC) for allowing my stay with my family in Italy during the year of 2016.

I thank all my teachers and mentors for providing me with not only rational knowledge but also the need to improve my skills day by day and to understand the relevance of education to transform this world in a better place. Thank you for helping me learn and grow.

To all who were not mentioned up to this line, but who also contributed directly or indirectly to the accomplishment of this work.

To the companies and institutions that provided data useful for the development of this work: AMC, Simple Taxi, Greenmile, and TrixLog.

To the Seek Project for funding the research grant that supported my research at CNR-Pisa (Italy) in mid-2015.

To Fundação Cearense de Apoio ao Desenvolvimento (Funcap) for funding the research grant of Project *Segurança Pública Integrada*, which is closely related to the research contributions of this Doctoral Thesis.

The best way to predict the future is to invent it.

(Allan Kay)

RESUMO

A maioria das redes de ruas está atualmente equipada com sensores para monitoramento do tráfego em tempo real. A enorme quantidade de dados históricos de sensores coletados constitui uma rica fonte de informações que pode ser usada para extrair conhecimento útil para municípios e cidadãos, além de contribuir para a implementação de sistemas de transporte inteligentes. Neste trabalho, estamos interessados em explorar esses dados para estimar a velocidade futura em sensores dinâmicos de tráfego e redes de ruas, já que predições precisas têm o potencial de melhorar a capacidade de tomada de decisões dos sistemas de gerenciamento de tráfego. A criação de modelos preditivos de velocidade eficazes nas grandes cidades representa desafio importante que resulta da complexidade dos padrões de tráfego, do número de sensores de tráfego normalmente implantados, e da natureza em evolução das redes de sensores. De fato, sensores são freqüentemente adicionados para monitorar novos segmentos de ruas ou substituídos / removidos devido a razões diferentes (por exemplo, manutenção). A utilização de um grande número de sensores para uma predição de velocidade efetiva, requer soluções inteligentes para coletar grandes volumes de dados e treinar modelos preditivos eficazes. Além disso, a natureza dinâmica das redes de sensores do mundo real exige soluções que sejam resilientes não apenas a mudanças no comportamento do tráfego, mas também a mudanças na estrutura das redes. Este trabalho propõe três abordagens diferentes no contexto de redes de sensores grandes e dinâmicas: local, global e baseada em cluster. A abordagem local cria um modelo preditivo específico para cada sensor da rede. Por outro lado, a abordagem global constrói um modelo preditivo único para toda a rede de sensores. Finalmente, a abordagem baseada em cluster agrupa sensores em clusters homogêneos e gera um modelo para cada cluster. Outra contribuição é o fornecimento de um grande conjunto de dados, gerado a partir de registros de $\sim$1.3 milhões coletados por até 272 sensores implantados em Fortaleza, Brasil. Ele é usado para avaliar experimentalmente a eficácia e resiliência dos modelos preditivos construídos de acordo com as três abordagens mencionadas anteriormente. Os resultados mostram que as abordagens global e baseada em cluster fornecem modelos preditivos bastante precisos, que se mostram robustos a mudanças no comportamento do tráfego e na estrutura das redes de sensores, que, por sua vez, incluem o problema de *cold start*. Além disso, esta Tese propõe ainda uma abordagem de domínio cruzado que aplica modelos preditivos originalmente construídos a partir do domínio de sensores, no domínio da trajetórias. Mais especificamente, a abordagem global aqui proposta é usada para construir modelos preditivos a partir de dados de sensores, e usá-los para executar predições

relativas ao domínio de dados de trajetória. Por fim, demonstra-se que as generalizações entre domínios não são triviais e os recursos devem ser cuidadosamente selecionados para ajudar a obter resultados mais precisos.

Palavras-chave: Predição de Velocidade. Redes Dinâmicas de Sensores de Tráfego. Redes de Ruas. Mobilidade Urbana.

ABSTRACT

Most urban road networks are nowadays equipped with sensors monitoring traffic in real-time. The huge amount of historical sensor data collected constitutes a rich source of information that can be used to extract knowledge useful for municipalities and citizens and to contribute to the realization of intelligent transportation systems. In this work, we are interested in exploiting such data to estimate future speed in dynamic traffic sensors and road networks, as accurate predictions have the potential to enhance decision-making capabilities of traffic management systems. Building effective speed prediction models in large cities poses important challenges that stem from the complexity of traffic patterns, the number of traffic sensors typically deployed, and the evolving nature of sensor networks. Indeed, sensors are frequently added to monitor new road segments or replaced/removed due to different reasons (e.g., maintenance). Exploiting a large number of sensors for effective speed prediction requires smart solutions to collect vast volumes of data and train effective predictive models. Furthermore, the dynamic nature of real-world sensor networks calls for solutions that are resilient not only to changes in traffic behavior but also to changes in the network structure. We study three different approaches in the context of large and dynamic sensor networks: local, global, and cluster-based. The local approach builds a specific prediction model for each sensor of the network. Conversely, the global approach builds a single prediction model for the whole sensor network. Finally, the cluster-based approach groups sensors into homogeneous clusters and generates a model for each cluster. We provide a large dataset, generated from $\sim$1.3 billion records collected by up to 272 sensors deployed in Fortaleza, Brazil, and use it to experimentally assess the effectiveness and resilience of prediction models built according to the three aforementioned approaches. The results show that the global and cluster-based approaches provide very accurate prediction models that prove to be robust to changes in traffic behavior and in the structure of sensor networks, which, in turn, includes the cold start problem. We also propose a cross-domain approach that uses prediction models from the sensor domain into the trajectory domain. More specifically, we apply our global approach to build prediction models from sensor data and use it to perform predictions regarding the domain of trajectory data. We demonstrate that cross-domain generalizations are not trivial and the features must be carefully selected to help in achieving more accurate results.

Keywords: Speed Prediction. Dynamic Traffic Sensor Networks. Road Networks. Urban Mobility.

LIST OF FIGURES

Figure 1 – Traffic sensors monitoring vehicles in fixed locations of a city. 27

Figure 2 – Sensor based approaches. *Left-hand side*: local (model per sensor). *Center*: Global (single model for all sensors). *Right-hand side*: Cluster-based (model per cluster) 31

Figure 3 – Raw data sample regarding sensors. 32

Figure 4 – Map of the traffic sensors deployed during January and February 2014. Each circle in the map represents a group of spatially close sensors – darker circles indicate multiple sensors monitoring different lanes. 33

Figure 5 – Diagram illustrating time-slot related features. 35

Figure 6 – *Left-hand side*: example of sensor clustering using spatial distance. *Right-hand side*: example of sensor clustering using similarity between time-series. 37

Figure 7 – Analysis on the relevance of features. *Left-hand side*: feature relevance with the local approach. *Center side*: feature relevance with the global approach. *Right-hand side*: feature relevance with the cluster-based approach. 42

Figure 8 – Comparison of the local, global, and cluster-based approaches on a *static* sensor network. The *left-hand side* plot presents the evaluation according to the MSE metric, while the *right-hand side* plot presents the evaluation according to the MAPE metric. Each curve is associated to a specific pair of months. 44

Figure 9 – Comparison of the global and cluster-based approaches on a *dynamic* network of sensors over the whole 2014. The *left-hand side* plot presents the evaluation according to the MSE metric, while the *right-hand side* plot presents the evaluation according to the MAPE metric. Each curve is associated with a specific pair of months. Notice the local approach is not present here since new sensors do not have a trained model 46

Figure 10 – Cross-domain strategy. Vehicle data collected from traffic sensors on roads A and D are represented by the elements highlighted in green (sensor domain). The elements in red on roads B and C represent trajectory data (trajectory domain). 51

Figure 11 – Cross-domain data. *Left-hand side*: Predictive models built from Sensor data. *Right-hand side*: Raw trajectory data. 51

Figure 12 – A sample of taxi trajectories from Fortaleza (Brazil) on 12th Dec 2015. Each colored line represents a trajectory. 55
Figure 13 – Heatmap of GPS positions from the Taxi Simples dataset on 12th Dec 2015 from 10:00 to 11:00AM. 56
Figure 14 – Linear interpolation/extrapolation using timestamps and distances from trajectory start to find the timestamps of nodes represented with question marks. 61
Figure 15 – Diagram illustrating time-slot related features and the label representing the average speed at the prediction time. Features concerning the interval starting at 5 and 30 minutes before the query time and ending at the query time are highlighted, respectively, in green and blue. 64

LIST OF TABLES

Table 1 – Salient details of the original data produced by the network of sensors monitoring the city of Fortaleza (Brazil) during the whole 2014. 33

Table 2 – List of the *attributes* associated with a *sensor* and a *5-minute time slot*. 35

Table 3 – List of the 25 features considered in our prediction models. 36

Table 4 – Salient details of the dataset containing the aggregated observations. 38

Table 5 – Local approach, evaluation of ML techniques – 99% conf. interval. The best performers are highlighted in **bold**. 40

Table 6 – Global approach, evaluation of ML techniques – 99% confidence interval. The best performers are highlighted in **bold**. 40

Table 7 – Evaluation of ML techniques in the context of the cluster-based approach. The clustering technique used is *K-Means TS*, with $k = 8$. Confidence interval is set to 99%. The best performers are highlighted in **bold**. 41

Table 8 – Evaluation of clustering techniques according to the MSE metric. The ML technique used is MLR. The best performers are highlighted in **bold**. 42

Table 9 – Evaluation of clustering techniques according to the MAPE metric. The ML technique used is MLR. The best performers are highlighted in **bold**. 42

Table 10 – MSE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a static network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**. 44

Table 11 – MAPE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a static network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**. 45

Table 12 – MSE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a dynamic network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**. 47

Table 13 – MAPE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a dynamic network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**. 47
Table 14 – Differences between partitions of the raw trajectory dataset. 56
Table 15 – Number of samples and processing steps regarding the Taxi Simples Dataset. 56
Table 16 – Number of trajectory ids before and after the splitting step – cleaning step (iv). 58
Table 17 – Number of elements regarding map matching input and output. 60
Table 18 – Percentage of missing data regarding the features *n_lanes* and *speed_limit* in the partitions of data from 2015 and 2016. 62
Table 19 – Groups of features and their allocation according to the three set of features presented in this chapter. Time. Time-slot related features in groups (iii) and (iv) are highlighted, respectively, in green and blue. 63
Table 20 – Evaluation Comparison between different sets of features considering a static network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). Winners are highlighted in blue. 68
Table 21 – Evaluation Comparison between different sets of features considering a dynamic network of sensors in different pairs of months of 2014. The ML technique used is GBTR. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). Winners are highlighted in blue. . 68
Table 22 – Evaluation of the cross-domain approach for different sets of features using Taxi Simples trajectory datasets in Dec/2015 and Dec/2016. Winners are highlighted in **bold**. 70
Table 23 – Evaluation of cross-domain (sensor and edge based) versus trajectory only domain (edge based) both according to the global approach. Winners are highlighted in **bold**. 70

Table 24 – Groups of features and their allocation according to the three set of features presented in this chapter. Time. Time-slot related features in groups (iii) and (iv) are highlighted, respectively, in green and blue. 76
Table 25 – Evaluation Comparison between different sets of features considering a static network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). 77
Table 26 – Evaluation Comparison between different sets of features considering a dynamic network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). 77

LIST OF ACRONYMS

AMC	*Autarquia Municipal de Trânsito e Cidadania*
ARIMA	AutoRegressive Integrated Moving Average
DBN	Deep Belief Network
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DGCNN	Disordered Graph Convolutional Neural Network
ED	Euclidean Distance
GBRT	Gradient Boosting Regression Trees
GPS	Global Positioning System
HA	Historical Average
HC	Hierarchical Clustering
HD	Haversine Distance
HMM	Hidden Markov Model
LWL	Locally Weighted Learning
ML	Machine Learning
MLR	Multivariable Linear Regression
MNR	Multivariate Non-parametric Regression
MSE	Mean Squared Error
OSM	Open Street Map
RD	Road Distance
RF	Random Forest
SAE	Stacked Autoencoder
ST-ResNet	Spatio-Temporal Residual Network
SVR	Support Vector Regression
TDRN	Time-Dependent Road Networks
TS	Time-Series

CONTENTS

1 INTRODUCTION 20

2 RELATED WORK 23

2.1 Traffic Prediction 23

2.2 Cross-domain machine learning 25

3 SPEED PREDICTION IN DYNAMIC TRAFFIC SENSOR NETWORKS 27

3.1 Problem definition 29

3.2 Dataset preparation 31

3.3 Experimental Evaluation 37

3.3.1 Experimental Setting 37

3.3.2 Experimental results 39

3.3.2.1 EQ1 – Evaluation of machine learning techniques and feature relevance . . 40

3.3.2.2 EQ2 – Evaluation of the aging of predictive models generated by the local, global, and cluster-based approaches with a static sensor network 43

3.3.2.3 EQ3 – Evaluation of the resilience of predictive models generated by the local, global, and cluster-based approaches with a dynamic sensor network . 45

3.4 Discussion 48

4 VALIDATING THE GLOBAL APPROACH WITH TRAJECTORY DATA 50

4.1 Preliminaries 52

4.2 Predicting the speed of vehicles with sensor and trajectory data 53

4.2.1 Datasets 54

4.2.2 Data cleaning and transformation 57

4.2.2.1 Cleaning 57

4.2.2.2 Map matching 59

4.2.2.3 Finding speeds on the edges of each trajectory after map matching 60

4.2.2.4 Feature engineering, sensor data aggregation and model generation. 61

4.3 Experimental evaluation 64

4.3.1 Experimental Setting 64

4.3.2 Experimental questions and experiments 65

4.3.3 EQ1 – Compare the evaluation of the global speed prediction approach built from the whole set of features with the one built from a selected subset of features 66

4.3.4 ***EQ2 – Evaluate the global approach built from sensor data in road segments covered exclusively by trajectories*** 67
4.4 **Discussion** 70
5 **CONCLUSIONS AND FUTURE WORK** 71
REFERENCES 72
APPENDIX A – EVALUATION OF DIFFERENT SETS OF FEATURES 76

1 INTRODUCTION

Intelligent Transportation Systems (ITS) continuously improve the efficiency of urban transportation services thanks to increasingly advanced algorithms, increasing computing power and distributed sensor networks capable of collecting large volumes of data from road networks (SMITH *et al.*, 2002). Traffic prediction techniques are fundamental to enhance urban mobility and smart transportation technologies. Indeed, the ability to produce accurate short-term traffic predictions enables to devise highly valued services that can significantly improve urban environments – for instance, dynamic route guidance, control and management of traffic congestion, optimal routing and dispatching, detection of road accidents, and so on.

Movements of vehicles on road networks are determined by complex traffic processes governed by stochastic and non-linear interactions between individual drivers. As such, the problem of predicting the speed of vehicles is as complex as predicting the underlying traffic processes, a problem that is considered challenging in the existing literature (HOOGENDOORN; BOVY, 2001). Approaches that attempt to tackle this problem by restricting themselves to some traditional analysis on historical data are typically limited by the inherent characteristics of such data – for instance, they cannot take into account unpredictable events capable of disrupting traffic flows, such as road accidents, breakdowns and structural changes affecting a road network infrastructure, and so on (SORENSEN *et al.*, 2008). These uncertain events can possibly combine with each other, thus making the problem even more complex and requiring the use of state-of-the-art non-linear prediction techniques. Such techniques overcome the limitations of linear models in the context of traffic predictions in roadways extensively studied in (SCHMITT; JULA, 2007).

In this work, we are interested in estimating future traffic speeds in sensor networks. To achieve our goal of forecasting traffic speeds, we need to overcome the following challenges. The huge volume of data collected by real-time traffic monitoring sensors and the emergence of smart transportation technologies in urban settings, coupled with the increased capabilities and availability of sensor devices, requires traffic prediction techniques that are fast, scalable and suitable for dynamic sensor networks, where sensors are continuously added and removed. Besides, it is difficult to generalize predictive models built from traffic sensor data and apply them over the whole road network. To the best of our knowledge, we do not know other works that address the latter challenge.

We address the challenges mentioned above with the following contributions:

- We propose and analyze three different approaches that can be used to train machine-learned prediction functions: local, global, and cluster-based. The local approach is the solution commonly used in the literature, where each sensor is considered separately from others to train a specific predictive function. It suffers the cold start problem and therefore hardly applies to dynamic sensor networks, where sensors are continuously added and removed. The global approach builds a single prediction model for the whole sensor network. It provides substantial benefits in terms of reduced complexity and costs; also, relying on a single prediction function that is independent of specific sensors. The cluster-based approach groups sensors into homogeneous clusters and builds a model for each cluster. It allows to find out a good compromise between the local and global approaches, that is, depending on the number of clusters used, the behavior of this approach may resemble the local approach (when a high number of clusters is used) or the global approach (when few clusters are used).
- We devise a set of features that can be successfully used to train robust and accurate speed prediction models and we also study the relevance of each of those features in our proposed approaches.
- We provide a comprehensive experimental evaluation to assess the effectiveness of the predictive models trained according to the three approaches. The training is conducted by using four different state-of-the-art machine learning algorithms – namely, historical average time-series, multivariable linear regression, random forests and gradient boosting. Machine learning algorithms play an important role in capturing the influence of upstream and downstream flow or other traffic-related factors (WANG *et al.*, 2016a). The evaluation shows that the models created using the global and cluster-based approaches represent good solutions for dynamic sensor networks since they prove to be accurate and resilient both to model aging and to structural changes in the sensor infrastructure. We also propose a cross-domain approach that uses prediction models from the sensor domain into the trajectory domain. More specifically, we apply our global approach to build prediction models from sensor data and use it to perform predictions regarding the domain of trajectory data. We demonstrate that cross-domain generalizations are not trivial and the features must be carefully selected to help in achieving more accurate results.
- We release to the scientific community the real-world dataset used to assess our proposals. The dataset originates from ~3 billion records collected during the whole 2014 by 272

different road traffic sensors deployed in the city of Fortaleza, Brazil. To the best of our knowledge, this is the largest and richest dataset made publicly available for research on speed prediction in dynamic sensor networks.

- We compare the evaluation of our local and global approaches using different sets of features.
- We propose and evaluate a cross-domain strategy to perform speed predictions using prediction models from the sensor domain over input data extracted from the trajectory domain, i.e., we use prediction models from the sensor domain in the trajectory domain.
- We propose a methodology to aggregate speed-related features from real moving object trajectory data that we can use to create new prediction models and to serve as inputs to existing prediction models.

The remainder of this document is organized as follows: Chapter 2 reports an overview of the related works dealing with the traffic prediction and cross-domain machine learning problems. Chapter 3 proposes three speed prediction approaches targeting dynamic sensor networks. Chapter 4 evaluates our global approach introduced in Chapter 3 with real-world trajectory data. Finally, Chapter 5 shows our conclusions and plans for future works.

2 RELATED WORK

This chapter reports an overview of the related works regarding the main research topics presented in Chapters 3 and 4 of this Thesis, which are traffic prediction and cross-domain machine learning.

2.1 Traffic Prediction

Short-term traffic prediction aims at estimating traffic conditions from few seconds to few hours in the future, based on current and past traffic information. The field has an extensive and longstanding research history that originates in the 1980s in the context of intelligent transportation systems. A comprehensive and recent survey (VLAHOGIANNI *et al.*, 2014) observes how this research area moved from a classical statistical perspective (e.g. ARIMA) to data-driven modeling techniques based on machine learning and neural networks. Most of the interest in this field focuses on developing methodologies that can be used to model traffic characteristics such as volume, density, speed, travel times, and produce estimates of future traffic conditions.

The IEEE ICDM 2010 Contest (WOJNARSKI *et al.*, 2010) fostered the development of Machine Learning solutions tackling traffic prediction. One of the tasks of the contest addressed speed prediction based on a real-time stream of synthetic data from vehicles in Warsaw (Poland). The data stream consisted of Global Positioning System (GPS) locations of the traveling vehicles sampled every 10 seconds, and the task asked to predict the average speed on selected road segments for a close time interval (0-6' minutes) and a farther one (24-30' minutes). The winning solution proposed the adoption of a random forest model (HAMNER, 2010). The authors employ two kinds of features to model the speed: i) features computed by a global traffic flow model common to all road segments and ii) features computed by a local traffic flow model that strictly depends on the road segment considered. The global traffic flow model outputs 68 features while the local traffic flow models compute from 6 to 42 features, depending on the road segment.

The most related aspect of the aforementioned article to our work is the adoption of some form of "global" knowledge to make the learned solutions more robust and effective. In this work we remark that we investigate machine learning models trained on all or subsets of sensors deployed in a large road network. On the one hand, we believe that the increasing availability

and heterogeneity of traffic data, pushed both by innovations in sensor technologies and the urgency of mobility problems faced in highly-populated urban areas, call for methodologies that are accurate, robust to variations in the characteristics of the road segments considered, and that can accommodate dynamic networks of traffic sensors. On the other hand, to the best of our knowledge state-of-the-art solutions for short-term speed prediction are limited to local models trained on historical data collected from individual sensors. Thus, while a local approach allows to train very effective prediction models at sensor level, it is also very demanding in that it requires to train and maintain periodically different models for each sensor – indeed, this issue becomes relevant when the number of sensors becomes large, or sensors are frequently added (removed) to (from) the network.

For what concerns the machine learning techniques considered in our work, we report that state-of-the-art solutions (ZHANG; HAGHANI, 2015; WANG *et al.*, 2016a) use models based on Gradient Boosting Regression Trees (GBRT), as this technique proves to be superior to others. We also report, however, that the existing literature evaluates the aforementioned techniques limitedly to some local approach.

Recent works experiment deep learning technologies for predicting short-term vehicle speed. Deep learning algorithms use multiple-layer deep architectures to extract inherent features from data to model patterns and structures. Indeed, deep learning allows to represent complex traffic features without previous knowledge (HUANG *et al.*, 2014; LV *et al.*, 2015). A deep learning architecture composed of a Deep Belief Network (DBN) and a multitask regression layer is proposed in (HUANG *et al.*, 2014). The DBN is used for unsupervised feature learning, while the multitask regression layer above the DBN is used to supervise the prediction through multitask learning. The experimental evaluation shows that the proposed approach outperforms well-established competitors such as AutoRegressive Integrated Moving Average (ARIMA) (VOORT *et al.*, 1996), Bayesian (SUN *et al.*, 2006), Support Vector Regression (SVR) (CASTRO-NETO *et al.*, 2009), Locally Weighted Learning (LWL) (SHUAI *et al.*, 2008), Multivariate Non-parametric Regression (MNR) (CLARK, 2003), and neural networks (CHAN *et al.*, 2012). (YU *et al.*, 2016) proposes a data grouping approach based on a convolutional neural network called Disordered Graph Convolutional Neural Network (DGCNN) to forecast urban short-term traffic flows, where the neural network uses spatial relations between traffic locations to train predictive models. In the experimental evaluation the authors show that the DGCNN produces more accurate predictions than competitors such as historical average, ARIMA, and

Stacked Autoencoder (SAE) (LV *et al.*, 2015). In (Li *et al.*, 2017) the authors propose the use of convolutional neural networks, centered on the notion of *diffusion convolution*, to capture spatial and temporal dependencies among traffic flows. Finally, in (Zhang *et al.*, 2017) the authors propose Spatio-Temporal Residual Network (ST-ResNet), a deep-learning-based approach that forecasts the inflow and outflow of crowds in spatial regions within urban areas – as such, we note that this work targets a different problem with respect to the one considered in our work. The approach relies on three residual neural networks to model some properties characterizing spatio-temporal data, more precisely, *temporal closeness*, *period*, and *trend properties* of crowd traffic. The aggregated output of the networks is then integrated with external factors, such as weather and day of the week, to predict at once the inflows and outflows of the spatial regions considered.

Overall, the works mentioned above are orthogonal to our proposal, as we investigate how to reduce management complexity by minimizing the number of models to be trained and maintained with respect to the local approach. It is recognized that deep learning models trained on large datasets outperform those using small training dataset. Although a similar investigation is out of the scope of this chapter, it is likely that short-term vehicle speed prediction based on deep neural networks can benefit from the global or cluster-based approaches we propose, thanks to the present availability of large amounts of training samples.

2.2 Cross-domain machine learning

Cross-domain machine learning also known as transfer learning is extensively applied in Chapter 4 and other works. Transfer learning techniques try to transfer the knowledge from some previous tasks to a target task when the latter has fewer high-quality training data (PAN *et al.*, 2010). Transfer learning and domain adaptation refer to the situation where what has been learned in one setting is exploited to improve generalization in another setting (GOODFELLOW *et al.*, 2016).

(ASIF *et al.*, 2014) state that factors such as changes in transportation infrastructure can significantly affect long-term traffic patterns, and that supervised learning methods may not work well in such cases. Then, they suggest the use of techniques based on transfer learning in such scenarios.

(XU *et al.*, 2016) propose an approach to learn a prediction model from graphical traffic condition data provided by Baidu Map, which is a commercial, close-source map provider

in China, and apply the model on Open Street Map (OSM) so that one can predict the traffic conditions with nearly 90% accuracy in various parts of Shanghai, China, even though no traffic data is available for that area from Baidu Map. The system can be used in urban planning, transportation dispatching as well as personal travel planning.

(WANG *et al.*, 2016b) propose a deep learning method with an Error-feedback Recurrent Convolutional Neural Network structure (eRCNN) for continuous traffic speed prediction. Another contribution is a weight pre-training method, which adopts a transfer-learning notion by clustering similar yet contiguous road segments into a group for the generation of a same set of initial weights. This not only helps to reduce the learning process of eRCNN for every road segment, but also improves the chance of finding better optimal solutions.

(MILLER; GUPTA, 2012) show the high level of transfer learning possible by training a model on one region and testing the model on incidents from a different year and region. This is in some extent similar to what we do when we create prediction models that generalize the behavior of groups of sensors, and use those models to perform predictions for new sensors.

This work also applies transfer learning by using predictive models built from sensor data in the trajectory domain. The aforementioned related works apply cross-domain machine learning, but in different domains and exploring other features.

3 SPEED PREDICTION IN DYNAMIC TRAFFIC SENSOR NETWORKS

Highly populated cities increasingly face mobility challenges caused by transport and traffic. The huge volume of data collected by real-time traffic monitoring sensors provides new opportunities to develop models and algorithms that enhance transportation services towards intelligent transportation systems, in particular those dealing with traffic predictions. Vehicle speeds on road networks are determined by complex traffic processes governed by stochastic and non-linear interactions between individual drivers (HOOGENDOORN; BOVY, 2001), hence predicting the speed of vehicles is as complex as predicting the underlying traffic processes. Short-term traffic prediction techniques have been investigated and exploited since some time (VLAHOGIANNI *et al.*, 2014). However, the emergence of smart transportation technologies in urban settings, coupled with the increased capabilities and availability of sensor devices, requires traffic prediction techniques that are fast, scalable and suitable for complex and heterogeneous urban networks.

Many different traffic sensor technologies are currently used to monitor road networks, such as those based on inductive-loop detectors, magnetometers, video image processors, microwave radar sensors, laser radar sensors, passive infrared sensors, ultrasonic sensors, passive acoustic sensors, and devices exploiting combinations of the aforementioned technologies (KLEIN *et al.*, 2006). In this chapter we focus on sensors capable of capturing the speed of vehicles traveling over large and dynamic sensor networks, where devices can be added or removed from the network for various reasons, and we address the problem of training accurate prediction models that are capable of maintaining their accuracy over time – we call this the *model aging* problem – and cope with structural changes affecting sensor networks – we call this

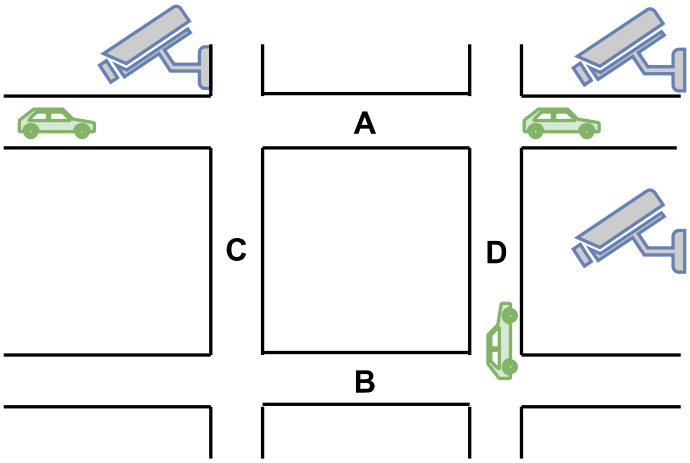

Figure 1 – Traffic sensors monitoring vehicles in fixed locations of a city.

the *network dynamicity* problem. Figure 1 represents three traffic sensors monitoring vehicles in a city.

We address these challenges by proposing and analyzing three different approaches that can be used to train machine-learned prediction functions: *local*, *global*, and *cluster-based*. The *local* approach is the solution commonly used in the literature, where each sensor is considered *separately* from others to train a specific predictive function. This approach suffers the *cold start* problem and therefore hardly applies to dynamic sensor networks, where sensors are continuously added and removed. Moreover, in large and dynamic sensor networks the local approach requires to train and maintain a large amount of different prediction models. To overcome these issues we propose the global and cluster-based approaches, where models are trained on data coming from all the sensors in the network (or groups of *similar* sensors, in the cluster-based case) to build resilient predictive functions. The global approach provides substantial benefits regarding reduced complexity and costs. Furthermore, by relying on a single prediction function that is independent of specific sensors, it naturally solves the *cold start* problem. The global approach is expected to be robust concerning structural changes occurring in sensor networks, thus also addressing the *dynamicity* problem. We also show that the cluster-based approach allows us to find out a good compromise between the local and global approaches: indeed, depending on the number of clusters, the behavior of this approach may resemble the local approach (when we use a high number of clusters) or the global approach (when we use few clusters). In general, this approach allows us to tune the number of clusters to fit the needs and characteristics of specific sensor networks.

The contributions of this chapter can be summarized as follows:

- we propose the *global* and *cluster-based* approaches for learning vehicle speed prediction functions in large and dynamic sensor networks;
- driven by three experimental questions, we provide a comprehensive evaluation to assess the effectiveness of the predictive models trained according to the three approaches. The training is conducted by using different state-of-the-art machine learning algorithms on a large, real-world sensors dataset. The dataset covers a time span of 12 months, during which 130 (145) sensors were added (removed) to (from) the network. The evaluation shows that the models created using the global and cluster-based approaches represent good solutions when dealing with dynamic sensor networks, as they prove to be accurate and resilient both to model aging and to structural changes in the sensor infrastructure

(which, in turn, includes the *cold start* problem);

- we release to the scientific community the real-world dataset used to assess our proposals. The dataset originates from ∼1.3 billion records collected during the whole 2014 by 272 different road traffic sensors deployed in the city of Fortaleza, Brazil. Due to privacy concerns we do not release the original raw data, but a dataset obtained after an aggregation and cleaning process. To the best of our knowledge, this is the largest and richest dataset made publicly available for research on speed prediction in dynamic sensor networks.

The chapter is structured as follows: Section 3.1 defines our prediction problem and discusses three approaches to solve the problem. Section 3.2 presents the dataset used in our experiments, as well as the pre-processing steps used to transform the data into a format suitable for speed prediction. Section 3.3 details the experimental evaluation and discusses the results. Finally, Section 3.4 draws the final conclusions and sketches potential lines of future research.

3.1 Problem definition

Let $S = \{s_1, \ldots, s_n\}$ be a network of n sensors overseeing the traffic conditions of a specific geographical area. Within a given time interval T, sensors in S produce a collection of observations, where each observation is a triple (t_j, s_j, x_{speed}) recording the time $t_j \in T$ of the event of a vehicle passing by some sensor $s_j \in S$ with a speed x_{speed}.

Let us then denote by O the set of *average speed observations* that are produced as follows: the whole time interval T is split in time-buckets of fixed length (e.g., 5 minutes each) and, for each bucket and sensor, the average speed of all the vehicles observed is computed. Each *average speed observation* is thus represented by a triple $(k, i, speed)$, where k and i are respectively the identifiers of the time bucket and sensor, while *speed* is the average vehicle speed observed. Moreover, let $y(k,i)$ be a function that, given the identifiers of the bucket and sensor, returns the observed average speed, i.e., $y(k,i) = speed$.

Then, we define the PREDICTSPEED problem as the problem of finding an accurate function f for predicting $y(k,i)$ given all the previous observations recorded in O, i.e., all the observations having a time bucket identifier lower than k.

Definition 3.1.1 (PREDICTSPEED) *The* PREDICTSPEED *problem requires to find a predictive function $\hat{f}$ over the class of all possible predictive functions H such that:*

$$\hat{f} = \arg\min_{f \in H} \Delta(f), \tag{3.1}$$

where Δ is a loss function assessing the quality of a candidate predictive function f over the observations in O. In this work we use the Mean Squared Error (MSE) loss function defined as:

$$\Delta_{MSE}(f) = \frac{1}{|O'|} \sum_{(k,i,speed) \in O'} (f(k,i) - y(k,i))^2 \tag{3.2}$$

where O' is the set of observations used to assess the quality of the prediction and $f(k,i)$ is the estimate returned by function f for $y(k,i)$. The smaller the value yielded by the above loss functions, the better the predictive performance of f.

We employ Machine Learning (ML) techniques to address the PREDICTSPEED problem. More specifically, we aim at learning from the observations in O some function $\hat{f}$ that minimizes the error measured by Δ. We train the prediction models on datasets containing examples built from past sensors observations, where each example is represented by an high-dimensional vector of features. The aim of these features is to model relations between the traffic conditions observed in time buckets prior to k and the speed $y(k,i)$ that will be recorded by sensor i within the time bucket k (in other words, the label to predict). To train the models we rely on state-of-the-art machine learning techniques for *regression* tasks (FRIEDMAN *et al.*, 2001): GBRT (FRIEDMAN, 2001; FRIEDMAN, 2002), Random Forests (BREIMAN, 2001) and Linear Regression (FREEDMAN, 2009).

In this work we address large traffic sensor networks typically instrumenting the roads of large cities. As such, we are interested in studying speed prediction techniques that are resilient not only to changes in traffic behavior, but also to changes in the network where sensors are frequently added to monitor new road segments or replaced/removed due to various maintenance reasons. To investigate this scenario we introduce three different approaches that can be used to learn $\hat{f}$, i.e., the *local*, *global* and *cluster-based* approaches. The *local* approach learns a different prediction function $\hat{f}_i$ for *each* sensor s_i using the observations recorded by s_i and defines the prediction function $\hat{f}$ in terms of n distinct *local* prediction functions $\hat{f}_i$. The *global* approach learns the prediction function $\hat{f}$ from the observations of *all* sensors. Finally, the *cluster-based* approach uses a similarity measure to partition the sensors into k disjoint clusters, and learns a distinct prediction function $\hat{f}_c$ for each cluster c via the observations recorded by the sensors associated with c.

To the best of our knowledge state-of-the-art techniques solving the speed prediction problem employ the local approach, as this strategy fits the dynamics of individual sensors within the network nicely. However, the local approach cannot be applied to new sensors added to the network due to the lack of historical data – this is also known as the *cold start* problem. Moreover,

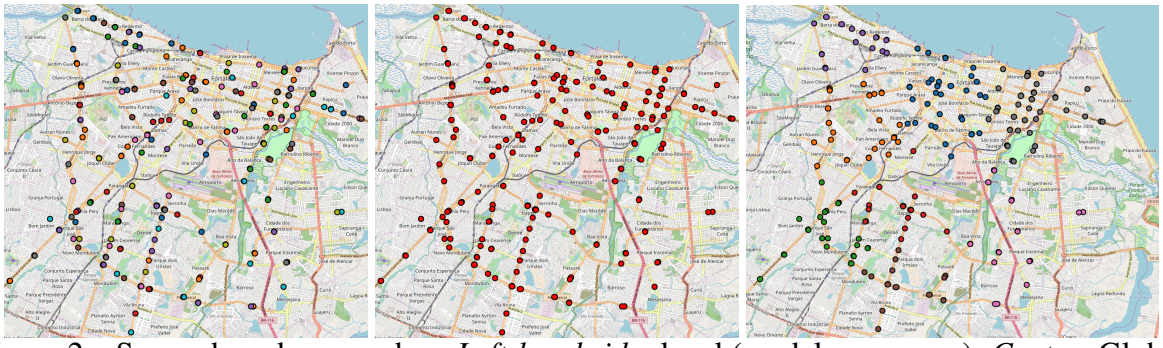

Figure 2 – Sensor based approaches. *Left-hand side*: local (model per sensor). *Center*: Global (single model for all sensors). *Right-hand side*: Cluster-based (model per cluster)

using the local approach in large networks typically implies a huge data management overhead due to the training and maintenance of possibly hundreds of different prediction models.

The most natural strategy to tackle this issue is to use the global approach, where a single prediction model is trained over data from the whole sensor network. Indeed, this approach is expected to generalize well over previously unseen sensors and adapts well to changes in traffic behaviors. Unfortunately this generalization power comes with a cost, as a global model might not fit the dynamics of individual (or groups of) sensors.

Finally, we argue that the cluster-based approach allows finding a proper trade-off between the advantages and disadvantages of the local and global approaches: the higher the number of clusters, the more the models generated by the cluster-based approach fit the dynamics of individual sensors; conversely, the lower the number of clusters, the more the generated models tend to capture *general* traffic dynamics. Figure 2 represents the three sensor-based approaches proposed in this chapter: local, global and cluster-based approaches, respectively, where each small circle represents a traffic sensor, and the color of each circle represents a distinct prediction model in the city of Fortaleza.

3.2 Dataset preparation

We evaluate the local, global, and cluster-based approaches introduced in Section 3.1 by means of a real-world dataset containing data from traffic sensors deployed in the city of Fortaleza (Brazil). The dataset is provided by *Autarquia Municipal de Trânsito e Cidadania* (AMC), the authority supervising Fortaleza's road-network. The raw dataset consists of about 1.3 billions records, collected by a network of 302 sensors during the whole year of 2014, for a total of 60 GB of data. Each record is associated with the passage of one vehicle in the area covered by one of the sensors. Each record contains five fields: i) *sensor ID*, representing the

sensor_id	timestamp	n_lanes	max_speed	speed
1	2014-01-31 00:00:06	2	60	49
1	2014-01-31 00:00:17	2	60	51
1	2014-01-31 00:00:22	2	60	48
1	2014-01-31 00:00:36	2	60	58
1	2014-01-31 00:01:08	2	60	51
1	2014-01-31 00:01:09	2	60	43
1	2014-01-31 00:01:17	2	60	64
1	2014-01-31 00:01:49	2	60	57
1	2014-01-31 00:04:18	2	60	62
1	2014-01-31 00:04:41	2	60	50

Figure 3 – Raw data sample regarding sensors.

identifier of the sensor that produced the record, (ii) *timestamp* t, indicating when the record was produced, (iii) *lane number* l, indicating the number of lanes monitored by the sensor, (iv) *maximum lane velocity* s_l, the maximum speed allowed in the lane(s), and (v) *speed* s of the vehicle that triggered the record creation. Figure 3 presents a sample of raw data regarding sensors.

Among the sensors in the dataset, only 154 were always continuously active during all the months of the year. Indeed, the sensor network was subjected to frequent additions and removals, mainly due to hardware malfunctions, contract expirations, contract renewals, and so on. Figure 4 shows the locations of 234 sensors that were active during January and February 2014, while Table 1 presents some characteristics of the dataset: the column **# records** reports the number of observations gathered during the associated month, while the column **# sens. added** reports the number of active sensors in a given month that were *not appearing* in the preceding month. Similarly, the column **# sens. removed** reports the number of sensors that were not active in a given month while they are active in the preceding one. Finally, the column **# active sens.** reports the overall number of active sensors within the associated month. From the figures in the Table we observe that the network of sensors is highly dynamic, thus indicating the importance of prediction models that are resilient to changes in the network.

In the next paragraphs we detail the data preparation phases needed to build from the raw dataset the dataset O of *average speed observations* given in input to the ML techniques considered in this work. More precisely, we illustrate how we *filtered out* the *outliers*, *aggregated* the data, and *engineered* the various features.

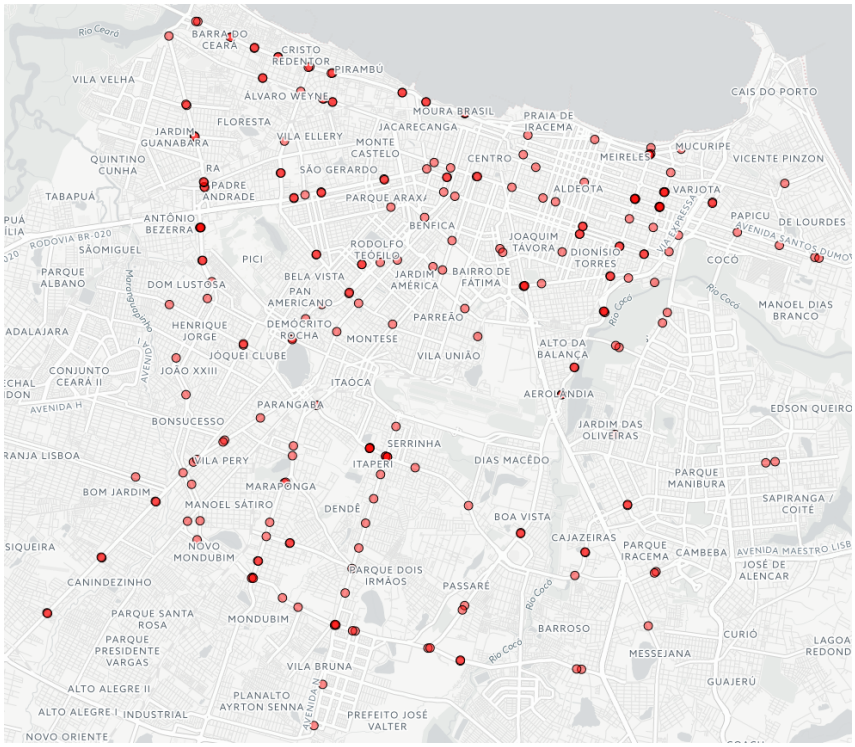

Figure 4 – Map of the traffic sensors deployed during January and February 2014. Each circle in the map represents a group of spatially close sensors – darker circles indicate multiple sensors monitoring different lanes.

Month	# records	# sens. added	# sens. removed	# active sens.
January	116,448,334	-	-	236
February	89,272,352	5	51	190
March	87,505,939	0	5	185
April	87,838,370	2	0	187
May	113,754,231	53	13	227
June	85,672,156	6	54	179
July	94,307,178	10	0	189
August	125,829,696	66	4	251
September	94,889,414	7	62	196
October	129,457,780	69	0	265
November	125,366,035	9	2	272
December	132,161,025	0	4	268
Tot. records	1,282,502,510			

Table 1 – Salient details of the original data produced by the network of sensors monitoring the city of Fortaleza (Brazil) during the whole 2014.

Data cleaning. We first filter out from the dataset observations that are possibly affected by anomalies. To this end, we partition the observations by month and compute the mean μ and the standard deviation σ of the speed within each month. Finally, we remove the observations whose speed has a *distance* from μ greater or equal than 3σ. The output of this phase consists of the

set of observations appearing in the original dataset *minus* the ones that are deemed anomalous by the above criterion.

Data aggregation. The goal of this second phase is to generate the set O of *average speed observations* from the data obtained at the end of the first phase. To this end we partition the data into six distinct time intervals, each spanning a period of two months, and aggregate the data in each partition according to 5-minute time slots. Then, for each sensor and 5-minute time slot pair we compute the attributes shown in Table 2.

The choice of using 5-minute time slots is common in state-of-the-art literature (CHEN *et al.*, 2003; CHEN, 2003; MIN; WYNTER, 2011; ZHANG; HAGHANI, 2015; ZHANG; ZHANG, 2016; RZESZÓTKO; NGUYEN, 2012), while the choice of intervals spanning 2 months represents a proper trade-off between the need to have enough data to perform the training, validation, and evaluation of the models, and the need to have a reasonable number of test sets spanning the whole dataset timeline in order to evaluate the robustness of the models learned with respect to *aging*.

The resulting dataset is the set of average speed observations O detailed in Table 4 of Section 3.3.1. We report that we *released* the aggregated dataset to the scientific community[1] to ensure the reproducibility of our results and promote research developments in this field.

Feature engineering. We aim at devising a *"good"* set of features that can be successfully used to train robust and accurate speed prediction models. Before introducing the features used, we explain how information in the temporal domain are exploited to derive them. Figure 5 provides a schema of the temporal intervals considered for feature modeling. From the Figure we first notice *Query time*, which represents the time instant in which the prediction request occurs. *Query time* is associated with a specific 5-minute time slot, i.e., the 5-minute time slot preceding the one in which *Query time* falls: this represents *Query time*'s time slot of reference and it is denoted by ts_{ref}.

The speed prediction refers to a future 5-minute time slot, ts_f. Inspired by several works available in the literature (VLAHOGIANNI *et al.*, 2004; SCHMITT; JULA, 2007; WOJNARSKI *et al.*, 2010; MIN; WYNTER, 2011), we use a predictive horizon of 30 minutes after the beginning of ts_{ref}[2]. Besides the fundamental time slots mentioned above, to perform accurate

[1] The dataset will be released upon acceptance of the manuscript.

[2] In general, we note that the width of all the intervals involved can be parametrized according to specific

Attribute	Description
sensor_id	Identifier of the sensor.
n_lanes	Number of lanes monitored by the sensor.
speed_limit	Maximum speed allowed in the road monitored by the sensor.
timestamp	Time instant associated with the start of the 5 minute time-slot.
vehicle_count	Number of vehicles (*throughput*) that pass by the sensor within the 5 minute time-slot.
avg_speed	Average speed of vehicles that pass by the sensor within the 5 minute time-slot.
std_speed	Standard deviation of the speed of vehicles that pass by the sensor within the 5 minute time-slot.
min_speed	Minimum speed of vehicles that pass by the sensor within the 5 minute time-slot.
max_speed	Maximum speed of vehicles that pass by the sensor within the 5 minute time-slot.

Table 2 – List of the *attributes* associated with a *sensor* and a *5-minute time slot*.

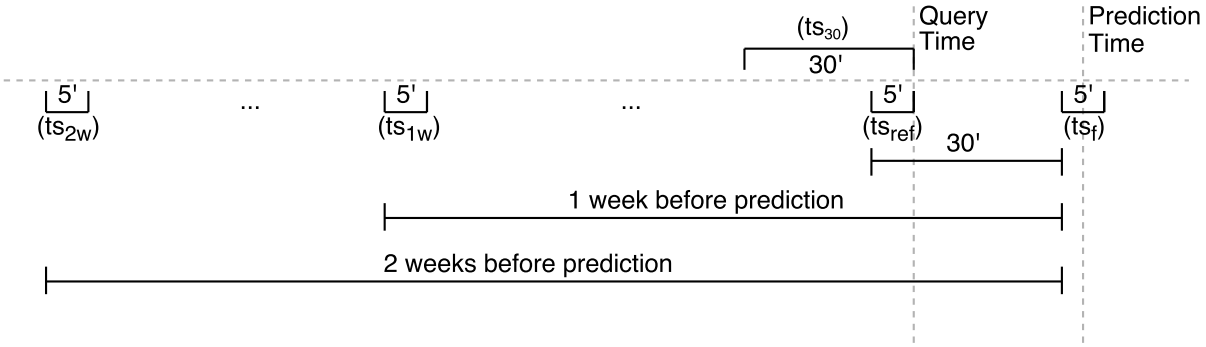

Figure 5 – Diagram illustrating time-slot related features.

predictions we leverage information contained within few other *selected* time slots related to *Query time*. More specifically we consider:

- ts_{30}: 30-minute time slot that ends at the same time instant of ts_{ref}'s ending.
- ts_{1w}: 5-minute time slot starting one week before the beginning of ts_f.
- ts_{2w}: 5-minute time slot starting two weeks before the beginning of ts_f.

For all the 5-minute time slots and all the sensors in S we use the schema highlighted above to design a set of 25 features that model traffic conditions. Table 3 reports the complete list, together with the time slots they refer to. These 25 features can be divided into six different groups as shown in the Table. We note that the second group contains categorical features; as such, we converted them to numerical values based on the following semantic:

application needs.

Group of Features	Acronym	Feature Name
(i) Sensor	n_lanes	Sensor number of lanes
	speed_limit	Speed limit
(ii) Time reference	day_of_week	Day of week
	slot_of_day	Slot of day
	working_day	Working day
(iii) 5 min last time slot (ts_{ref})	v_count5	Number of vehicles
	min5	Minimum speed
	max5	Maximum speed
	avg5	Average speed
	std5	Standard Deviation
(iv) 30 min last time slot (ts_{30})	v_count30	Number of vehicles
	min30	Minimum speed
	max30	Maximum speed
	avg30	Average speed
	std30	Standard Deviation
(v) One week before prediction time slot (ts_{1w})	v_count1w	Number of vehicles
	min1w	Minimum speed
	max1w	Maximum speed
	avg1w	Average speed
	std1w	Standard Deviation
(vi) Two weeks before prediction time slot (ts_{2w})	v_count2w	Number of vehicles
	min2w	Minimum speed
	max2w	Maximum speed
	avg2w	Average speed
	std2w	Standard Deviation

Table 3 – List of the 25 features considered in our prediction models.

- *day of the week*: 0 (Monday), 1 (Tuesday), 2 (Wednesday), 3 (Thursday), 4 (Friday), 5 (Saturday) or 6 (Sunday).
- *slot of day*: value comprised between 0 to 287, due to the discretization of time into 5-minutes time slots (i.e., $24 \cdot (60/5)$ slots).
- *working day*: equal to 1 if the time slot falls within a working day, 0 otherwise.

Finally, we use the average speed in the future time slot (ts_f) as the prediction label of the current observations. It is worth noticing that information concerning sensor identifiers or their geographical coordinates are not included among our features since we want to train models that are able to *generalize* over different sensors.

The features in groups (iii) and (iv) capture two different time slots close to *Query time*. We included both of them as they may capture specific time-dependent traffic trends. To the best of our knowledge, this is the first work that addresses the construction of speed prediction models based on time-dependent groups of features.

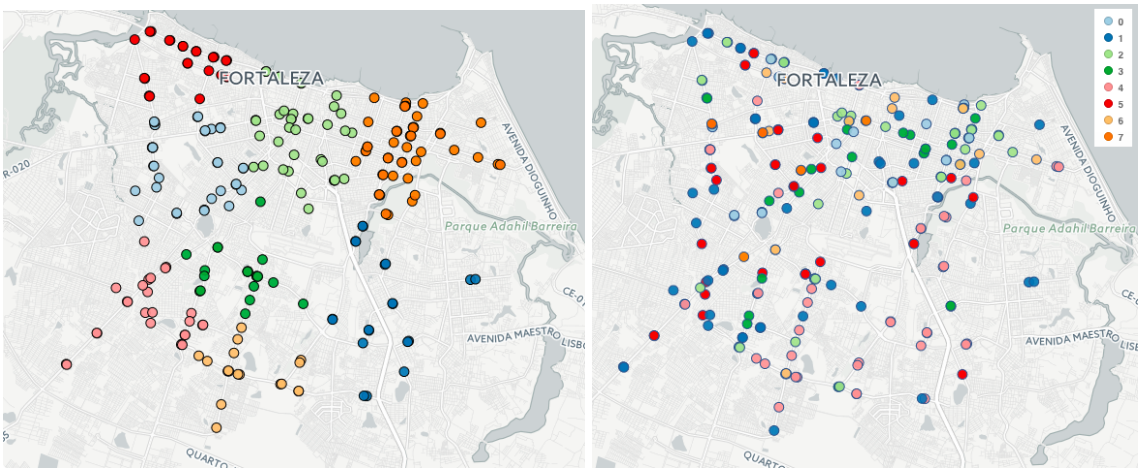

Figure 6 – *Left-hand side*: example of sensor clustering using spatial distance. *Right-hand side*: example of sensor clustering using similarity between time-series.

3.3 Experimental Evaluation

In this section we discuss the experiments conducted to generate different prediction models and assess their performance. More specifically, Section 3.3.1 introduces the experimental setting used to conduct the results evaluation, while Section 3.3.2 introduces the experimental questions and discusses the results.

3.3.1 Experimental Setting

Test system. We conduct our experiments on a server with 16 Xeon E5520 Intel CPUs, each clocked at 2.27GHz, with 8192 KB L3 cache, 24 GB of RAM, and Ubuntu OS (16.04 LTS).

Sensor clustering. The cluster-based approach relies on some clustering strategy to operate. To this end, we evaluate two different strategies: the first one exploits *sensor geolocation* information and employs a spatial distance function to compute distances between pairs of sensors. For the purposes of this work we experimented with the *Euclidean*, *haversine*, and *road network* distances. The second strategy focuses on the *similarity* of traffic behaviors observed by different sensors and clusters sensors accordingly. We implement the latter strategy by modeling sensors as time series containing sequences of sensor observations. Specifically, each sensor is modeled by a weekly sequence of average speeds computed for each 5-minute time slot, *weighed* by the *number of cars* observed. In this way, each sensor is represented by a time series of $7 \times 24 \times 60/5 = 2,016$ values. Consequently, the *similarity* between a pair of time series can be

Months	# agg. obs.	# sensors	# added	# removed	# intersection
Jan/Feb	2,155,944	235	-	-	-
Mar/Apr	1,939,679	181	0	54	181
May/Jun	2,063,709	225	8	18	217
Jul/Aug	2,316,110	241	27	19	214
Sep/Oct	2,273,177	255	48	26	207
Nov/Dec	1,306,081	261	57	28	204

Table 4 – Salient details of the dataset containing the aggregated observations.

determined as the Euclidean distance between the associated multidimensional points. Figure 6 provides examples of clusters found with the latter strategy.

We established the best clustering strategy by performing an extensive experimental evaluation, comparing the MSE yielded by models generated by the cluster-based approach through different clustering techniques – for the sake of brevity we omit the discussion of these experiments and report that the clustering strategy achieving the best results uses *K-Means++* (ARTHUR; VASSILVITSKII, 2007) to cluster sensors modeled as *time-series*. Consequently, in the experiments presented in Section 3.3.2 the cluster-based approach employs this strategy.

Dataset. The dataset used in the experimental evaluation, built as discussed in Section 3.2, contains 12,054,700 records arranged into six partitions, each representing a pair of consecutive months. Table 4 provides the most salient details about the dataset. For each pair of months we report the number of aggregated observations (**# agg. obs.**) and the number of operating sensors. The **# added** (**# removed**) columns refer to sensors that have been added (removed) with respect to the data partition covering January and February. Finally, the column **# intersection** reports the number of sensors that a partition has in common with the one covering *January* and *February*.

Training, validation, and test sets. We split the data into three distinct partitions, i.e., training, validation, and test sets. The training and validation sets span a temporal interval covering January and February 2014, while the temporal interval spanned by the test set depends on the specific experiment. From Section 3.2 we remember that some of the features refer back to two weeks before prediction time. Moreover, we report that the cluster-based approach requires 2 weeks of data to cluster the sensors. Consequently, the first two weeks of any temporal partition are used exclusively for feature and cluster computation, thus leaving the remaining 6 weeks for training, validation, or evaluation.

To create the three sets we perform a *stratified* sampling, based on the identifier of the sensors, to distribute homogeneously the data of each sensor across the partitions, thus avoiding to perform predictions that involve sensors not present in the training set. The training and validation sets include respectively the 65% and 15% of the data of each sensor, while the fraction of data in the test set depends on the specific experiment.

Machine learning methods. The machine learning methods considered in this work to train models are Multivariable Linear Regression (MLR) (FREEDMAN, 2009), Random Forest (RF) (BREIMAN, 2001), and GBRT (FRIEDMAN, 2001; FRIEDMAN, 2002). The implementations of MLR and RF are provided by the Scikit-Learn machine learning library (PEDREGOSA *et al.*, 2011), while the implementation of GBRT is provided by XGBoost (CHEN; GUESTRIN, 2016; CHEN; HE, 2015). We also consider Historical Average (HA), a baseline algorithm that predicts the average speed in a given time slot ts_i by averaging the speed of all the training examples having the same day_of_week and $slot\_of\_day$ of ts_i.

For what concerns the hyperparameters needed by RF and GBRT, we determine the best combination by means of a grid search. The first hyperparameter required by RF and GBRT is the *maximum tree depth* (max_depth) – to this end we consider the range $[3,9]$ (with step 2). GBRT and RF also require to provide the number of trees ($n\_estimators$) to be used in a model; to this end the implementation of GBRT employs an early stopping technique to find out the best value, while for RF we consider the range $[20,100]$ (with step 10) when considering the local approach, the range $[1000,4000]$ (with step 500) with the global approach, and the range $[50,2000]$ (with step 100) with the cluster-based approach. Finally, GBRT requires a third hyperparameter, the learning rate ($learning\_rate$), for which we consider the range [0.05, 0.2] (with step 0.05). Overall, for each possible combination we generate a model and pick the one whose model yields the lowest MSE. The evaluation is conducted over the validation set.

3.3.2 Experimental results

The experiments aim at comprehensively answering the following experimental questions:

EQ1 Which machine learning algorithms achieve the best results when used to address the PREDICTSPEED problem in the context of a static sensor network? Also, which are the most relevant features for the local, global, and cluster-based approaches?

EQ2 In the context of a static sensor network, are the models trained according to the local, global and cluster-based approaches resilient with respect to *aging*?

EQ3 Are the prediction models trained according to the local, global, and cluster-based approaches robust in managing effectively the structural changes affecting a real-world *dynamic* network of sensors? Are the global and cluster-based approaches able to address the *cold start* problem?

In the following sections we answer the above questions.

3.3.2.1 EQ1 – Evaluation of machine learning techniques and feature relevance

The main goal of this study is to evaluate and establish the best machine learning technique among those tested for each approach. In this context we consider a static sensor network and compare the following methods: Historical Average (Hist.Avg.), a commonly used baseline, Linear Regression (MLR), Random Forest (RF) and Gradient Boosting Regression Trees (GBRT). The evaluation uses a *test set* limited to the 20% of sensor data covering January and February 2014. To counterbalance the sparsity possibly characterizing the data, we consider only those observations associated with more than 22 passing vehicles within the range of the sensors in the 30-minutes time slot. We use the MSE and MAPE metrics defined in Section 3.1 to assess the quality of the models. Finally, we report the use of 99% confidence intervals to determine the best performers.

Table 5 provides a comparison of the competing techniques in the context of the *local* approach. From the Table we see that RF and GBRT perform similarly and outperform the other techniques. Table 6 provides a comparison in the context of the *global* approach. From

Algorithm	Mean MSE	Conf. Interval MSE	Mean MAPE	Conf. Interval MAPE
Hist.Avg.	16.1995	(15.9955, 16.4035)	0.0889	(0.0883, 0.0894)
MLR	11.4155	(11.2787, 11.5524)	0.0765	(0.0760, 0.0770)
RF	**10.6961**	**(10.5643, 10.8280)**	**0.0735**	**(0.0730, 0.0740)**
GBRT	**10.8910**	**(10.7556, 11.0264)**	**0.0733**	**(0.0728, 0.0738)**

Table 5 – Local approach, evaluation of ML techniques – 99% conf. interval. The best performers are highlighted in **bold**.

Algorithm	Mean MSE	Conf. Interval MSE	Mean MAPE	Conf. Interval MAPE
Hist.Avg.	94.1161	(93.7075, 94.5246)	0.2864	(0.2852, 0.2876)
MLR	12.2855	(12.1393, 12.4317)	0.0792	(0.0787, 0.0797)
RF	12.4181	(12.2711, 12.5652)	0.0798	(0.0793, 0.0803)
GBRT	**11.5756**	**(11.4354, 11.7159)**	**0.0767**	**(0.0762, 0.0772)**

Table 6 – Global approach, evaluation of ML techniques – 99% confidence interval. The best performers are highlighted in **bold**.

the Table we see that GBRT is the clear winner, with RF and MLR following closely. Finally, Table 7 provides a comparison in the context of the *cluster-based* approach, where the approach uses $k = 8$ clusters. From the Table we see that GBRT is the clear winner, with RF and MLR following closely. We report that using different values for the number of clusters k yielded similar results, here omitted for brevity.

We have evaluated different clustering techniques in the context of the cluster-based approach. The clustering algorithms used to implement the first strategy are *K-Means* (HARTIGAN, 1975), Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (ESTER *et al.*, 1996) and Hierarchical Clustering (HC) (ROKACH; MAIMON, 2005). We report that with the DBSCAN approach we use the Euclidean Distance (ED), Haversine Distance (HD), and Road Distance (RD). For K-Means we use ED and also a customized version of K-Means that create clusters based on the similarity between the speed Time-Series (TS) of sensors. Figure 6 provides an overview on the geolocation of clustered sensors using different clustering strategies. Finally, with HC we report that we use the RD with the *ward*, *complete* and *average* techniques.

K-Means TS strategy attempts to cluster the sensors according to the similarity characterizing their speed time-series, where each time-serie represents a sequence of *5-minute slots* having *weekly period*. As such, each time-serie contains 2016 time-slots that span a week (i.e., 288 time-slots per single day or 12 time-slots per hour).

Finally, each slot is associated with the *average speed of sensors* registered within the slot weighted by the *number of cars that passed by* the sensor within the time slot.

For brevity, we present the evaluation by employing the MLR technique; however, we report that the findings hold for the other ML techniques as well. Tables 8 and 9 presents the results. From the experiments, we observe that increasing the number of clusters has the effect of yielding better results for all the clustering approaches considered. Finally, from the tables we observe that our customized K-Means TS clustering strategy achieves the best results.

Algorithm	**Mean MSE**	**Conf. Interval MSE**	**Mean MAPE**	**Conf. Interval MAPE**
Hist.Avg.	17.6529	(17.4730, 17.8328)	0.1020	(0.1014, 0.1027)
MLR	11.9701	(11.8283, 12.1119)	0.0784	(0.0779, 0.0789)
RF	11.7423	(11.6019, 11.8826)	0.0776	(0.0771, 0.0781)
GBRT	**11.1659**	**(11.0303, 11.3014)**	**0.0754**	**(0.0749, 0.0759)**

Table 7 – Evaluation of ML techniques in the context of the cluster-based approach. The clustering technique used is *K-Means TS*, with $k = 8$. Confidence interval is set to 99%. The best performers are highlighted in **bold**.

# clusters	DBScan RD	DBScan ED	DBScan HD	HC Ward RD	K-Means ED	K-Means TS
2	12.2753	12.2790	12.2753	12.2683	12.2706	**12.2083**
4	12.2656	12.2670	12.2203	12.2477	12.2438	**12.1147**
8	12.2000	12.1848	12.2019	12.2119	12.2081	**11.9703**
16	12.1521	12.1988	12.1988	12.1512	12.1238	**11.8712**
32	12.0580	12.0439	12.0533	12.0250	12.0258	**11.7692**
64	11.8993	11.8621	11.8374	11.8369	11.8678	**11.6713**

Table 8 – Evaluation of clustering techniques according to the MSE metric. The ML technique used is MLR. The best performers are highlighted in **bold**.

# clusters	DBScan RD	DBScan ED	DBScan HD	HC Ward RD	K-Means ED	K-Means TS
2	0.0792	0.0792	0.0792	0.0792	0.0792	**0.0791**
4	0.0791	0.0791	0.0790	0.0791	0.0791	**0.0788**
8	0.0790	0.0790	0.0790	0.0790	0.0791	**0.0784**
16	0.0789	0.0789	0.0789	0.0789	0.0788	**0.0781**
32	0.0786	0.0786	0.0786	0.0786	0.0786	**0.0778**
64	0.0782	0.0781	0.0781	0.0780	0.0781	**0.0775**

Table 9 – Evaluation of clustering techniques according to the MAPE metric. The ML technique used is MLR. The best performers are highlighted in **bold**.

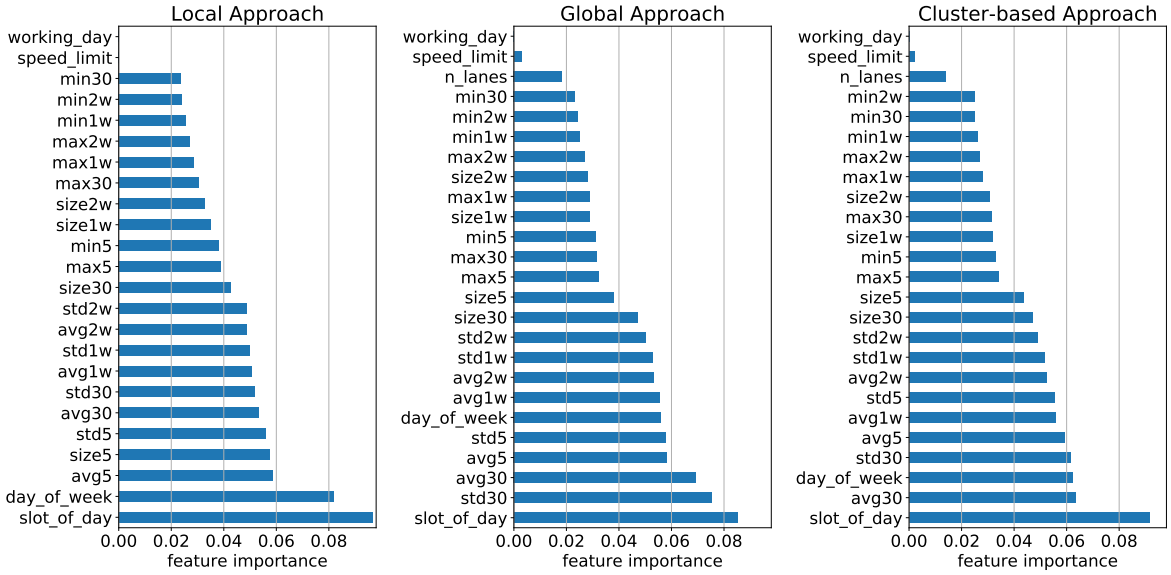

Figure 7 – Analysis on the relevance of features. *Left-hand side*: feature relevance with the local approach. *Center side*: feature relevance with the global approach. *Right-hand side*: feature relevance with the cluster-based approach.

Relevance of features. In the batch of experiments that follows we study the relevance of the features introduced in Section 3.2, Table 3, in the contexts of the local, global, and cluster-based approaches. We employ GBRT, since in the previous study it proved to be the best ML technique. The relevance of each feature is estimated by counting the number of times it is used in a split node of any decision tree in the GBRT forest. Figure 7 presents the results. The more an attribute is used in decision trees, the higher its relative importance.

From the plots we observe how the *temporal-dependent* features exhibit the highest relevance; in particular, some of the features belonging to group *(ii)*, namely *slot of the day* and *day of the week*, exhibit maximal or near-maximal relevance for all the approaches. This represents a quite intuitive result, as *time of the day* and *day of the week* help to characterize traffic flows. Similarly, other temporal-dependent features exhibiting high relevance are those related to the time slots associated with prediction time – namely, the features of group *(iii)* (5 min last time slot), group *(iv)* (30 min last time slot), group *(v)* (one week before prediction time slot), and finally group *(vi)* (two weeks before prediction time slot). In general we observe that, the more a feature group refers to a time slot distant from prediction time, the less relevant the associated features. Finally, we observe that the non-temporal features belonging to group *(i)* exhibit the lowest relevance.

In conclusion, this study shows that GBRT represents the best ML technique for all the considered approaches, with RF following closely. The study also shows that the most relevant features are the ones concerning temporal information, with features associated to time slots close to prediction time exhibiting more relevance than those associated with farther time slots.

3.3.2.2 EQ2 – Evaluation of the aging of predictive models generated by the local, global, and cluster-based approaches with a static sensor network

The main goal of this study is to evaluate how predictive models generated by the local, global, and cluster-based approaches age over time. Indeed, we recall that traffic behavior tend to change due to holiday periods, large events, changes in the road network, seasonal trends, and so on. It is therefore reasonable to suspect that a model built on data covering a specific period may not represent well the traffic behavior in subsequent periods. Assuming that we want to avoid a frequent re-training of the models, this study aims to understand which approaches achieve the most consistent predictive performance over time.

In the batch of experiments that follows we consider a scenario with a static sensor network and analyze the performance of the predictive models built on January and February over a period of one year. We train our models on the training set ds_{train} and consider six different *test sets*, one for each pair of months available in the dataset (Table 4). To simulate a static sensor network, we limit our analysis to the set of 156 sensors that operated continuously across the whole 2014. We observe that the test set associated with January and February covers the 20%

of sensor data, as the remaining data is used for training and validation. For what concerns the other pairs of months, the related test sets cover the last 6 weeks, since the first 2 weeks are used to compute specific subsets of features – more precisely, the groups (v) and (vi) – and to cluster the sensors. The performance of the approaches is evaluated by means of the MSE and MAPE metrics (Section 3.1). Specifically, we consider (i) the average performance and (ii) the fraction of sensors for which the global and cluster-based approaches perform better than the local one.

The ML technique used to generate the models is GBRT. Finally, the cluster-based approach uses k values comprised in the $[2,64]$ range – indeed, this range represents an appropriate transition from the global approach to the local one. Tables 10 and 11 summarize the MSE and MAPE values yielded by the local, cluster-based and global approaches using models trained only on sensors present in January and February (i.e., our reference months). Figure 8 provides a pictorial overview of the results.

From the results we observe how the local approach achieves the best results only in January and February, while its performance degrades noticeably in the months that follow. Conversely, the results highlight the robustness and resilience of the cluster-based (with $k \in [2,8]$)

Partition	**Local**	**Cluster (k = 8)**	**Cluster (k = 4)**	**Cluster (k = 2)**	**Global**
Jan-Feb	**10.89**	11.17 (37.4%)	11.30 (27.7%)	11.42 (21.7%)	11.58 (18.3%)
Mar-Apr	13.81	**13.00 (81.2%)**	13.00 (79.6%)	12.99 (79.0%)	13.13 (67.4%)
May-Jun	14.23	12.64 (86.6%)	12.26 (86.6%)	**12.18 (87.6%)**	12.33 (84.3%)
Jul-Aug	14.47	**11.99 (81.8%)**	11.88 (80.4%)	11.83 (76.6%)	11.84 (72.4%)
Sep-Oct	15.49	**12.62 (89.4%)**	12.49 (87.9%)	12.18 (86.5%)	12.23 (82.6%)
Nov-Dec	15.60	12.19 (87.7%)	**12.02 (87.7%)**	11.74 (85.8%)	11.75 (85.3%)

Table 10 – MSE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a static network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**.

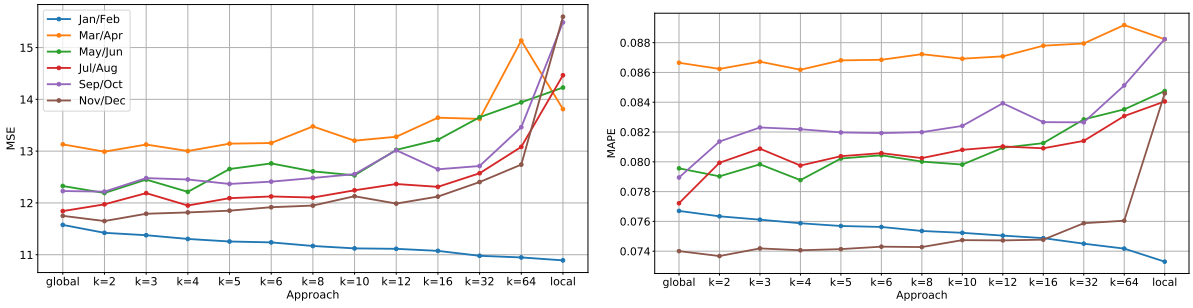

Figure 8 – Comparison of the local, global, and cluster-based approaches on a *static* sensor network. The *left-hand side* plot presents the evaluation according to the MSE metric, while the *right-hand side* plot presents the evaluation according to the MAPE metric. Each curve is associated to a specific pair of months.

Partition	Local	Cluster (k = 8)	Cluster (k = 4)	Cluster (k = 2)	Global
Jan-Feb	**0.0733**	0.0754 (17.9%)	0.0759 (13.6%)	0.0763 (8.5%)	0.0767 (8.1%)
Mar-Apr	0.0882	0.0862 (58.6%)	**0.0863 (61.9%)**	0.0862 (59.7%)	0.0867 (52.5%)
May-Jun	0.0848	**0.0807 (79.3%)**	0.0793 (77.0%)	0.0791 (74.2%)	0.0796 (70.0%)
Jul-Aug	0.0841	**0.0780 (71.5%)**	0.0775 (68.7%)	0.0774 (68.7%)	0.0772 (66.4%)
Sep-Oct	0.0882	0.0805 (76.8%)	**0.0800 (77.3%)**	0.0789 (73.4%)	0.0789 (72.0%)
Nov-Dec	0.0846	**0.0756 (79.9%)**	0.0750 (78.4%)	0.0742 (77.9%)	0.0740 (77.9%)

Table 11 – MAPE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a static network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**.

and global approaches, as they consistently achieve better accuracy than the local approach and their performance tend to remain stable across the months. The very same trends can be observed when considering the fraction of sensors for which the global and cluster-based approaches perform better than the local one. In general, we argue that the local approach performs consistently worse due to its inability to *generalize* traffic behavior. The global and cluster-based approaches consistently achieve very good performance, thus suggesting that they are capable to effectively capture changes in *global* traffic behavior over time. Finally, the plots in Figure 8 show that the cluster-based approach tends to generate increasingly less accurate models as the number of clusters becomes large: indeed, for sufficiently large values this approach suffers from the same limitations of the local approach.

3.3.2.3 EQ3 – Evaluation of the resilience of predictive models generated by the local, global, and cluster-based approaches with a dynamic sensor network

In this study we consider a scenario with a dynamic sensor network, and analyze the performance of the global and cluster-based approaches to assess their resilience with respect to changes that affect the network over time. Thus, differently from the previous studies we do not limit ourselves to a fixed subset of sensors but consider also sensors that are added to the network outside the temporal interval spanned by the training set (i.e., beyond the end of February 2014). To this end we conduct two different batches of experiments: in the first batch we focus on the performance of the global and cluster-based approaches over one year of data, and verify that it is consistent with the performance observed in the case of a static sensor network; note that in this context we do not consider the local approach, as the training data within ds_{train} does not allow to create models for sensors added to the network beyond the end of February 2014. In the second batch of experiments we compare the performance of the local, global, and cluster-based

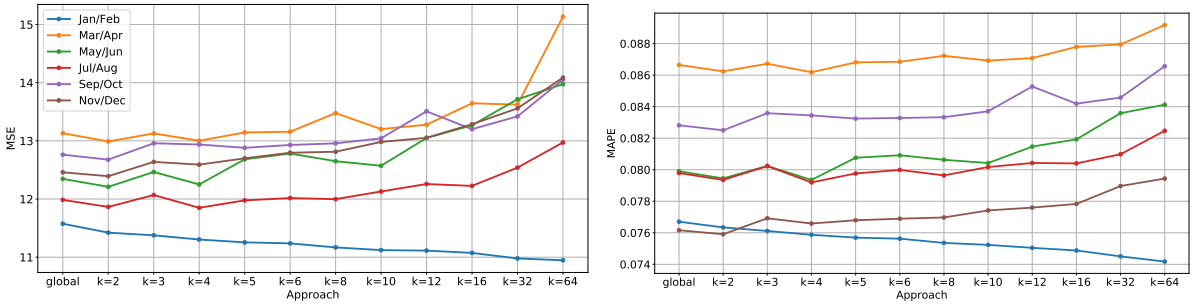

Figure 9 – Comparison of the global and cluster-based approaches on a *dynamic* network of sensors over the whole 2014. The *left-hand side* plot presents the evaluation according to the MSE metric, while the *right-hand side* plot presents the evaluation according to the MAPE metric. Each curve is associated with a specific pair of months. Notice the local approach is not present here since new sensors do not have a trained model

approaches limitedly to some of the sensors added to the network beyond the end of February 2014, to prove the superior predictive performance of the latter approaches.

In the first batch of experiments we train our models on the training set ds_{train} and consider six different *test sets*, one for each pair of months available in the dataset (Table 4). We observe that the test set associated with January and February covers the 20% of sensor data, as the remaining data is used for training and validation. For what concerns the other pairs of months, the related test sets cover the last 6 weeks, since the first 2 weeks are used to compute specific subsets of features – more precisely the groups (v) and (vi) – and to cluster the sensors. The performance of the approaches is evaluated by means of the MSE and MAPE metrics (Section 3.1). The ML technique used to generate the models is GBRT. Finally, the cluster-based approach uses k values in the $[2,64]$ range. Figure 9 presents the results.

From the Figure we observe that the results resemble closely the ones achieved in the context of EQ2 (Figure 8), thus proving that the global and cluster-based approaches are able to manage effectively *structural changes* in *dynamic* sensor networks.

In the second batch of experiments we compare the performance of the local, global, and cluster-based approaches. In this context we notice that the global and cluster-based models used in the first batch of experiments can be straightforwardly reused. Due to its characteristics, however, the local approach requires a different training procedure since it requires to train a model each time a new sensor is added to the network. This in turn requires to limit the analysis to the set of sensors added to the network beyond the end of February 2014. Accordingly, each local model is trained over the first three weeks in the pair of months where its sensor appears, and evaluated using the data within the subsequent five weeks. This implies that we have to consider only those sensors whose data is present both in the training and test sets, and explains

Temporal Partition	# Sensors Added	Local	Cluster (k=8)	Cluster (k=4)	Cluster (k=2)	Global
Mar-Apr	0	–	–	–	–	–
May-Jun	2	17.34	17.42 (50.0%)	16.59 (100.0%)	**14.63 (100.0%)**	15.05 (100.0%)
Jul-Aug	10	10.97	11.01 (60.0%)	10.91 (60.0%)	**10.80 (60.0%)**	10.81 (60.0)%)
Sep-Oct	26	**15.47**	16.32 (30.77%)	16.36 (34.6%)	15.90 (42.3%)	15.48 (46.2%)
Nov-Dec	52	15.87	16.76 (51.92%)	16.11 (51.9%)	15.72 (53.9%)	**15.42 (65.4%)**

Table 12 – MSE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a dynamic network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**.

Temporal Partition	# Sensors Added	Local	Cluster (k=8)	Cluster (k=4)	Cluster (k=2)	Global
Mar-Apr	0	–	–	–	–	–
May-Jun	2	0.1442	0.1452 (50.0%)	0.1406 (100.0%)	**0.1249 (100.0%)**	0.1281 (100.0%)
Jul-Aug	10	0.0766	0.0754 (70.0%)	0.0757 (70.0%)	**0.0754 (80.0%)**	0.0756 (80.0%)
Sep-Oct	26	0.0895	0.0926 (34.6%)	0.0920 (38.5%)	0.0904 (53.9%)	**0.0889 (61.5%)**
Nov-Dec	52	0.0876	0.0900 (42.3%)	0.0889 (51.9%)	0.0867 (63.5%)	**0.0857 (69.2%)**

Table 13 – MAPE yielded by the local, global and cluster-based approaches (with 8, 4 and 2 clusters) in a dynamic network of sensors for each pair of months. The percentages in parentheses represent the *fraction* of sensors for which the cluster-based or global approaches performs better than the local one. Winners are highlighted in **bold**.

why the number of added sensors may be different between Tables 12, 13 and Table 4. We assess the performance of the approaches by considering the average error achieved over all the sensors considered and by considering the fraction of sensors for which the global and cluster-based approaches perform better than the local one. Tables 12 and 13 presents the results – note that the tables do not provide any result for March and April, as no new sensors were added with respect to the preceding pair of months.

The results show that the global and cluster-based (with $k = 2$) approaches generally outperform their competitor, both in terms of average accuracy and in the fraction of sensors where they achieve better performance. This proves that the global and cluster-based approaches are able to effectively tackle changes in dynamic sensor networks. Furthermore, the results demonstrate that they properly address the *cold start* problem since the global and cluster-based approaches achieve comparable or superior accuracy to the local approach without the need to retrain a model each time a new sensor is added to the network. Finally, we notice that all the approaches achieve their best performance over the months of July and August: we argue that this is due to the similarity of traffic behavior between January/February and July/August – indeed, both pairs of months include extensive holiday periods.

3.4 Discussion

In this chapter we consider the problem of predicting the speed of vehicles by analyzing data collected from a large and dynamic network of sensors. To this end we evaluate three different approaches, called the local, global and cluster-based, that leverage state-of-the-art supervised machine learning techniques. These approaches have complementary advantages: the local approach, which is the traditional strategy adopted by state-of-the-art literature, trains a model for each sensor, and generally achieves high performance in the presence of long term historical data. However, this approach suffers from the cold start problem and can be hardly applied to dynamic sensor networks, where sensors are frequently added and removed. Moreover, the local approach entails consistent data management costs with large and dynamic sensor networks, as it requires to train and maintain many different prediction models.

To tackle the limitations of the local approach, in this chapter we propose the cluster-based and global approaches. The cluster-based approach trains prediction models on clusters of sensors to capture traffic behavior observed by "similar" sensors, while the global approach trains a single prediction model over the observations of all the sensors of the network. Subsequently, we formally introduce the speed prediction problem and design a methodology that allows to train models according to the three approaches.

To evaluate the approaches we conduct an extensive experimental evaluation on a very large dataset collected in the city of Fortaleza, Brazil – we also report that we made the dataset publicly available to ensure the reproducibility of our results and promote research developments in this field. Driven by three experimental questions, we first analyze three state-of-the-art machine learning techniques for the speed prediction problem, and prove that gradient boosting with regression trees outperforms the other techniques. Subsequently, we focus on the characteristics of static sensors networks, studying how the models, trained according to the three approaches, age over time. Here we observe how the local approach achieves the best results when tested over the same period covered by the training set, while it degrades noticeably over the subsequent temporal periods due to its inability to generalize different traffic behavior. On the contrary, the global approach addresses the *aging* problem effectively, as it allows to train models that capture traffic changes without the need to perform expensive re-trainings, with the cluster-based approach following closely. Finally and most importantly, the evaluation shows that the global and cluster-based approaches consistently outperform the local approach when facing consistent structural changes in dynamic sensor networks, thus addressing effectively the

network dynamicity problem and the *cold start* problem.

Overall, the superior performance of the global and cluster-based approaches is determined by their resilience to model aging and to structural changes affecting dynamic sensor networks, and we believe that these results have the potential to impact intelligent transportation systems and current data management practices in the field of speed prediction.

One possible line of future research deals with feature engineering; for instance, one may consider introducing features related to information in various domains to further improve the accuracy, such as weather conditions, accidents, road-network maintenance, and so on. Other features may be engineered to consider information related to *spatially close* (according to some proximity function or clustering process) sensors. Finally, another possible line of research may leverage the approaches presented in this work to generate cost functions in the context of *time-dependent road networks* (MAGALHÃES *et al.*, 2015).

4 VALIDATING THE GLOBAL APPROACH WITH TRAJECTORY DATA

In Chapter 3 we considered the problem of predicting the speed of vehicles using mobility data generated by dynamic sensor networks. To address this problem we evaluated three different approaches and demonstrated that the global approach yields the best results. Indeed, the experimental evaluation shows that global models maintain high levels of accuracy over time and are resilient to changes in the network, thus demonstrating that the global approach allows us to capture and exploit urban-wide traffic behaviors effectively.

Using sensor data as the sole source of information allows us making predictions only over road segments covered by the sensor network. In this chapter, we aim at expanding the reach of predictive models by combining the use of sensor data with trajectory data. Trajectory data is a sequence of time-stamped points regarding a moving object, each of which contains the information of latitude and longitude. Optionally, we could also use the altitude, but it is out of the scope of this work. Figure 10 shows an example of a cross-domain strategy. Vehicle data collected from traffic sensors ($ds_{sensors}$) placed in roads A and D are used to create the predictive models (M). Those elements are highlighted in green. On the other hand, the elements in red represent trajectory data (ds_{traj}) in roads B and C. We can use the features extracted from ds_{traj} and compatible with the features from models M as inputs to M to perform speed predictions in roads B or C, that is, in roads not covered by M. Figure 11 illustrates data from both domains with predictive models built from sensor data on the left-side and the raw trajectory data on the right-side of the figure.

Specifically, we propose a hybrid strategy: on the one hand, we use sensor data to extract traffic behaviors (i.e., train the models), since sensor networks constitute a reliable and continuous source of traffic information that operates round the clock. On the other hand, we aim at using trajectory data to gather information over road segments that are not covered by sensors to aggregate features and use them as a test set. In the following, we evaluate models built from sensor data over the test set. The evaluated models follow the global approach considering three sets of features. The evaluation results compare the effectiveness of the global approach concerning speed predictions in those sets of features. We also consider two periods of time to split our trajectory dataset into two partitions and evaluate them independently.

The main contributions of this chapter can be summed up as follows:

- We propose a cross-domain approach that uses data from one domain in another one. More specifically, we apply our global approach to build a prediction model from sensor data

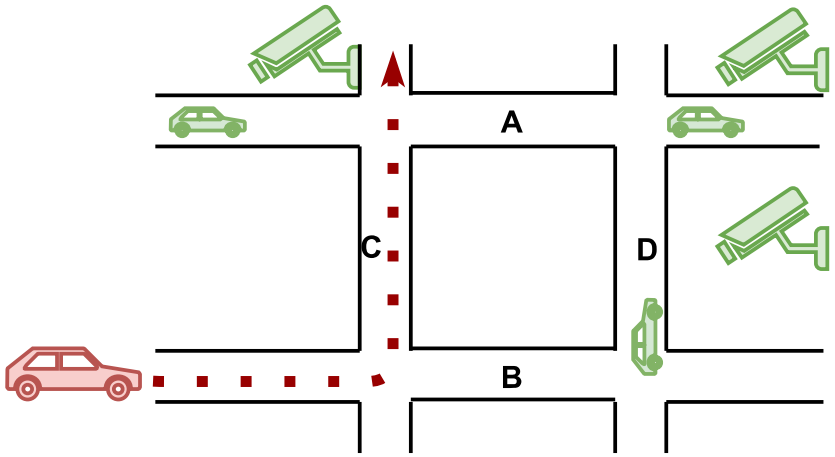

Figure 10 – Cross-domain strategy. Vehicle data collected from traffic sensors on roads A and D are represented by the elements highlighted in green (sensor domain). The elements in red on roads B and C represent trajectory data (trajectory domain).

id	lat	lon	time
148869	-3.805515	-38.586766	2015-12-01 04:36:30
173450	-3.746524	-38.595895	2015-12-01 04:36:38
175306	-3.755231	-38.597673	2015-12-01 04:36:38
191676	-3.734355	-38.587616	2015-12-01 04:36:39
211845	-3.745625	-38.596039	2015-12-01 04:36:39

Figure 11 – Cross-domain data. *Left-hand side*: Predictive models built from Sensor data. *Right-hand side*: Raw trajectory data.

and use it to perform predictions regarding the domain of trajectory data.

- We engineer three sets of features that allow us to combine the use of sensor data with trajectory data.
- We propose a methodology to aggregate speed related features over road segments traversed by trajectories.
- We evaluate the proposed strategy over test sets built from trajectory data and compare its outputs with results achieved by using solely sensor data, i.e., the approach presented in Chapter 3.
- We have developed an Open Source Python library called PyRoad[1] to simplify queries and visualization of trajectories and other spatial-temporal data.

The chapter is structured as follows: Section 4.1 presents some preliminary notions used throughout the chapter. Section 4.2 presents the methodology used to evaluate the global approach with trajectory data. Section 4.2.1 presents the data sets used in our experiments and the

[1] <https://github.com/InsightLab/PyRoad>

steps to pre-process such data into a format suitable for data prediction. Section 4.3 presents the experimental evaluation. Finally, Section 4.4 draws the final conclusions and presents potential future research.

4.1 Preliminaries

This section presents key concepts and definitions regarding our strategy of evaluating a prediction model built from sensor data by using test sets made from trajectory data.

Definition 4.1.1 (Road network) *The structure of a road network can be modeled by a graph where the nodes represent the intersections, start and end points of a road segment (e.g. a street or avenue), and the edges connect the nodes. Depending on the application, additional points may represent changes in the curvature or the maximum velocity of a segment. Consider graph* $G = (V, E, C)$ *where: (i)* $V = \{v_1, ..., v_n\}$ *is a set of vertices (nodes); (ii)* $E = \{(v_i, v_j) | v_i, v_j \in V, i \neq j\}$ *is a set of edges; (iii)* $C = \{w_{(v_i,v_j)} | (v_i, v_j) \in E, w_{(v_i,v_j)} \in \mathbb{R}_{\geq 0}\}$ *is a set of costs, where* $w_{(v_i,v_j)}$ *is a cost that assigns a positive weight to the edge* (v_i, v_j)*. In road networks, this is the distance or the travel time between intersections* v_i *and* v_j*. In this chapter we consider C as the distance between intersections* v_i *and* v_j*. The existence of an edge* (v_i, v_j) *does not imply the existence of* (v_j, v_i)*. Besides, it allows that the opposite edges,* (v_i, v_j) *and* (v_j, v_i)*, may hold* $w_{v_i,v_j} \neq w_{v_j,v_i}$*.*

The movements of objects moving over a road network are typically captured by GPS-enabled devices and subsequently represented in the form of trajectories. To this end, it is useful to provide some basic definitions.

Definition 4.1.2 (Trajectory) *The movement of a moving object* o_j *can be described in terms of a continuous function* $M_j : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^2$ *from the domain of real non-negative numbers, representing time instants, to a 2D space. Given an object* o_j *and a time interval* $[tBegin, tEnd)$*, a trajectory* TR *is the restriction of the movement* M_j *with respect to* $[tBegin, tEnd)$*.*

In mobility applications it is common to assume that the continuous movement of a moving object cannot be completely known. As such, trajectories are often described in terms of a finite set of time-stamped positions, also known as *trajectory tracks*.

Definition 4.1.3 (Trajectory track) *Given a temporally ordered sequence* $< t_1, \cdots, t_n >$ *of timestamps, the track of a trajectory T for the given timestamps is represented by the tem-*

porally ordered sequence of trajectory points $< p_1 = (x_1, y_1, t_1), \cdots, p_n = (x_n, y_n, t_n) >$*, where the pair* (x_i, y_i) *represents the geographical coordinates associated with* p_i*, while* t_i *represents the associated timestamp. Finally, we use* $T(t_i) = (x_i, y_i)$ *to denote the geographical coordinates of* T *at time* t_i*.*

For the sake of brevity we use the term *trajectory* instead of *trajectory track* throughout the rest of the chapter.

Definition 4.1.4 (Map Matching) *Map matching is a process to convert a sequence of raw latitude/longitude coordinates to a sequence of road segments (ZHENG, 2015). The key problem in map matching is the tradeoff between the roads suggested by the location data and the feasibility of the path.*

For the purposes of this work we assume map matching as the matching of the original geographic coordinates of trajectories to the edges of the underlying road network. (ZHENG, 2015) presents a comprehensive overview of the existing algorithms. This work uses a Hidden Markov Model (HMM) map matching algorithm (NEWSON; KRUMM, 2009) to find the most likely road route represented by a time-stamped sequence of latitude/longitude pairs. The HMM accounts for measurement noise and the layout of the road network. The states of the HMM are the individual road segments, and the state measurements are the noisy vehicle location measurements. The goal is to match each location measurement with the proper road segment. This state representation naturally fits the HMM, because transitions between road segments are governed by the connectivity of the road network.

More formally, the discrete states of the HMM are the N road r segments, r_i, where $i = 1...N_r$. Distinct road segments run between intersections. For each 2D latitude/longitude location measurement z_t, the goal is to find the most probable path through the lattice by picking one road segment for each t. This path should be sensitive to both the measurements and the reasonability of the paths between the road segments. This tradeoff is made based on the probabilities governing the measurements and probabilities governing the transitions between the road choices at each time.

4.2 Predicting the speed of vehicles with sensor and trajectory data

This section proposes a strategy to perform speed predictions in road segments covered and even not covered by traffic sensors. To this end, we aggregate features from

trajectory data and use them as input to prediction models based on the global approach defined in Chapter 3. The proposed strategy allows us to evaluate those models by using a test set built from trajectory data.

The basic requirement to perform a speed prediction using a predictive model m is to provide a new observation o, represented as a vector of features, as input to the model. Then, the model m receives o as input and outputs the predicted speed s.

Let $G = (V, E, C)$ be a graph representing a road-network (Definition 4.1.1). Let also $TR = \{T_1, \cdots, T_m\}$ be the set of trajectories of objects moving over the road-network represented by a graph G (Section 4.1.4). The main goal of this section is to expand the prediction reach of models built from sensor data to edges of the road network represented by G, where we can aggregate features from TR. Therefore, we use TR to compute a set O of observations associated with specific edges of the road network represented by the graph G. The set O of observations reflects traffic behaviors contained within TR. O denotes the set of speed observations that are produced as follows: from the travel time tt and the length of edge e, we can derive the speed s in an edge; the whole time interval T is split in time-buckets of fixed length (e.g., 5 minutes each); for each bucket and each edge we compute the minimum, maximum, mean and standard deviation of speeds from the trajectories that enter the edge at a time instant contained within the bucket. Each speed observation is thus represented by a triple (k, e, s), where k and e denote the identifiers of the time bucket and edge respectively, while s is the quadruple ($s_{min}, s_{max}, s_{avg}$ and s_{std}) where the elements represent respectively the minimum, maximum, average and standard deviation of speeds observed in the time bucket k for the edge e.

4.2.1 Datasets

For this work we use two main datasets: the first one contains historical sensor data provided by the Autarquia Municipal de Transito e Cidadania (AMC), the traffic agency operating in Fortaleza (Brazil) that we already presented in Chapter 3. The second dataset is a historical trajectory dataset released by *Taxi Simples*, a Brazilian cab fleet monitoring company. The dataset contains GPS data regarding moving cabs in Fortaleza city. Figure 12 presents a visual representation of sample data from taxi trajectories. Since, each taxi driver at the *Taxi Simples* company has a mobile phone equipped with a GPS device, this dataset stores information about the location of taxi drivers. We use the data collected in two periods: Dec/2015 and Dec/2016, summing up 30,971,689 GPS Points from trajectories referring to travels cab

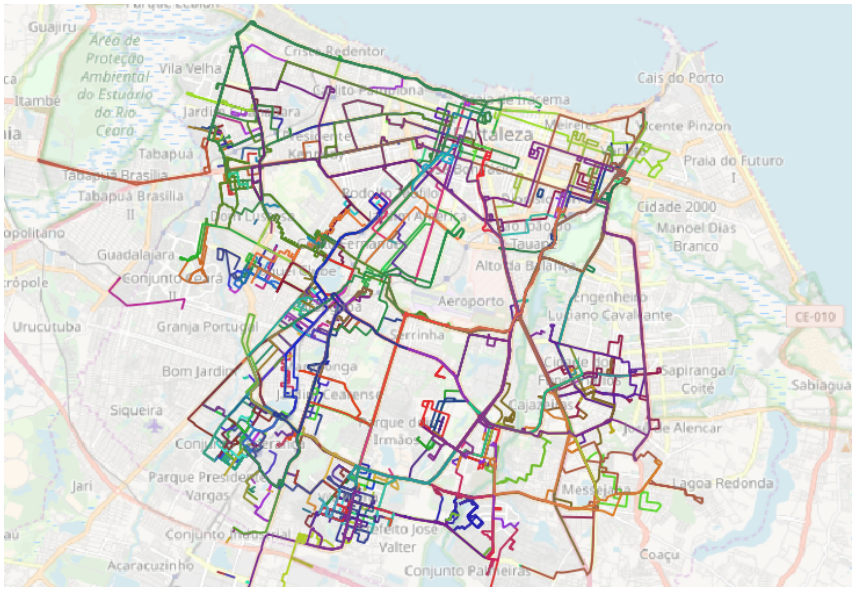

Figure 12 – A sample of taxi trajectories from Fortaleza (Brazil) on 12th Dec 2015. Each colored line represents a trajectory.

around the city. Table 15 details the number of samples considered in each step of the data cleaning and transformation processes presented in this chapter. Each record of taxi trajectories has the following attributes: trajectoryId, latitude, longitude, and timestamp.

Due to differences in the characteristics of the data collected in distinct years, which we detail in the following, we have decided to split the whole dataset into two partitions corresponding to different periods of time with a gap of one year between each other. The first partition (ds_{2015}) was collected from 1st to 31st December of 2015 and has 22,727,694 GPS points, where 97.3% of them are from December, 12th. The other partition (ds_{2016}) comprises data from 1st to 31st December of 2016. It has 8,243,995 GPS points, where 97.25% of the data concerns the 1st and 2nd of December. Besides the temporal distance between both partitions, they also keep distinct characteristics that we discuss in the following.

The **average sampling rate** of ds_{2015} is 0.573s for 96% of its data, that is, the interval between consecutive GPS points is around 0.573s on average. While the average sampling rate of ds_{2016} is 4.204s for 96% of its data. Another difference from both dataset partitions concerns the **meaning of the trajectory id**. The trajectory id in ds_{2015} is represented by the taxi identification and it has only 718 unique ids, while in ds_{2016} the trajectory id is the identification of a taxi trip and it has 294,418 unique ids. In other words, ds_{2015} collapses many taxi trips into a single trajectory id. Such difference brings implications to the results as we demonstrate in the following sessions. We illustrate the aforemetioned differences between ds_{2015} and ds_{2015} in Table 14.

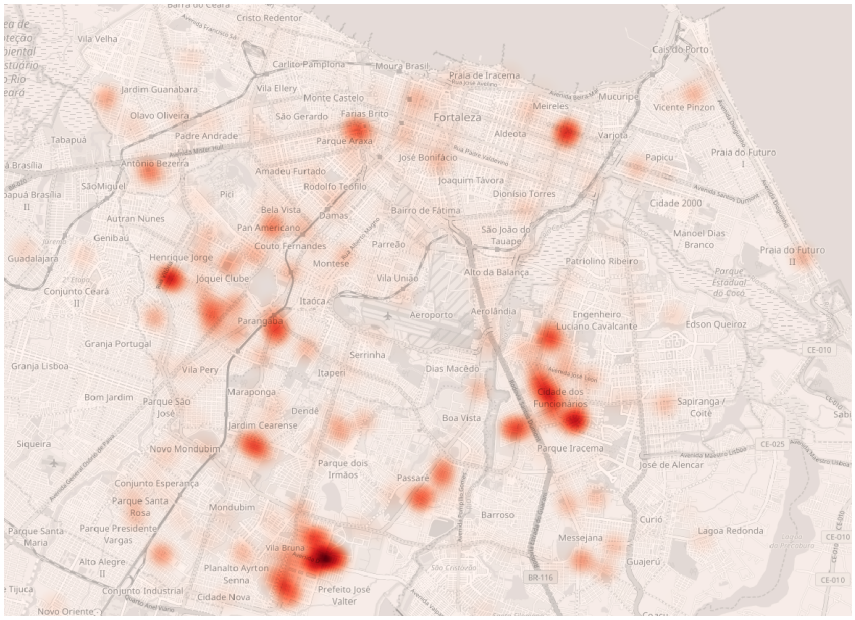

Figure 13 – Heatmap of GPS positions from the Taxi Simples dataset on 12th Dec 2015 from 10:00 to 11:00AM.

Figure 13 shows a heatmap of GPS positions on 12th Dec 2015 from 10:00 to 11:00 AM, which is the hour of the day with more GPS data.

We report that regarding the sensor data, we follow the same cleaning and pre-processing steps defined in Section 3.2. However, we define different steps to pre-process the trajectory dataset.

	Dec/2015	**Dec/2016**
Sampling Rate	0.573s	4.204s
#Trajectory ids in raw data	718	294,418
#Trajectory ids before splitting	703	186,527
#Trajectory ids after splitting	2,466	203,520

Table 14 – Differences between partitions of the raw trajectory dataset.

Step	Data Description	**# Samples**		
		Dec/2015	**Dec/2016**	**Total**
Raw Data	GPS Points	22,727,694	8,243,995	30,971,689
Cleaning	GPS Points	6,725,109	4,810,881	11,535,990
Map-matching	Snapped Points + Nodes/Edges	3,634,900	29,261,068	32,895,968
Add features	Edges + features	917,761	12,504,670	13,422,431
Feature Aggregation / Test Set	Edges + aggregated features. Test Set built from trajectory data.	155,403	2,028,049	2,183,452

Table 15 – Number of samples and processing steps regarding the Taxi Simples Dataset.

4.2.2 Data cleaning and transformation

In the data cleaning and transformation step, we remove outliers and perform some updates to the trajectory dataset to improve the data quality and to achieve more accurate results. We have found the following main issues in our raw trajectory data: (i) trajectory data with wrong timestamps (e.g., timestamps from years 1969, 1970 and 2055); (ii) duplicated data; (iii) insufficient trajectory data regarding road network edges, so that, we do not have enough data to aggregate features in edges of the road network; (iv) trajectories from the 2015 partition are identified by the taxi id and represent many trips of one taxi. Conversely, trajectories from the 2016 partition are uniquely identified by taxi' trips. In the following, we detail the data cleaning strategy adopted in this work.

4.2.2.1 Cleaning

The main goal of the data cleaning process is to remove outliers or noisy points, that is, points added in the trajectory because of a GPS error or any other kind of error. Therefore, we perform the following five cleaning steps:

(i) Filter out samples with invalid timestamps or from years with an insufficient amount of data. We keep only the data from 2015 and 2016 because the remaining data has invalid timestamps or a small amount of data. From this step on we split the dataset into two partitions (i.e., partitions with data from 2015 and 2016) and apply the following steps in each one of them.

(ii) Select an area and remove trajectory records placed out of it. We keep only the GPS coordinates in the bounding box of Fortaleza (Brazil). A bounding box (bbox) is an area defined by a quadruple that consists of two longitudes and two latitudes as follows: (minimum Longitude, minimum Latitude, maximum Longitude, maximum Latitude). We use the bbox (-38.67, -3.90, -38.38, -3.68) to limit the area of Fortaleza.

(iii) Remove duplicate records. We order the trajectory dataset by trajectory id and timestamp attributes, and later we remove consecutive duplicate records. Table 15 shows that the cleaning step changes the number of samples from 22,727,694 to only 6,725,109 in the partition representing Dec/2015 (ds_{2015}). The duplicate removal step is responsible for most of the records

(15,815,159) removed from ds_{2015} in the cleaning phase. The high sampling rate of ds_{2015} adds a lot of duplicate records that are removed.

(iv) Split trajectories based on the inference of new trips inside each trajectory. We split trajectories, i.e., define a new trajectory id when the delta time between consecutive points surpasses a given threshold (e.g., 15 minutes). This step is especially useful when the trajectory is identified by the vehicle id and, therefore, the dataset has very long trajectories. This situation occurs in the 2015 partition of the *Taxi Simples* dataset (Table 14). Trajectories from that partition usually mix multiple taxi trips under the same trajectory id represented by the identification of the vehicle. Table 16 shows the amount of trajectory ids before and after splitting in the partitions ds_{2015} and ds_{2016}. Very long trajectories demands for much more memory footprint then shorter ones, because some steps require processing of the whole trajectory data. Besides, the mix of multiple trips under one trajectory id virtually creates non-existing paths/connections between the different trips. In other words, the last point of a trip is considered connected to the starting point of another trip as they are under the same trajectory id. The main goal of the splitting step is to avoid such problems and to give the semantics of trajectory ids represented by trips to any dataset. At the end of the splitting step, Table 16 shows that the number of trajectory ids increased a lot for ds_{2015} and ds_{2016}, but there is still a huge difference in the number of trajectories between them. We hypothesize that such difference may affect subsequent steps and the final prediction results.

#Trajectory ids	**Dec/2015**	**Dec/2016**
Before splitting step (iv)	703	186,527
After splitting step (iv)	2,466	203,520

Table 16 – Number of trajectory ids before and after the splitting step – cleaning step (iv).

(v) Remove very short trajectories. We need at least two points in a trajectory to compute features like distance, delta time and speed between points. Therefore, we remove trajectories with a single point to avoid waste of resources to process and store them for subsequent steps.

After the aforementioned cleaning steps the trajectory dataset is comprised of: 11,535,990 GPS points (6,725,109 for 2015 and 4,810,881 for 2016).

4.2.2.2 Map matching

A key element in our strategy to aggregate features in different locations of a road network is to identify the sequence of edges associated with the GPS points derived from c previous cleaning steps. This is possible through a technique called map matching.

In summary, map matching is a process to convert a sequence of raw latitude/longitude coordinates to a sequence of road segments (ZHENG, 2015).

We use *GraphHopper*[2], a robust, fast and widely adopted open-source implementation of a map matching algorithm that employs a HMM to find the most likely road route represented by a time-stamped sequence of latitude/longitude pairs (NEWSON; KRUMM, 2009; KARICH; SCHRÖDER, 2014). The HMM accounts for measurement noise and the layout of the road network. The map matching procedure associates each point in the original trajectory to a position in the underlying road network. It also maps each matched point to an edge of the road network. The final result in a trajectory from point A to point B is a path p with all the edges traversed from A to B as well as all the timestamps and points matched from the original GPS points. Map matching consists of matching measured locations to the road network in order to infer the vehicle's actual path.

The *GraphHopper* map matching algorithm works as follows. For each input GPS position, a number of map matching candidates within a certain radius around the GPS position is computed. The Viterbi algorithm is used to compute the most likely sequence of map matching candidates. The Viterbi algorithm (VITERBI, 1967) uses dynamic programming to quickly find the path through the lattice that maximizes the product of the measurement probabilities and transition probabilities. This gives an inference of the correct road segment for each location measurement. Thereby, the distances between GPS positions and map matching candidates as well as the routing distances between consecutive map matching candidates are taken into account. The GraphHopper routing engine is used to find candidates and to compute routing distances.

In this work, the inputs for the map matching algorithm are the road network, as well as the trajectories, where each trajectory contains the trajectory id, GPS points, and respective timestamps.

Table 17 shows the number of elements before and after the map matching process. The mix of trips under the same trajectory reported in Section 4.2.1 and 4.2.2.1 affect negatively

[2] https://github.com/graphhopper/map-matching

the map matching results. We report that the number of GPS Points converted to Matched Points decreases 15 times (from 6,725,109 to 445,708) in ds_{2015}, but only 3,5 times (from 4,810,881 to 1,375,228) in ds_{2016}. The reason is that many non-existing connections between trips under the same trajectory result in difficulties to match points. We observe that the number of trajectory ids remains the same in ds_{2016}, while we lose 237 (2,466 - 2,229) trajectories in ds_{2015} after the map matching. We also hypothesize that the final quality of the map matching is not good for ds_{2015}.

	Map matching input		**Map matching output**		
	#GPS Points	**#Trajectory ids**	**#Snapped Points**	**#Nodes**	**#Trajectory ids**
Dec/2015	6,725,109	2,466	445,708	3,189,192	2,229
Dec/2016	4,810,881	195,660	1,375,228	27,885,840	195,660

Table 17 – Number of elements regarding map matching input and output.

We need to highlight and describe some details about the outputs from the GraphHopper map matching because they will be used in the following steps. In summary, the map matching process outputs trajectory paths containing a sequence of edges with information about their nodes and matched points. Each edge is annotated with edge id, length in meters from start to end node (d_e – edge distance) and OSM Way Id. The OSM Way Id is the identification of the road segment in the OSM service[3], and it is useful to find some attributes like the number of lanes and the speed limit of the road segment. The matched points are annotated with their timestamps. The nodes are annotated with their Latitude and Longitude. However, they are not associated with a timestamp. Therefore, we apply the linear interpolation/extrapolation method to compute the timestamp in nodes as we detail in Section 4.2.2.3.

4.2.2.3 Finding speeds on the edges of each trajectory after map matching

In this section, we propose a sequence of steps to find speeds on the edges of each trajectory yielded from the map matching approach (Section 4.2.2.2). The steps follow a pipeline that receives as input the resulting output of the previous step.

First, we need to find the timestamps in the start and end nodes of each edge. To achieve this intermediate goal, for each trajectory we perform the following:

- Augment the samples of the trajectories with the following attributes: t_{delta} (time delta), d_{delta} (distance delta) and speed s between adjacent matched points or nodes, where $s = d_{delta}/t_{delta}$.

[3] https://www.openstreetmap.org/

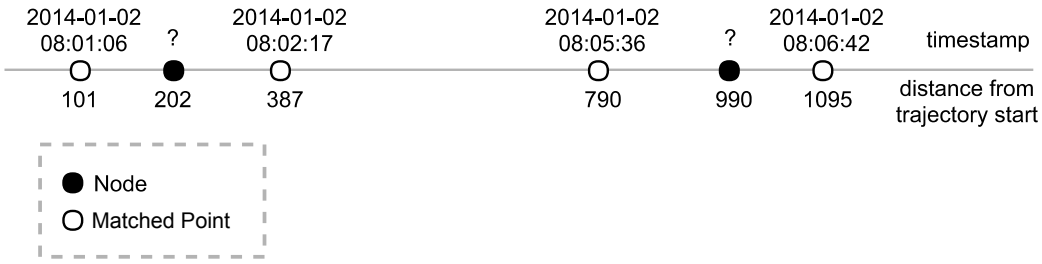

Figure 14 – Linear interpolation/extrapolation using timestamps and distances from trajectory start to find the timestamps of nodes represented with question marks.

- Detect transitions between stops and moves of vehicles, and split trajectories, when those transitions are inferred[4]. We detect the occurrence of a stop when t_{delta}, d_{delta} or s surpasses a given threshold for each of those features. Specifically, we use 750m, 200 seconds and 90km/h, respectively for t_{delta}, d_{delta} and s. Those values are computed by the mean plus 3 times the standard deviation (3σ) of each feature.
- Apply the linear interpolation/extrapolation method to find the timestamps of start and end nodes of each edge yielded from map matching. The interpolation/extrapolation inputs are the timestamps of matched points and the distance from the trajectory start to nodes / matched points. The interpolation/extrapolation method outputs the timestamp of each node, represented with question marks in Figure 14.

Second, we compute the travel time and speed of edges in each trajectory yielded from map matching. The travel time $tt_{u,v}$ associated to an edge (u, v) is the time needed to travel from u to v, that is the delta time between nodes u and v. It is given by the difference between the timestamp in node v (start node) and the timestamp in u (end node). To compute the average speed of a trajectory in edge (u, v), we divide the travel time $tt_{u,v}$ by the road distance $d_{u,v}$ between u and v.

Finally, at the end of these steps we have for each trajectory, a sequence of tuples composed of the following features: trajectory id, edge id, OSM Way Id, timestamp of start node (t_s) and speed (S). For brevity, we call such sequence of tuples as our list of edges E.

4.2.2.4 Feature engineering, sensor data aggregation and model generation.

We use the same approach described in Section 3.2 to perform feature engineering, sensor data aggregation, and model generation, except that in this chapter we consider a **set of**

[4] This step is similar to the one from Section 4.2.2.1, but now we use more parameters that we didn't have before, like distance delta and speed. Therefore, we want to remove some occurrences such as a parked car that should not be considered when determining traffic conditions.

Feature	% of missing data	
	2015	2016
n_lanes	77.3%	77.5%
speed_limit	79.9%	75.7%

Table 18 – Percentage of missing data regarding the features *n_lanes* and *speed_limit* in the partitions of data from 2015 and 2016.

features with the **15 features** in groups (i) to (iv) from our two datasets and reported in Table 3. Given that we want to generate a test set from our list of edges E (Section 4.2.2.3), we have to make it compatible with the features extracted from the aggregated data from sensors used as the training set of our prediction models. In other words, we need to generate a test set derived from trajectory data with the equivalent features used in the training set originated from sensor data.

In Section 3.2 we defined six groups of features extracted from our first dataset. Groups (iii) to (vi) are time-slot related features. Groups (v) and (vi) represent features regarding one and two weeks before the prediction time slot, respectively. Our evaluation in Section 3.3.2.1 demonstrates that the features present in groups (v) and (vi) are not so relevant as the features from other groups of time-slot related features. Besides, for many edges we don't have enough trajectory data that can be used to extract features regarding one and two weeks before the prediction time slot.

We also consider a **second set of features** consisting of 13 features that do not take into account the features number of lanes (n_lanes) and speed_limit. We decided to evaluate such a set of features because we did not find a reliable and complete dataset containing the features (n_lanes) and speed_limit. As an example, more than 75% of those features are missing in the Open Street Map (OSM) dataset regarding Fortaleza. In such case, we replace the missing data values with the mean value of each feature. Table 18 presents the percentage of missing features concerning features n_lanes and speed_limit in the OSM dataset from Fortaleza in 2015 and 2016.

Besides, we evaluate the **third set of features** that is a subset of the second set of features with only **11 features**, which suppresses the features representing the number of vehicles (v_count5 and v_count30). We hypothesize that the features v_count5 and v_count30 have different roles and values in the domains of sensors and trajectories, and therefore, their use can negatively affect the predictions. In the domain of sensors, the number of vehicles represents the number of all vehicles that pass through the sensor. However, in the trajectory domain, the number of vehicles represents only the amount of trajectories that crossed an edge in a given slot

(interval) of time and, therefore, those features in the sensor domain are orders of magnitude higher than in the trajectory domain. We highlight that because we are using the domains of sensors and trajectories, their features expose characteristics associated respectively with sensors and edges, and therefore we need to find the best possible equivalence between the features applied in both contexts. Table 19 summarizes the three sets of features evaluated in this chapter.

Group of Features	#	Acronym	Feature Name	Set of features		
				1	2	3
(i) Sensor	1	n_lanes	Sensor number of lanes	■		
	2	speed_limit	Speed limit	■		
(ii) Time reference	3	day_of_week	Day of week	■	■	■
	4	slot_of_day	Slot of day	■	■	■
	5	working_day	Working day	■	■	■
(iii) 5 min last time slot (ts_{ref})	6	v_count5	Number of vehicles	■	■	
	7	min5	Minimum speed	■	■	■
	8	max5	Maximum speed	■	■	■
	9	avg5	Average speed	■	■	■
	10	std5	Standard Deviation	■	■	■
(iv) 30 min last time slot (ts_{30})	11	v_count30	Number of vehicles	■	■	
	12	min30	Minimum speed	■	■	■
	13	max30	Maximum speed	■	■	■
	14	avg30	Average speed	■	■	■
	15	std30	Standard Deviation	■	■	■

Table 19 – Groups of features and their allocation according to the three set of features presented in this chapter. Time. Time-slot related features in groups (iii) and (iv) are highlighted, respectively, in green and blue.

In the following, we propose the steps to generate observations with the sets of features from Table 19 in the trajectory domain.

Add number of lanes and speed limit features – group (i) of features – from OSM data. After the definition of the three sets of features, we get the feature OSM Way Id from each tuple of the list of edges E, and use it to find the number of lanes and speed limit by querying the OSM data of Fortaleza through our PyRoad API project[5]. Next, we add them to each element of E.

Compute and add day of week, slot of day and working day features – group (ii) of features – to the list of edges E. They are computed from the timestamp on the start node of each

[5] https://github.com/InsightLab/PyRoad

edge.

Aggregate data in time slots and add time slot related features and label. Our speed prediction approach relies on time-dependent features regarding the speed attributes annotated in specific edges of the road network. Therefore, we aggregate the count of trajectories that traversed the edge, as well as, the minimum, maximum, average, standard deviation of speeds computed for each five and thirty minute time-slots, respectively the groups (iii) and (iv) of features concerning the interval starting at 5 and 30 minutes before the query time and ending at the query time as illustrated in Figure 15. The figure also represents our label, that is, the average speed prediction at a 5-minute time slot, ts_f, 25 minutes after the query time.

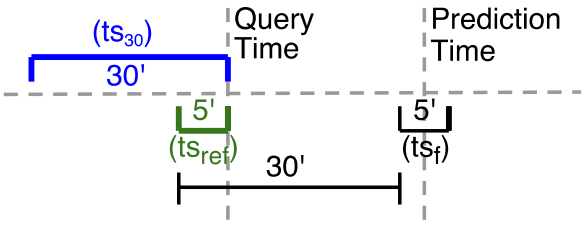

Figure 15 – Diagram illustrating time-slot related features and the label representing the average speed at the prediction time. Features concerning the interval starting at 5 and 30 minutes before the query time and ending at the query time are highlighted, respectively, in green and blue.

4.3 Experimental evaluation

In this section we discuss the experiments that we perform to generate different prediction models and assess their performance. More specifically, Section 4.3.1 introduces the experimental setting used to conduct the experiments, while Section 4.3.2 introduces the experimental questions we want to answer with our research.

4.3.1 Experimental Setting

Test system. We conduct our experiments on a server with 16 Xeon E5520 Intel CPUs, each clocked at 2.27GHz, with 8192 KB L3 cache, 24 GB of RAM, and Ubuntu OS (16.04 LTS).

Datasets. We use two main datasets regarding pre-processed and aggregated sensor and trajectory

data, respectively. We present the first one in Sections 3.3.1. We partition the whole data from 2014 in six datasets, each related to a pair of months. We also split each partition in training, validation and test sets, where each set has the features presented in Table 19. The training, validation, and test sets include respectively the 65%, 15% and 20% of the data regarding each sensor. They are created for each of our three sets of features as detailed in Section 4.2.2.4.

For each of our three sets of features, we generate the second dataset from aggregated trajectory data provided by the company *Taxi Simples*, a Brazilian cab fleet monitoring company. We use the resulting dataset as a test set comprising the structure defined in Table 19 according to the considered sets of features.

Machine learning method. The ML method considered in this chapter is GBRT, a state-of-art ML algorithm successfully applied in traffic prediction works (Chapter 2) and also the best performer algorithm evaluated in Chapter 3 using features also considered in this chapter. We apply the local and global approaches proposed in Chapter 3 to build speed prediction models from sensor data for the three subsets of the features reported in Section 4.2.2.4. The cluster-based approach is out of the scope of this chapter because it would require additional steps and analysis that are not in the main concerns of this chapter. However, we evaluate the cluster-based approach in comparison with the other approaches for different sets of features in Appendix A. Such evaluation is limited to sensor data, and therefore it does not use trajectory data. We also plan to exploit the cluster-based approach in future works.

For what concerns the hyperparameters needed by GBRT, we determine the best combination using a grid search. The first hyperparameter required by GBRT is the maximum tree depth (max_depth) – to this end, we consider the range [3, 9] (with step 2). GBRT also require the number of trees (n_estimators) to be used in a model; to this end the implementation of GBRT employs an early stopping technique to find out the best value. Finally, GBRT requires a third hyperparameter, the learning rate (learning_rate), for which we consider the range [0.05, 0.2] (with step 0.05). Overall, for each possible combination we generate a model and pick the one whose model yields the lowest MSE. The evaluation is conducted over the validation set.

4.3.2 Experimental questions and experiments

We propose a batch of experiments to comprehensively answer the following experimental questions:

EQ1 Can a global speed prediction model built with a selected subset of features yield similar evaluation performance to the global model built with the set of features used in Chapter 3?

EQ2 Can the global approach built from sensor data perform accurate predictions over road segments covered exclusively by trajectories?

In the following sections we propose experiments to answer the above questions.

4.3.3 EQ1 – Compare the evaluation of the global speed prediction approach built from the whole set of features with the one built from a selected subset of features

We want to know if global models built using subsets of features can achieve similar evaluation results to a model built with all the features. We consider only sensor data regarding EQ1.

From Chapter 3 we already have the global GBRT model (m_{all}) and evaluation results regarding the whole set of features presented in Table 3. So additionally we need to generate and evaluate GBRT models for the three sets of features defined in Section 4.2.2.4 that use the selected subsets of the features shown in Table 19.

We perform two batch of experiments concerning $EQ1$ to check the feasibility of reducing the dimensionality of our sensor dataset and use it as well as the global models derived from it in the subsequent experiments. In the first batch of experiments we consider a static network of sensors, while the second one takes into account a dynamic network of sensors.

Both batches of experiments compare the evaluation of the complete set of 25 features detailed in Chapter 3 with the subsets of 15, 13 and 11 features, respectively related with the sets of features 1, 2 and 3 defined in this chapter. The results are presented in Tables 20 and 21. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features. Winners are highlighted in blue.

We report that the global approach performs better when using the whole set of 25 features. It is also the winner when we consider the percentage of sensors that perform better using the global approach rather than the local approach. When considering only the sets of features with 15, 13 and 11 features respectively, we observe that more features usually yields slightly better results for the static network of sensors. However, the subset of 11 features performs better in the dynamic network of sensors in comparison with the subsets of 15 and

13 features. Another insight regarding the dynamic network of sensors is that the predictions improve when we decrease the number of features in the sets of features with 15, 13 and 11 features. The conclusion is that the four features removed (n_lanes, speed_limit, v_count5, and v_count10) were affecting negatively the results when we consider the dynamic network of sensors.

Overall the results indicate that even using fewer features in the three sets of features that we consider in this chapter, we still get acceptable prediction results, and therefore it is feasible to evaluate if the prediction models following the global approach are also effective for achieving accurate predictions concerning test sets built from trajectory data. In other words, it is feasible to find answers to our experimental question 2 ($EQ2$).

We present a complementary evaluation comparison between the use of different sets of features in Appendix A, including the cluster-based approach and also sets of features not covered in this chapter. Appendix A also presents some situations where it may be feasible to remove some features, even when getting slightly worse prediction results.

4.3.4 EQ2 – Evaluate the global approach built from sensor data in road segments covered exclusively by trajectories

To answer $EQ2$, we compare the evaluation of the prediction models trained from sensor data yielded in $EQ1$ with the performance of the same models applied to perform predictions over test sets built from trajectory data for the three sets of features. We evaluate the global approach built from sensor data, comparing the predicted speed in edges from our test sets with the real average speed in edges extracted from the trajectory data.

Table 23 shows that the test set of Dec/2016 (ds_{2016}) performs much better than the test set of Dec/2015 (ds_{2015}). We have reported in Section 4.2.2.2 the issues with the map matching quality for ds_{2015}. Those issues hindered subsequent steps and affected its evaluation negatively. Therefore we plan, as future work, to improve the detection of trips and consequently the splitting of trajectories before map-matching.

We also observe that in all sets of features and test sets, the results improve as we decrease the number of features from 15 to 11. The four features removed (n_lanes, speed_limit, v_count5, and v_count10) were affecting negatively the results.

In the domain concerning sensors, the features v_count5 and $v\_count10$ represent all the vehicles crossing a road segment monitored by a sensor in a given time slot. On the other

MSE - Local						MSE - Global			
		#Features				#Features			
Months	#Sensors	25	15	13	11	25	15	13	11
Jan/Feb	235	10.80	10.71	**10.70**	10.75	**11.62 (9.8%)**	12.30 (3.0%)	12.58 (2.6%)	12.78 (3%)
Mar/Apr	181	13.56	**13.22**	**13.22**	13.27	**13.08 (51.4%)**	13.38 (25.4%)	13.58 (22.7%)	13.67 (22.1%)
May/Jun	217	13.90	13.72	13.71	**13.67**	**12.27 (74.7%)**	12.82 (42.9%)	12.87 (35.5%)	13.01 (35%)
Jul/Ago	214	14.16	14.16	14.14	**13.94**	**11.82 (62.6%)**	12.82 (38.3%)	12.9 (35.5%)	12.99 (33.2%)
Sep/Oct	207	15.28	15.41	15.40	**15.10**	**12.20 (66.7%)**	13.27 (43.5%)	13.35 (38.2%)	13.53 (34.8%)
Nov/Dec	204	15.14	15.22	15.19	**14.93**	**11.68 (73.0%)**	12.68 (51.0%)	12.82 (45.1%)	12.95 (43.1%)

MAPE - Local						MAPE - Global			
		#Features				#Features			
Months	#Sensors	25	15	13	11	25	15	13	11
Jan/Feb	235	0.0733	**0.0730**	**0.0730**	0.0732	**0.0769 (3.8%)**	0.0794 (1.3%)	0.0804 (0.9%)	0.081 (0.9%)
Mar/Apr	181	0.0877	**0.0868**	**0.0868**	0.0869	**0.0866 (42.0%)**	0.0877 (27.1%)	0.0882 (23.8%)	0.0884 (24.3%)
May/Jun	217	0.0841	0.0841	0.0841	**0.0839**	**0.0794 (59.0%)**	0.0819 (36.9%)	0.0817 (36.4%)	0.082 (35.5%)
Jul/Ago	214	0.0835	0.0838	0.0838	**0.0832**	**0.0773 (57.5%)**	0.0810 (37.9%)	0.0809 (34.6%)	0.081 (31.3%)
Sep/Oct	207	0.0883	0.0886	0.0886	**0.0878**	**0.0788 (65.2%)**	0.0830 (42.5%)	0.0825 (39.1%)	0.083 (38.2%)
Nov/Dec	204	0.0840	0.0843	0.0843	**0.0836**	**0.0739 (70.1%)**	0.0772 (50.5%)	0.0771 (47.5%)	0.0775 (44.1%)

Table 20 – Evaluation Comparison between different sets of features considering a static network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). Winners are highlighted in blue.

MSE - Local						MSE - Global			
		#Features				#Features			
Months	#Sensors	25	15	13	11	25	15	13	11
Jan/Feb	0	-	-	-	-	-	-	-	-
Mar/Apr	0	-	-	-	-	-	-	-	-
May/Jun	2	**17.34**	18.62	18.62	21.16	**14.92 (100%)**	15.62 (100%)	15.71 (100%)	15.56 (100%)
Jul/Ago	10	**10.97**	11.52	11.60	11.63	**10.81 (60%)**	11.61 (40%)	11.61 (60%)	11.63 (60%)
Sep/Oct	26	**15.47**	16.09	16.37	16.49	**15.49 (50%)**	17.14 (30.8%)	16.77 (38.5%)	16.74 (46.2%)
Nov/Dec	52	**15.87**	16.85	16.50	16.83	**15.48 (63.5%)**	17.41 (34.6%)	16.89 (42.3%)	16.86 (53.8%)

MAPE - Local						MAPE - Global			
		#Features				#Features			
Months	#Sensors	25	15	13	11	25	15	13	11
Jan/Feb	0	-	-	-	-	-	-	-	-
Mar/Apr	0	-	-	-	-	-	-	-	-
May/Jun	2	**0.1442**	0.1510	0.1510	0.1620	0.1271 (100%)	0.1284 (100%)	0.129 (100%)	**0.1259 (100%)**
Jul/Ago	10	**0.0766**	0.0780	0.0786	0.0787	**0.0756 (80%)**	0.0793 (40%)	0.0791 (50%)	0.079 (50%)
Sep/Oct	26	**0.0895**	0.0913	0.0919	0.0920	**0.0893 (61.5%)**	0.0949 (26.9%)	0.0924 (38.5%)	0.0923 (53.8%)
Nov/Dec	52	**0.0876**	0.0905	0.0897	0.0906	**0.0866 (65.4%)**	0.0927 (28.8%)	0.0897 (44.2%)	0.0898 (55.8%)

Table 21 – Evaluation Comparison between different sets of features considering a dynamic network of sensors in different pairs of months of 2014. The ML technique used is GBTR. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features). Winners are highlighted in blue.

hand, in the trajectory domain, they represent only the amount of the available trajectories that cross a road segment in a given time slot, that is, not all vehicles, but only the ones that we know. Our conclusion based on those two features is that the compatibility of features between the two domains (sensor and trajectory) must be carefully analyzed and that features with different semantics/roles between domains should not be used.

Regarding the features *n_lanes* and *speed_limit*, it is not worth using them in the trajectory domain, because, in more than 75% of our data (Table 18), we could not get reliable values for them from OSM. Therefore, we use average values for those missing data in the trajectory domain, but real and reliable values for them in the sensor domain. Such a discrepancy in the robustness/reliability of the data for some features between domains affect the prediction results negatively and, therefore, we conclude that such kind of features should not be considered.

We also report that even our best results in the set of features with 11 features (2016), still have to improve a lot to be competitive in comparison to use the global approach in the sensor domain. We plan to do a per edge detailed analysis to understand better in which conditions or locations we obtain feasible results. We can also use other datasets of sensors and trajectories within the same period. In this chapter, we used sensor data from Jan-Feb/2014 to build the prediction models and trajectory data of Dec/2015 and Dec/2016. We also have to solve possible incompatibilities between domains, like always having speed limits in roads monitored by sensors, but not always having such limits on roads traversed by vehicles. Some drivers can also have one behavior when crossing a road monitored by a sensor, but another behavior in roads not monitored.

Besides, we propose a new batch of experiments to verify if we can generate more accurate prediction models using solely trajectory data than the prediction models following our cross-domain approach. To this end, for each of the three sets of features considered in this chapter, we generate a training set, validation set and test set, comprising respectively, 65%, 15% and 20% of the trajectory data for Dec/2015 and Dec/2016. In previous experiments with trajectory data, we have used the whole trajectory data as test sets for Dec/2015 and Dec/2016. We report the results in Table 23. We show that prediction models generated only with trajectory data achieve better results than using our cross-domain strategy. However, as 80% of the trajectory data is used for training and validation purposes, then such strategy cannot be used to solve the cold start problem. The superior results of models based only in the trajectory domain indicate that the cross-domain strategy still needs improvements in the compatibility between

Set	#features	MSE		MAPE	
		2015	2016	2015	2016
1	15	204.69	129.03	2.56	1.35
2	13	97.92	62.75	1.67	0.85
3	11	88.17	**46.35**	1.64	**0.77**

Table 22 – Evaluation of the cross-domain approach for different sets of features using Taxi Simples trajectory datasets in Dec/2015 and Dec/2016. Winners are highlighted in **bold**.

Set	#Features	MSE				MAPE			
		Dec/2015		Dec/2016		Dec/2015		Dec/2016	
		Cross domain	Traj. domain	Cross domain	Traj. domain	Cross domain	Traj. domain	Cross domain	Traj. domain
1	15	219.19	20.56	95.17	9.55	2.6611	0.8361	0.9984	0.3187
2	13	95.12	19.51	38.39	9.30	1.6642	0.8182	0.6169	0.3167
3	11	93.34	**19.17**	38.61	**9.21**	1.9480	**0.8104**	0.6904	**0.3156**

Table 23 – Evaluation of cross-domain (sensor and edge based) versus trajectory only domain (edge based) both according to the global approach. Winners are highlighted in **bold**.

domains.

4.4 Discussion

This chapter proposed a methodology for evaluating the global approach introduced in Chapter 3 using a test set built from trajectory data. We proposed and ran a methodology to know if the prediction models generated by the global approach can provide good generalizations of traffic behaviors present in trajectory data. In the meanwhile, we have developed the PyRoad library[6] to simplify queries and visualization of trajectories and other spatial-temporal data. Our experiments demonstrate that cross-domain generalizations are not trivial and the features must be carefully analyzed to observe which are compatible with both domains and if they are relevant to help in achieving accurate results. We also conclude that the mix of vehicle trips under the same trajectory id affect the results negatively and that we need to improve the splitting of trajectories before map matching. We want to perform a detailed per edge evaluation analysis to better understand how to improve the prediction results. With the improvement of the cross-domain generalizations, we can model more accurate cost functions for Time-Dependent Road Networks (TDRN) or even build more realistic traffic simulators. A TDRN is a road network where the edge costs vary with time. The next chapter presents conclusions about this thesis and highlights some plans for future works.

[6] https://github.com/InsightLab/PyRoad

5 CONCLUSIONS AND FUTURE WORK

In this thesis, we presented and evaluated three different approaches based on state-of-the-art supervised machine learning approaches for training models for the speed prediction problem in a large and dynamic network of traffic sensors. The approaches, namely, the local, cluster-based, and global approaches have complementary advantages. The local approach which is the traditional strategy adopted by earlier works fits the dynamics of each sensor of the network and achieves high performance in the presence of rich historical data. However, it suffers the cold start problem and cannot be applied to new sensors added to the network. Moreover, using the local approach in large sensor networks involves training and maintaining a large number of different prediction models. The evaluation showed that the novel global and cluster-based approaches proposed in the chapter consistently outperform the local approach when a dynamic network of sensors is considered.

We made publicly available a very large dataset collected in the city of Fortaleza, Brazil to ensure the reproducibility of our results and promote research developments in this field.

We first analyzed state-of-the-art supervised algorithms for the speed prediction problem, and proved that GBRT algorithm outperforms the other methods experimented. We also analyzed the relevance of the features engineered for the approaches, showing that the most relevant ones are associated with information in the temporal domain.

We proposed a cross-domain methodology to use our global approach based on the sensor domain to perform traffic predictions in road segments covered by data extrated from the trajectory domain. We have developed an open-source Python library[1] to help researchers in dealing and visualizing spatial-temporal data. We detected issues and challenges to guide our next steps in improving the proposed methodology.

As future works we want to: improve pre-processing steps before map matching; perform a detailed per edge evaluation analysis to better understand how to improve the prediction results; evaluate our cluster-based approach with a test set built from trajectory data, as we did in Chapter 4 using the global approach; derive cost functions from the global and cluster-based approaches to be accurately applied in Travel Time predictions over TDRN.

[1] https://github.com/InsightLab/PyRoad

REFERENCES

ARTHUR, D.; VASSILVITSKII, S. k-means++: The advantages of careful seeding. In: SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS. **Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms**. [S.l.], 2007. p. 1027–1035.

ASIF, M. T.; DAUWELS, J.; GOH, C. Y.; ORAN, A.; FATHI, E.; XU, M.; DHANYA, M. M.; MITROVIC, N.; JAILLET, P. Spatiotemporal patterns in large-scale traffic speed prediction. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 15, n. 2, p. 794–804, 2014.

BREIMAN, L. Random forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 2001. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1023/A:1010933404324>.

CASTRO-NETO, M.; JEONG, Y.-S.; JEONG, M.-K.; HAN, L. D. Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions. **Expert systems with applications**, Elsevier, v. 36, n. 3, p. 6164–6173, 2009.

CHAN, K. Y.; DILLON, T. S.; SINGH, J.; CHANG, E. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and levenberg–marquardt algorithm. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 13, n. 2, p. 644–654, 2012.

CHEN, C. Freeway performance measurement system (pems). **California Partners for Advanced Transit and Highways (PATH)**, 2003.

CHEN, C.; SKABARDONIS, A.; VARAIYA, P. A system for displaying travel times on changeable message signs. **University of California, Berkeley**, v. 94720, p. 1720, 2003.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: ACM. **Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining**. [S.l.], 2016. p. 785–794.

CHEN, T.; HE, T. Xgboost: extreme gradient boosting. **R package version 0.4-2**, 2015.

CLARK, S. Traffic prediction using multivariate nonparametric regression. **Journal of transportation engineering**, American Society of Civil Engineers, v. 129, n. 2, p. 161–168, 2003.

ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. *et al.* A density-based algorithm for discovering clusters in large spatial databases with noise. In: **Kdd**. [S.l.: s.n.], 1996. v. 96, n. 34, p. 226–231.

FREEDMAN, D. A. **Statistical models: theory and practice**. [S.l.]: cambridge university press, 2009.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. **The elements of statistical learning**. [S.l.]: Springer series in statistics New York, 2001. v. 1.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. **Annals of statistics**, JSTOR, p. 1189–1232, 2001.

FRIEDMAN, J. H. Stochastic gradient boosting. **Computational Statistics & Data Analysis**, Elsevier, v. 38, n. 4, p. 367–378, 2002.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

HAMNER, B. Predicting travel times with context-dependent random forests by modeling local and aggregate traffic flow. In: IEEE. **2010 IEEE International Conference on Data Mining Workshops**. [S.l.], 2010. p. 1357–1359.

HARTIGAN, J. A. **Clustering algorithms**. [S.l.]: Wiley New York, 1975. v. 209.

HOOGENDOORN, S. P.; BOVY, P. H. State-of-the-art of vehicular traffic flow modelling. **Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering**, Sage Publications, v. 215, n. 4, p. 283–303, 2001.

HUANG, W.; SONG, G.; HONG, H.; XIE, K. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 15, n. 5, p. 2191–2201, 2014.

KARICH, P.; SCHRÖDER, S. Graphhopper. **<http://www.graphhopper.com>, last accessed: 2018-05-05**, v. 4, n. 2, p. 15, 2014.

KLEIN, L. A.; MILLS, M. K.; GIBSON, D. R. **Traffic Detector Handbook: Volume I**. [S.l.], 2006.

Li, Y.; Yu, R.; Shahabi, C.; Liu, Y. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. **ArXiv e-prints**, jul. 2017.

LV, Y.; DUAN, Y.; KANG, W.; LI, Z.; WANG, F.-Y. Traffic flow prediction with big data: a deep learning approach. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 16, n. 2, p. 865–873, 2015.

MAGALHÃES, R. P.; COUTINHO, G.; MACÊDO, J.; FERREIRA, C.; CRUZ, L.; NASCIMENTO, M. Graphast: an extensible framework for building applications on time-dependent networks. In: ACM. **Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems**. [S.l.], 2015. p. 93.

MILLER, M.; GUPTA, C. Mining traffic incidents to forecast impact. In: ACM. **Proceedings of the ACM SIGKDD international workshop on urban computing**. [S.l.], 2012. p. 33–40.

MIN, W.; WYNTER, L. Real-time road traffic prediction with spatio-temporal correlations. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 19, n. 4, p. 606–616, 2011.

NEWSON, P.; KRUMM, J. Hidden markov map matching through noise and sparseness. In: ACM. **Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems**. [S.l.], 2009. p. 336–343.

PAN, S. J.; YANG, Q. *et al.* A survey on transfer learning. **IEEE Transactions on knowledge and data engineering**, Institute of Electrical and Electronics Engineers, Inc., 345 E. 47 th St. NY NY 10017-2394 USA, v. 22, n. 10, p. 1345–1359, 2010.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. *et al.* Scikit-learn: Machine learning in python. **Journal of Machine Learning Research**, v. 12, n. Oct, p. 2825–2830, 2011.

ROKACH, L.; MAIMON, O. Clustering methods. In: **Data mining and knowledge discovery handbook**. [S.l.]: Springer, 2005. p. 321–352.

RZESZÓTKO, J.; NGUYEN, S. H. Machine learning for traffic prediction. **Fundam. Inf.**, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 119, n. 3-4, p. 407–420, ago. 2012. ISSN 0169-2968. Disponível em: <http://dl.acm.org/citation.cfm?id=2385625.2385635>.

SCHMITT, E. J.; JULA, H. On the limitations of linear models in predicting travel times. In: IEEE. **2007 IEEE Intelligent Transportation Systems Conference**. [S.l.], 2007. p. 830–835.

SHUAI, M.; XIE, K.; PU, W.; SONG, G.; MA, X. An online approach based on locally weighted learning for short-term traffic flow prediction. In: ACM. **Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems**. [S.l.], 2008. p. 45.

SMITH, B. L.; WILLIAMS, B. M.; OSWALD, R. K. Comparison of parametric and nonparametric models for traffic flow forecasting. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 10, n. 4, p. 303–321, 2002.

SORENSEN, P.; WACHS, M.; MIN, E. Y.; KOFNER, A.; ECOLA, L. **Moving Los Angeles: Short-term policy options for improving transportation**. [S.l.]: Rand Corporation, 2008.

SUN, S.; ZHANG, C.; YU, G. A bayesian network approach to traffic flow forecasting. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, v. 7, n. 1, p. 124–132, 2006.

VITERBI, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. **IEEE transactions on Information Theory**, IEEE, v. 13, n. 2, p. 260–269, 1967.

VLAHOGIANNI, E. I.; GOLIAS, J. C.; KARLAFTIS, M. G. Short-term traffic forecasting: Overview of objectives and methods. **Transport reviews**, Taylor & Francis, v. 24, n. 5, p. 533–557, 2004.

VLAHOGIANNI, E. I.; KARLAFTIS, M. G.; GOLIAS, J. C. Short-term traffic forecasting: Where we are and where we're going. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 43, p. 3–19, 2014.

VOORT, M. V. D.; DOUGHERTY, M.; WATSON, S. Combining kohonen maps with arima time series models to forecast traffic flow. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 4, n. 5, p. 307–318, 1996.

WANG, D.; ZHANG, Q.; WU, S.; LI, X.; WANG, R. Traffic flow forecast with urban transport network. In: IEEE. **Intelligent Transportation Engineering (ICITE), IEEE International Conference on**. [S.l.], 2016. p. 139–143.

WANG, J.; GU, Q.; WU, J.; LIU, G.; XIONG, Z. Traffic speed prediction and congestion source exploration: A deep learning method. In: IEEE. **Data Mining (ICDM), 2016 IEEE 16th International Conference on**. [S.l.], 2016. p. 499–508.

WOJNARSKI, M.; GORA, P.; SZCZUKA, M.; NGUYEN, H. S.; SWIETLICKA, J.; ZEINALIPOUR, D. Ieee icdm 2010 contest: Tomtom traffic prediction for intelligent gps navigation. In: **2010 IEEE International Conference on Data Mining Workshops**. [S.l.: s.n.], 2010. p. 1372–1376. ISSN 2375-9232.

XU, F. F.; LIN, B. Y.; LU, Q.; HUANG, Y.; ZHU, K. Q. Cross-region traffic prediction for china on openstreetmap. In: ACM. **Proceedings of the 9th ACM SIGSPATIAL International Workshop on Computational Transportation Science**. [S.l.], 2016. p. 37–42.

YU, D.; LIU, Y.; YU, X. A data grouping cnn algorithm for short-term traffic flow forecasting. In: SPRINGER. **Asia-Pacific Web Conference**. [S.l.], 2016. p. 92–103.

Zhang, J.; Zheng, Y.; Qi, D.; Li, R.; Yi, X.; Li, T. Predicting Citywide Crowd Flows Using Deep Spatio-Temporal Residual Networks. **ArXiv e-prints**, jan. 2017.

ZHANG, Y.; HAGHANI, A. A gradient boosting method to improve travel time prediction. **Transportation Research Part C: Emerging Technologies**, v. 58, Part B, p. 308 – 324, 2015. ISSN 0968-090X. Big Data in Transportation and Traffic Engineering.

ZHANG, Y.; ZHANG, Y. A comparative study of three multivariate short-term freeway traffic flow forecasting methods with missing data. **Journal of Intelligent Transportation Systems**, Taylor & Francis, v. 20, n. 3, p. 205–218, 2016.

ZHENG, Y. Trajectory data mining: an overview. **ACM Transactions on Intelligent Systems and Technology (TIST)**, ACM, v. 6, n. 3, p. 29, 2015.

APPENDIX A – EVALUATION OF DIFFERENT SETS OF FEATURES

This appendix presents an evaluation comparison between using different sets of features for our three proposed approaches: local, global and cluster-based.

We consider three sets of features. The first set of features is composed by the 25 features shown in Table 24.

Group of Features	#	Acronym	Feature Name	Set of features		
				1	2	3
(i) Sensor	1	n_lanes	Sensor number of lanes			
	2	speed_limit	Speed limit			
(ii) Time reference	3	day_of_week	Day of week			
	4	slot_of_day	Slot of day			
	5	working_day	Working day			
(iii) 5 min last time slot (ts_{ref})	6	v_count5	Number of vehicles			
	7	min5	Minimum speed			
	8	max5	Maximum speed			
	9	avg5	Average speed			
	10	std5	Standard Deviation			
(iv) 30 min last time slot (ts_{30})	11	v_count30	Number of vehicles			
	12	min30	Minimum speed			
	13	max30	Maximum speed			
	14	avg30	Average speed			
	15	std30	Standard Deviation			

Table 24 – Groups of features and their allocation according to the three set of features presented in this chapter. Time. Time-slot related features in groups (iii) and (iv) are highlighted, respectively, in green and blue.

The second set of features is a subset of the first one containing 15 features from the groups of features (i) to (iv). Groups (v) and (vi) are not considered.

Finally, the third set of features is a subset of the second one containing 10 features from the groups of features (i) to (iii).

The evaluation shows the effectiveness of using all the 25 features proposed in chapter 3 in the three approaches. We can also observe that even removing ten features from 25 to 15, the results are only slightly worse. However, when we move from 15 to 10 features, the results become much worse, demonstrating that it is not feasible to use only those ten features. However, although the set of 25 features is clearly the winner, the subset of 15 features may be a feasible option if: (i) the computational cost (space and time) to compute and store the features

from one and two weeks before the prediction time is high; (ii) we do not have enough sensor data to compute those features that requires at least two weeks of data before the initial query time; (iii) we want to overcome the cold start problem by not having to collect at least two weeks of data before creating a prediction model that depend on those features.

MSE - Local				
		#Features		
Months	#Sensors	25	15	10
Jan/Feb	235	10.80	10.71	11.02
Mar/Apr	181	13.56	13.22	13.86
May/Jun	217	13.90	13.72	14.43
Jul/Ago	214	14.16	14.16	14.51
Sep/Oct	207	15.28	15.41	15.95
Nov/Dec	204	15.14	15.22	15.82

MSE - Global		
#Features		
25	15	10
11.62 (9.8%)	12.30 (3.0%)	15.65 (0.9%)
13.08 (51.4%)	13.38 (25.4%)	17.00 (9.9%)
12.27 (74.7%)	12.82 (42.9%)	16.41 (17.5%)
11.82 (62.6%)	12.82 (38.3%)	16.24 (18.7%)
12.20 (66.7%)	13.27 (43.5%)	16.64 (26.1%)
11.68 (73.0%)	12.68 (51.0%)	15.72 (26.5%)

MSE - Cluster-Based (k=2)		
#Features		
25	15	10
11.43 (13.6%)	11.98 (4.7%)	14.52 (1.3%)
12.99 (61.3%)	13.18 (30.4%)	15.91 (13.8%)
12.20 (74.2%)	12.69 (44.2%)	15.49 (19.8%)
11.81 (65.9%)	12.77 (43.0%)	15.39 (20.1%)
12.17 (69.6%)	13.12 (45.4%)	15.75 (28.0%)
11.74 (76.0%)	12.66 (53.4%)	15.03 (28.4%)

MAPE - Local				
		#Features		
Months	#Sensors	25	15	10
Jan/Feb	235	0.0733	0.0730	0.0741
Mar/Apr	181	0.0877	0.0868	0.0895
May/Jun	217	0.0841	0.0841	0.0870
Jul/Ago	214	0.0835	0.0838	0.0859
Sep/Oct	207	0.0883	0.0886	0.0915
Nov/Dec	204	0.0840	0.0843	0.0876

MAPE - Global		
#Features		
25	15	10
0.0769 (3.8%)	0.0794 (1.3%)	0.0913 (0.4%)
0.0866 (42.0%)	0.0877 (27.1%)	0.1009 (13.3%)
0.0794 (59.0%)	0.0819 (36.9%)	0.0947 (16.1%)
0.0773 (57.5%)	0.0810 (37.9%)	0.0932 (15.9%)
0.0788 (65.2%)	0.0830 (42.5%)	0.0953 (23.2%)
0.0739 (70.1%)	0.0772 (50.5%)	0.0887 (27.5%)

MAPE - Cluster-Based (k=2)		
#Features		
25	15	10
0.0764 (4.3%)	0.0786 (1.7%)	0.0884 (0.4%)
0.0862 (51.4%)	0.0871 (29.8%)	0.0983 (14.9%)
0.0791 (61.8%)	0.0813 (38.2%)	0.0924 (18.9%)
0.0773 (61.2%)	0.0807 (38.8%)	0.0910 (16.4%)
0.0788 (64.7%)	0.0825 (46.4%)	0.0931 (26.6%)
0.0741 (71.6%)	0.0773 (51.0%)	0.0871 (29.4%)

Table 25 – Evaluation Comparison between different sets of features considering a static network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features).

MSE - Local				
		#Features		
Months	#Sensors	25	15	10
Jan/Feb	0	-	-	-
Mar/Apr	0	-	-	-
May/Jun	2	17.34	18.62	26.81
Jul/Ago	10	10.97	11.52	12.90
Sep/Oct	26	15.47	16.09	19.13
Nov/Dec	52	15.87	16.85	20.00

MSE - Global		
#Features		
25	15	10
-	-	-
-	-	-
14.92 (100%)	15.62 (100%)	18.72 (100%)
10.81 (60%)	11.61 (40%)	14.63 (10%)
15.49 (50%)	17.14 (30.8%)	21.38 (11.5%)
15.48 (63.5%)	17.41 (34.6%)	22.31 (21.2%)

MSE - Cluster-Based (k=2)		
#Features		
25	15	10
-	-	-
-	-	-
14.61 (100%)	15.12 (100%)	18.08 (100%)
10.79 (60%)	11.59 (50%)	14.4 (10%)
15.85 (42.3%)	17.54 (30.8%)	22.22 (26.9%)
15.71 (57.7%)	17.45 (38.5%)	22.26 (38.5%)

MAPE - Local				
		#Features		
Months	#Sensors	25	15	10
Jan/Feb	0	-	-	-
Mar/Apr	0	-	-	-
May/Jun	2	0.1442	0.1510	0.1852
Jul/Ago	10	0.0766	0.0780	0.0837
Sep/Oct	26	0.0895	0.0913	0.1012
Nov/Dec	52	0.0876	0.0905	0.1005

MAPE - Global		
#Features		
25	15	10
-	-	-
-	-	-
0.1271 (100%)	0.1284 (100%)	0.1434 (100%)
0.0756 (80%)	0.0793 (40%)	0.0915 (10%)
0.0893 (61.5%)	0.0949 (26.9%)	0.1088 (15.4%)
0.0866 (65.4%)	0.0927 (28.8%)	0.107 (25%)

MAPE - Cluster-Based (k=2)		
#Features		
25	15	10
-	-	-
-	-	-
0.1249 (100%)	0.1252 (100%)	0.1397 (100%)
0.0753 (80%)	0.0792 (50%)	0.0904 (10%)
0.0903 (53.8%)	0.0957 (26.9%)	0.1085 (30.8%)
0.0866 (65.4%)	0.0919 (40.4%)	0.105 (42.3%)

Table 26 – Evaluation Comparison between different sets of features considering a dynamic network of sensors in different pairs of months of 2014. The ML technique used is GBRT. The percentages in parentheses represent the fraction of sensors for which the global approach performs better than the local one for the same subset of features (#Features).