Speed Prediction in Large and Dynamic Traffic Sensor Networks

Regis Pires Magalhaes[a,*], Francesco Lettich[a,d], Jose Antonio Macedo[a],
Franco Maria Nardini[b], Raffaele Perego[b], Chiara Renso[b], Roberto Trani[b,c]

[a] *Computer Science Department, Federal University of Ceará, Fortaleza, Brazil*
[b] *ISTI-CNR, Pisa, Italy*
[c] *Computer Science Department, University of Pisa, Pisa, Italy*
[d] *Computing Science Department, University of Alberta, Edmonton, Canada*

Abstract

Smart cities are nowadays equipped with pervasive networks of sensors that monitor traffic in real-time and record huge volumes of traffic data. These datasets constitute a rich source of information that can be used to extract knowledge useful for municipalities and citizens. In this paper we are interested in exploiting such data to estimate future speed in traffic sensor 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 thus requires smart solutions to collect vast volumes of data and train effective prediction 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, where the cold start problem represents an important challenge. 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 ~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.

Keywords: Smart Cities, Intelligent transportation systems, Short-term Traffic Prediction, Dynamic Sensor Networks, Machine Learning, Urban Mobility.

1. Introduction

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 [15], 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 [29]. However, the emergence of smart cities, where urban areas are covered by massive amounts of sensors, combined with the development of transportation technologies, requires traffic prediction techniques that are fast, scalable, and suitable for complex and heterogeneous sensors networks like those deployed in smart cities. 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 [17].

In this work we focus on sensors capable of capturing the speed of vehicles traveling over large and dynamic road networks, where sensors can be added or removed from the network for various reasons, and 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 the *network dynamicity* problem. In this context we assume that sensors collect their observations in the form

*Corresponding author
Email addresses: regismagalhaes@ufc.br (Regis Pires Magalhaes), flettich@ualberta.ca (Francesco Lettich), jose.macedo@lia.ufc.br (Jose Antonio Macedo), f.nardini@isti.cnr.it (Franco Maria Nardini), r.perego@isti.cnr.it (Raffaele Perego), c.renso@isti.cnr.it (Chiara Renso), r.trani@isti.cnr.it (Roberto Trani)

of textual data and periodically send such information to a centralized entity. We also assume that some centralized entity is in charge of training prediction models according to the available sensor observations.

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 may be continuously added and removed on a daily basis. 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 in terms of reduced complexity and costs. Furthermore, by relying on a single prediction function that is independent from specific sensors, the global approach naturally solves the *cold start* problem. Moreover, the global approach is expected to be robust with respect to structural changes occurring in sensor networks, thus also addressing the *dynamicity* problem.

We also tested a cluster-based approach to prove its potential in representing a viable compromise between the local and global approaches. Specifically, the cluster-based approach trains distinct predictive functions for groups of similar sensors, where sensors are clustered according to some similarity metric; depending on the number of clusters, the behavior of this approach resembles the one of the local approach (when a high number of clusters is used) or the behavior of the global one (when few clusters are used). From the experimental evaluation we cannot conclude yet that this approach indeed represents a good compromise, since results are discordant and further work is needed. The contributions of this paper 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 approach 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 paper is structured as follows: Section 2 reports an overview of the related works dealing with the traffic prediction problem. Section 3 defines our prediction problem and discusses three approaches to solve the problem. Section 4 presents the dataset used in our experiments, as well as the preprocessing steps used to transform the data into a format suitable for speed prediction. Section 5 details the experimental evaluation and discusses the results. Finally, Section 6 draws the final conclusions and sketches potential lines of future research.

2. Related Work

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 [29] 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 [31] 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 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 [14]. The authors employ two kind 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 learned 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 [16, 19, 34] 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 traffic flow and speed prediction solutions [30, 34] use models based on gradient boosting regression trees (GBRT), as this technique proves to be superior to AutoRegressive Integrated Moving Average (ARIMA), Support Vector Machines (SVM), and Random Forests. 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 [16, 20]. A deep learning architecture composed of a deep belief network (DBN) and a multitask regression layer is proposed in [16]. 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 ARIMA [27], Bayesian [26], Support Vector Regression (SVR) [3], Locally Weighted Learning (LWL) [25], Multivariate Nonparametric Regression (MNR) [9], and neural networks [4]. [32] proposes a data grouping approach based on a convolutional neural network called 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 SAE (stacked autoencoder) [20]. In [18] 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 [33] the authors propose 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 outperforms those using small training dataset. Although a similar investigation is out of the scope of this paper, 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.

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

$$\hat{f} = \arg\min_{f \in H} \Delta(f), \tag{1}$$

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 two different loss functions: the *Mean Squared Error* (MSE) and the *Mean Absolute Percentage Error* (MAPE). These functions are defined as:

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

$$\Delta_{MAPE}(f) = \frac{1}{|O'|} \sum_{(k,i,speed) \in O'} \frac{|f(k, i) - y(k, i)|}{y(k, i)}, \tag{3}$$

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 [11]: Gradient Boosting Regression Trees (GBRT) [12, 13], Random Forests [2] and Linear Regression [10].

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 nicely the dynamics of individual sensors within the network. 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, 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 to find 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.

4. Dataset preparation

We evaluate the local, global, and cluster-based approaches introduced in Section 3 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 and contains five fields: i) *sensor ID*, representing the 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. We report that each record has a fixed size of 168 bytes and that each sensor produced on average 18 records per minute. We also report that the maximum number of records produced by a single sensor in a minute was observed to be 186.

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 1 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.

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

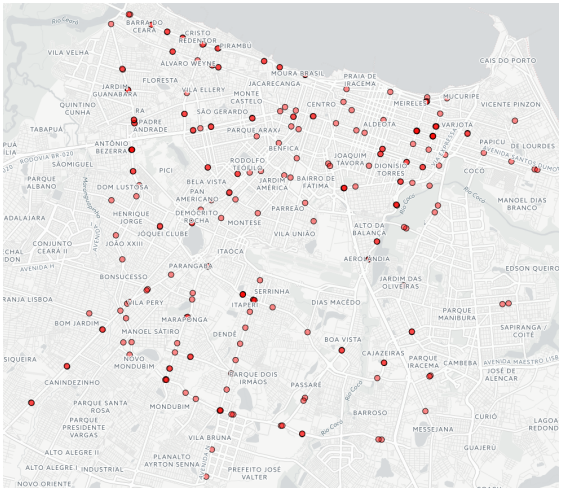

Figure 1: Map of the traffic sensors deployed during January and February 2014. Each circle in the map represents a group of spatially close sensors – darker circles indicate multiple sensors monitoring different lanes.

Month	# records	# sen. added	# sen. removed	# active sen.
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.

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 ten 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 [6, 5, 21, 34, 35, 23], 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 5.1. 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.

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

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 each sensor and 5-minute time-slot pair.

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 2 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 [28, 24, 31, 21], 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 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 in-

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

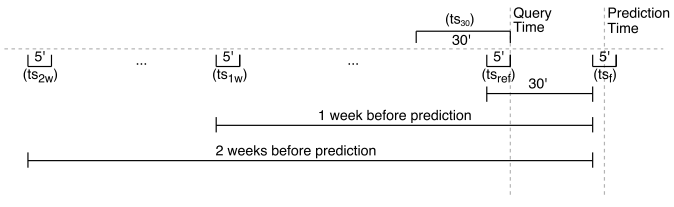

Figure 2: Diagram illustrating time-slot related features.

Group	Features	Name	Description
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 features used in our prediction models.

stant 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:

- *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.

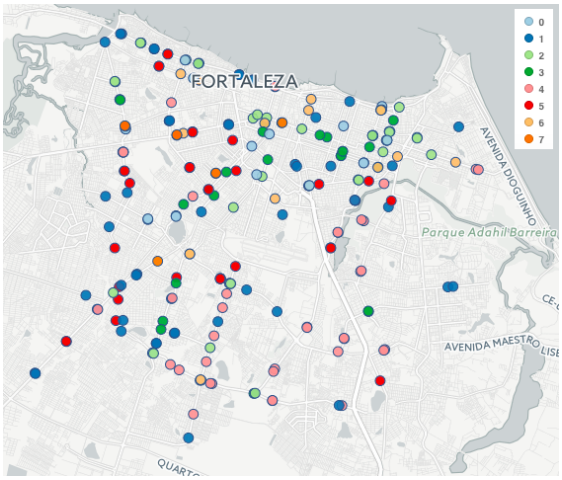

Figure 3: Example of sensor clustering conducted according to the similarity between time-series.

5. Experimental Evaluation

In this section we discuss the experiments conducted to generate different prediction models and assess their performance. More specifically, Section 5.1 introduces the experimental setting used to conduct the results evaluation, while Section 5.2 introduces the experimental questions and discusses the results.

5.1. 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 determined as the Euclidean distance between the associated multidimensional points. Figure 3 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++* [1] to

Partition	# agg. obs.	# sensors
Jan/Feb	2,266,620	240
Feb/Mar	2,038,634	189
Mar/Apr	2,040,453	184
Apr/May	2,162,768	236
May/Jun	2,152,463	228
Jun/Jul	2,053,947	182
Jul/Aug	2,420,851	244
Aug/Sep	2,322,379	249
Sep/Oct	2,388,983	258
Oct/Nov	2,722,819	266

Table 4: Details of the ten partitions defined on the reference dataset.

cluster sensors modeled as *time-series*. Consequently, in the experiments presented in Section 5.2 the cluster-based approach employs this strategy.

Dataset. The dataset used in the experimental evaluation, built as discussed in Section 4, contains 12,054,700 records. We split the data into several distinct partitions, i.e., training, validation, and test sets.

First, we define a *training set* as a temporal interval covering two consecutive months of data. By doing so, the dataset is thus arranged into ten partitions. We build the training sets so that two subsequent partitions of two months each overlap by one month, e.g., the *second* month of the *first* partition is also the *first* month of the *second* partition. Consequently, we have ten partitions ranging from January to November 2014: Jan/Feb, Feb/Mar, Mar/Apr, Apr/May, May/Jun, Jun/Jul, Jul/Ago, Ago/Sep, Sep/Oct, Oct/Nov. From Section 4 we remember that some of the features refer back to two weeks before prediction time. Moreover, we report that the cluster-based approach requires two weeks of data to cluster the sensors. Consequently, the first two weeks of any partition are used exclusively for feature and cluster computation. This means that we use the remaining six weeks of data to build the samples in the training set. Moreover, we filter out all the samples where the actual speed to predict is zero. Table 4 provides the most salient details about the training sets. For each training set of two months, we report the number of aggregated observations (**# agg. obs.**) and the number of operating sensors (**# sensors.**).

We then define *validation* and *test* sets as a temporal split covering a single month of data, from March to December. For each training set we use the month of data that follows to build the validation and test sets – for instance, when training on January and February 2014 we define the associated validation and test sets on March 2014.

To create validation and test sets we consider the sensors present *both* in the training set and in the month that follows it. By doing so, we avoid performing predictions that involve sensors not present in the training set. We also perform a *randomized stratified sampling*, based on the sensor identifiers, to homogeneously distribute the data of each sensor across the validation (50%) and test (50%) sets.

Machine learning methods. The machine learning methods considered in this work to train models are *Multivariable Linear Regression* (MLR) [10], *Random Forest* (RF) [2], and *Gradient Boosting Regression Trees* (GBRT) [12, 13]. The implementations of MLR and RF are provided by the Scikit-Learn machine learning library [22], while the implementation of GBRT is provided by XGBoost [7, 8]. 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 hyper-parameters needed by RF and GBRT, we determine the best combination by means of a grid search with different hyper-parameter subsampling ranging in {0.1, 0.5, 1.0}. The first hyper-parameter required by RF and GBRT is the *maximum tree depth* (max_depth) – to this end we consider the range $[5, 20]$ (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 implementations of RF and GBRT employ an early stopping technique to find out the best value. Finally, GBRT requires a third hyper-parameter, the learning rate ($learning\_rate$), for which we consider the range [0.05, 0.2] (with step 0.05). We also test the L1 and L2 regularization strategies, ranging respectively in the sets {0.0, 0.01, 1.0} and {0.0, 1.0, 100}. Overall, for each possible combination we generate a model and pick the one that yields the lowest MSE. The evaluation is conducted over the validation set.

5.2. 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? Also, do the global and cluster-based approaches address effectively the *cold start* problem?

5.2.1. 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 static sensor networks 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).

We generate the training, validation, and test sets according to the methodology described in Section 5.1. We also impose

an additional constraint that ensures we are dealing with static sensor networks; more specifically, we require that samples in the training, validation, and test sets refer to the same set of sensors. For instance, if a training set covers February and March, and the associated validation and test sets cover April, we only use samples collected by sensors that were working during all these months.

Tables 5 and 6 show the MSE and MAPE measured for the prediction models trained according the global, cluster and local approaches with the various ML algorithms tested. The first consideration we can do by looking at the figures reported in the tables is that GBRT outperforms the other techniques tested in all the experiments conducted, with MLR and RF following closely. Furthermore, we see that the local approach consistently provides prediction models achieving the lowest MSE and MAPE error, independently from the period considered. This result was largely expected since the local approach allows to train models that can capture the specific behavior and context of each sensor. On the other hand, we have to consider the huge cost for achieving this level of accuracy. For instance, let us consider the number of different models trained for performing the experiments reported in Table 5 and 6. For the local approach we had to train more than $2,000$ models for each algorithm. Conversely, with the global approach we had to train only 10 prediction models for each algorithm, one for each row of the tables.

However, in terms of MSE and MAPE achieved, the advantage of the local approach over the global one appears to be relatively low. Indeed, the performance of the global approach follows closely that of the local one, while the error difference between the two does not seem to justify the higher complexity of the local approach – an issue that becomes apparent when managing large sensor networks.

Finally, it is worth noting that although the prediction error achieved by models generated via the cluster-based approach is close to those obtained with models generated via the global and local ones, the cluster-based approach does not provide the level of accuracy we were expecting; we hypothesize that the similarity metrics we experimented may not be sufficient to capture the actual specificity of the data collected by sensors. Further work is thus needed to explore different similarity metrics that may allow the cluster-based approach to be a proper trade-off between the global and the local ones for what concerns prediction error and management complexity.

Relevance of features.

In the batch of experiments that follows we study the relevance of the features introduced in Section 4, 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 performing 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. For the local, cluster-based and global approaches we computed the feature importance as the average of all feature importance of all local models of all the ten months. Figure 4 presents the results.

From the plots we observe how the *slot_of_the_day* feature is the most relevant for the local approach and the second most relevant for the cluster-based approach intuitively indicating that the time of the day is a good indicator of the local traffic flow. On the other hand, the most relevant feature for the global and cluster-based approaches results the average of the last 30 minutes. Again, intuitively, this shows that predictions from the global and cluster-based approaches are based on the average of behaviour of sensors more than the specific time of the date that is more related to a single sensor. 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 MLR and RF following closely. The study also shows that the most relevant feature for the local approach is related to the time of the day thus reflecting the typical traffic flow behavior local to a given sensor while for global and cluster-based approaches the most relevant features are related to the average speed in the previous 30 days.

5.2.2. *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 static sensor networks, and analyze the performance of predictive models. We generate the training, validation, and test sets according to the methodology described in Section 5.1. We also impose an additional constraint that ensures we are dealing with static sensor networks; more specifically, we require that samples in the training, validation, and test sets refer to the same set of sensors. For instance, if a training set covers February and March, and the associated validation and test sets cover April, we will use samples collected only by sensors that were working during all these months.

The performance of the approaches is evaluated in terms of average MSE metric (Section 3). The ML technique used to generate the models is GBRT. Finally, the cluster-based approach uses k values comprised in the range $[2, 64]$ – indeed, this range represents an appropriate transition from the global approach to the local one.

Table 8 summarizes the average MSE yielded by the local, cluster-based and global approaches. In the first column we see the 10 different training sets while in the first row we see the test and validation months. As we anticipated in Section 5.1, for each pair of months we build a model that is then tested over the months that follow. This leads to the upper triangular matrix shown in Table 8. From the results we observe

training	val&test	test_size	ML algorithm	global	cl.k=2	cl.k=4	cl.k=8	local
Jan/Feb	**Mar**	375992	GBRT	14.62	14.65	14.61	14.60	**14.38**
			RF	15.30	15.05	15.00	14.96	14.62
			MLR	15.23	15.23	15.22	15.22	14.82
			HIST_AVG	92.80	92.93	92.59	92.51	19.83
Feb/Mar	**Apr**	644235	GBRT	15.54	15.59	15.62	15.51	**14.99**
			RF	16.35	16.12	16.08	15.97	15.17
			MLR	16.39	16.38	16.38	16.35	15.61
			HIST_AVG	92.58	92.56	91.97	91.18	20.91
Mar/Apr	**May**	332851	GBRT	14.18	14.30	14.23	14.13	**13.82**
			RF	15.09	14.88	14.82	14.70	14.15
			MLR	14.99	14.99	14.98	14.96	14.38
			HIST_AVG	90.95	89.47	88.33	87.06	19.39
Apr/May	**Jun**	633282	GBRT	14.14	14.22	14.17	14.08	**13.99**
			RF	14.77	14.59	14.55	14.41	14.21
			MLR	14.81	14.80	14.80	14.78	14.28
			HIST_AVG	93.35	92.46	91.51	89.27	21.71
May/Jun	**Jul**	379778	GBRT	14.00	14.05	14.23	14.12	**13.64**
			RF	14.82	14.57	14.57	14.50	14.11
			MLR	14.79	14.79	14.79	14.78	14.27
			HIST_AVG	87.52	87.52	87.48	87.57	19.43
Jun/Jul	**Aug**	703642	GBRT	13.99	14.06	14.02	14.08	**13.64**
			RF	14.88	14.58	14.52	14.50	13.88
			MLR	14.81	14.81	14.80	14.80	14.17
			HIST_AVG	90.62	90.85	89.10	88.28	18.75
Jul/Aug	**Sep**	332969	GBRT	14.47	14.54	14.52	14.48	**14.01**
			RF	15.36	15.04	14.98	14.89	14.11
			MLR	15.29	15.29	15.29	15.26	14.50
			HIST_AVG	89.88	88.97	87.42	84.90	19.84
Aug/Sep	**Oct**	832012	GBRT	14.42	14.54	14.56	14.49	**14.22**
			RF	15.36	15.08	15.05	14.95	14.94
			MLR	15.34	15.33	15.32	15.28	14.96
			HIST_AVG	97.22	97.25	97.11	96.30	21.30
Sep/Oct	**Nov**	493406	GBRT	13.77	13.83	13.85	13.76	**13.69**
			RF	14.68	14.37	14.30	14.21	13.86
			MLR	14.56	14.58	14.57	14.58	14.09
			HIST_AVG	95.86	95.91	96.07	96.41	20.56
Oct/Nov	**Dec**	176542	GBRT	14.78	14.84	14.81	14.75	**14.50**
			RF	15.64	15.35	15.28	15.17	14.61
			MLR	15.62	15.62	15.61	15.57	14.85
			HIST_AVG	94.89	94.64	92.53	92.59	20.24

Table 5: EQ1, MSE evaluation of ML techniques for the *global*, *cluster-based* (with $k \in [2, 8]$), and *local* approaches. Best performers are highlighted in **bold**.

training	val&test	test_size	ML algorithm	global	cl.k=2	cl.k=4	cl.k=8	local
Jan/Feb	**Mar**	375992	GBRT	0.0920	0.0922	0.0922	0.0921	**0.0906**
			RF	0.0943	0.0944	0.0944	0.0941	0.0923
			MLR	0.0938	0.0938	0.0938	0.0938	0.0932
			HIST_AVG	0.3006	0.3007	0.2996	0.2970	0.1063
Feb/Mar	**Apr**	644235	GBRT	0.0958	0.0959	0.0961	0.0957	**0.0934**
			RF	0.0985	0.0987	0.0986	0.0981	0.0948
			MLR	0.0982	0.0982	0.0982	0.0981	0.0964
			HIST_AVG	0.3022	0.3017	0.3000	0.2950	0.1107
Mar/Apr	**May**	332851	GBRT	0.0906	0.0911	0.0908	0.0905	**0.0892**
			RF	0.0936	0.0939	0.0937	0.0934	0.0914
			MLR	0.0928	0.0928	0.0928	0.0928	0.0924
			HIST_AVG	0.2962	0.2930	0.2899	0.2854	0.1064
Apr/May	**Jun**	633282	GBRT	0.0884	0.0887	0.0886	0.0883	**0.0868**
			RF	0.0904	0.0910	0.0909	0.0903	0.0887
			MLR	0.0906	0.0905	0.0905	0.0904	0.0890
			HIST_AVG	0.2915	0.2892	0.2869	0.2808	0.1085
May/Jun	**Jul**	379778	GBRT	0.0875	0.0876	0.0888	0.0884	**0.0861**
			RF	0.0902	0.0901	0.0905	0.0903	0.0881
			MLR	0.0901	0.0901	0.0902	0.0901	0.0892
			HIST_AVG	0.2816	0.2809	0.2805	0.2802	0.1016
Jun/Jul	**Aug**	703642	GBRT	0.0868	0.0871	0.0869	0.0878	**0.0854**
			RF	0.0898	0.0896	0.0894	0.0900	0.0870
			MLR	0.0896	0.0896	0.0896	0.0896	0.0885
			HIST_AVG	0.2869	0.2868	0.2831	0.2785	0.0998
Jul/Aug	**Sep**	332969	GBRT	0.0874	0.0876	0.0877	0.0874	**0.0854**
			RF	0.0905	0.0902	0.0900	0.0897	0.0864
			MLR	0.0903	0.0903	0.0903	0.0902	0.0878
			HIST_AVG	0.2812	0.2781	0.2749	0.2680	0.1009
Aug/Sep	**Oct**	832012	GBRT	0.0858	0.0862	0.0864	0.0861	**0.0848**
			RF	0.0888	0.0893	0.0893	0.0890	0.0879
			MLR	0.0889	0.0888	0.0888	0.0887	0.0885
			HIST_AVG	0.2918	0.2907	0.2899	0.2878	0.1030
Sep/Oct	**Nov**	493406	GBRT	0.0801	0.0805	0.0805	0.0803	**0.0793**
			RF	0.0829	0.0829	0.0826	0.0825	0.0803
			MLR	0.0824	0.0825	0.0825	0.0825	0.0815
			HIST_AVG	0.2765	0.2764	0.2760	0.2750	0.0960
Oct/Nov	**Dec**	176542	GBRT	0.0841	0.0845	0.0845	0.0842	**0.0824**
			RF	0.0871	0.0869	0.0867	0.0863	0.0832
			MLR	0.0866	0.0865	0.0865	0.0865	0.0847
			HIST_AVG	0.2791	0.2785	0.2732	0.2730	0.0973

Table 6: EQ1, MAPE evaluation of ML techniques for the *global*, *cluster-based* (with $k \in [2, 8]$), and *local* approaches. Best performers are highlighted in **bold**.

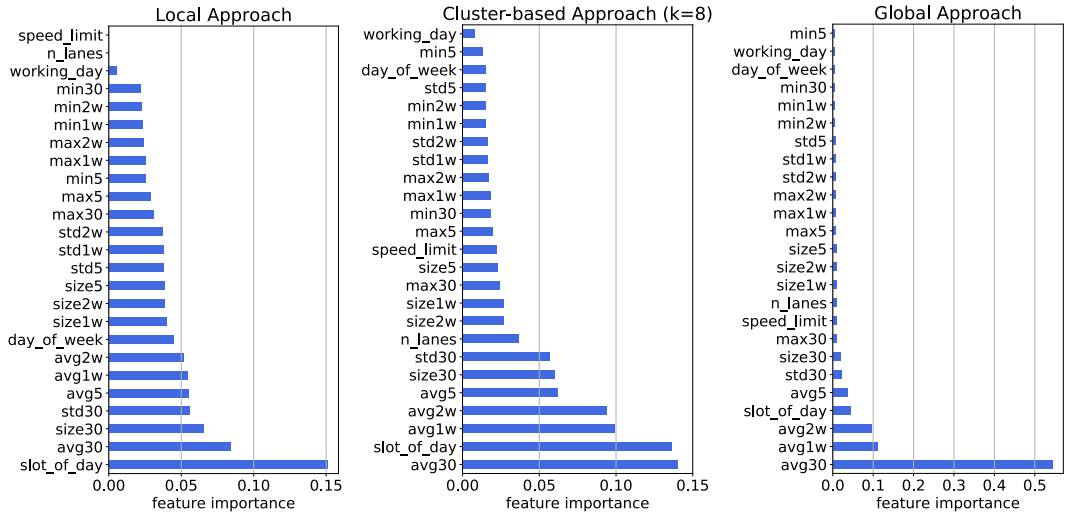

Figure 4: EQ1, analysis on the relevance of features. *Left-hand side*: local approach. *Center side*: cluster-based approach (k=8). *Right-hand side*: global approach.

how the local approach often achieves the best results in the first month of test, while its performance degrades noticeably in the months that follow. Conversely, the results highlight the robustness and resilience of the cluster-based (with $k = 2$) and global approaches, as their performance tend to remain stable across the months. In general, we argue that the local approach performs consistently worse due to its inability to *generalize* traffic behavior. On the other hand, 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.

Figure 5 provides a pictorial overview of the results. The plot clearly shows the trend of model aging: while models trained with the local approach tend to age significantly (this is signalled by a significant increase of the average MSE over time), thus demonstrating a high sensitivity to unseen data, global prediction models are definitely more robust to model aging, as they exhibit a stable average MSE.

Figure 5: EQ2, aging effect measured in terms of average MSE by increasing the distance of the test set from the training set. The average MSE is computed for the ten training partitions.

5.2.3. *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 sets.

To this end we conduct the following experiment: we compare the performance of the local, global, and cluster-based approaches limitedly to some of the sensors added to the network beyond the end of the pair of months used as training set, to prove the superior predictive performance of the latter approaches. We perform the experiment on eight months, from May to December 2014. We do not provide any result for March and April 2014, as no new sensors were added to the network in that period. Table 7 details the number of sensors added to the dynamic network for each considered month.

In this context we notice that the global and cluster-based

training set \ test set	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	3	7	12	27	28	48	55	57
Feb/Mar	52	7	12	76	28	96	102	103
Mar/Apr	52	7	12	76	28	96	102	103
Apr/May		5	10	24	26	45	52	54
May/Jun			5	19	21	40	48	50
Jun/Jul				64	16	84	91	92
Jul/Aug					5	21	29	31
Aug/Sep						16	24	26
Sep/Oct							9	11
Oct/Nov								2

Table 7: EQ3, number of sensors added to the dynamic sensor network. Each cell represents the number of sensors that were not present in the network during the temporal interval covered by the training set but appear in the temporal interval covered by the test set.

LOCAL

training set \ test set	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	14.38	15.89	14.74	16.00	15.25	16.08	16.32	17.09	16.01	17.79
Feb/Mar		14.99	14.39	15.38	14.61	15.47	15.86	16.73	15.45	17.09
Mar/Apr			13.82	14.78	14.75	15.64	15.81	16.49	15.55	16.97
Apr/May				13.99	14.25	15.03	15.29	16.63	15.06	16.70
May/Jun					13.64	14.26	14.61	15.99	14.46	16.26
Jun/Jul						13.64	14.01	15.35	13.88	15.92
Jul/Aug							14.01	15.56	14.49	16.43
Aug/Sep								14.22	13.98	15.39
Sep/Oct									13.69	15.32
Oct/Nov										14.50

CLUSTER-BASED (k = 2)

training set \ test set	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	14.65	15.63	14.25	14.20	13.99	14.25	14.42	14.77	13.69	15.18
Feb/Mar		15.59	14.31	14.24	14.07	14.43	14.55	15.04	13.79	15.05
Mar/Apr			14.30	14.15	14.18	14.38	14.47	15.02	13.88	15.08
Apr/May				14.22	14.06	14.28	14.40	14.79	13.76	15.00
May/Jun					14.05	14.33	14.60	14.74	13.81	15.10
Jun/Jul						14.06	14.21	14.73	13.55	14.83
Jul/Aug							14.54	15.13	14.07	15.16
Aug/Sep								14.54	13.71	14.47
Sep/Oct									13.83	14.84
Oct/Nov										14.84

GLOBAL

training set \ test set	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	14.62	15.76	13.93	14.41	14.05	13.79	14.31	14.47	13.50	14.17
Feb/Mar		15.54	14.26	14.25	14.05	14.25	14.33	14.77	13.77	15.12
Mar/Apr			14.18	14.17	14.05	14.26	14.37	14.76	13.78	15.06
Apr/May				14.14	13.97	13.69	14.29	14.33	13.39	14.03
May/Jun					14.00	13.68	14.33	14.41	13.42	13.98
Jun/Jul						13.99	14.13	14.41	13.49	14.74
Jul/Aug							14.47	14.49	13.74	14.47
Aug/Sep								14.42	13.64	14.41
Sep/Oct									13.77	14.67
Oct/Nov										14.78

Table 8: EQ2, average MSE over different months for models generated via the *local* (top table), *cluster-based* (k=2, middle table), and *global* (bottom table) approaches.

models used in the first batch of experiments (i.e., EQ1) can be reused. Due to its characteristics, however, the local approach requires a different training procedure since it must train a model each time a new sensor is added to the network. Accordingly, each local model is trained and evaluated over the month of data where the new sensor appears. Note that, in EQ1 and EQ2, validation and test sets were built over each month, splitting the samples of that month in half between the two sets. However, in this experiment training data is required for each added sensor: to this end, we use the validation set previously defined, i.e., 50% of the data of the month, to train the local model. We then split the test set in two halves (25% of the original set each) and use the first half to validate the model and the other half to test it. We assess the performance of the approaches by considering the average MSE achieved over all the sensors considered. Table 9 presents the results of the investigation while Figure 6 presents the average prediction error achieved by the three strategies for each specific test month.

The results show that the local approach is able to handle new sensors even if the global and cluster-based (with $k = 2$) approaches behave similarly to the local approach in terms of average MSE. The experiment thus shows that the global and cluster-based approaches are able to effectively tackle changes in dynamic sensor networks. The competitiveness of the global and cluster-based approaches becomes even more evident when dealing with older models, i.e., models built months before prediction time. Here, the local approach performs worse as it does not deal properly with aging (see results for EQ2). This result has interesting practical implications: first, our proposed global and cluster-based approaches allow for a reduced management complexity, meaning that they do not require per-sensor training; instead, they allow for an easy development and deployment of prediction models. Furthermore, the results demonstrate that the global and cluster-based approaches tackle effectively the *cold start* problem, since they achieve comparable

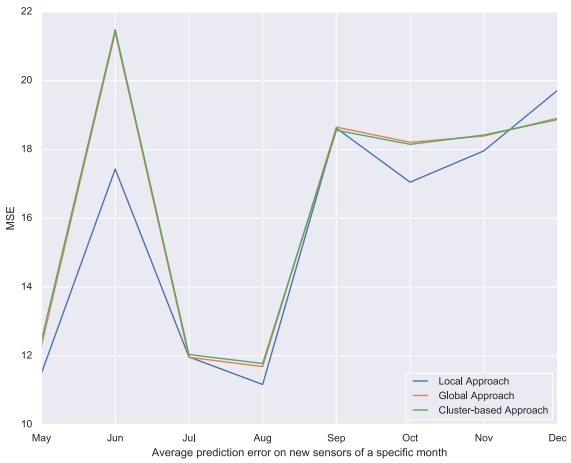

Figure 6: EQ3, average prediction error on new sensors of a specific month. The result is obtained by averaging the performance of all the prediction models available for a specific test month.

results w.r.t. the local approach without the need to retrain a model each time a new sensor is added to the network. Finally, as the global and cluster-based approaches are more robust to aging they do not require frequent re-training on specific (i.e., per-sensor) data.

6. Conclusion and future work

Traffic forecasts should be accurate and robust to changes in traffic monitoring networks. When such changes may occur, traffic management systems should optimize management and advisory strategies to enhance decision-making capabilities and maintain an appropriate level of service. In this context we consider the problem of predicting the speed of vehicles by analyzing data collected from a large and dynamic network of sensors, where sensing devices are continuously added and removed to the network. 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.

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 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 paper we propose the global and cluster-based approaches. The global approach trains a single prediction model over the observations of all the sensors of the network, while the cluster-based approach trains prediction models on clusters of sensors to capture traffic behavior observed by "similar" sensors. 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 retrainings, with the cluster-based approach following closely. Finally and most importantly, the evaluation shows that the global

LOCAL

training set \ test set	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	12.05	16.98	12.02	11.95	17.32	16.8	18.39	20.83
Feb/Mar	11.23	16.98	12.02	10.28	17.32	14.49	14.38	15.51
Mar/Apr	11.23	16.98	12.02	10.28	17.32	14.49	14.38	15.51
Apr/May		18.79	11.52	12.54	17.29	17.09	19.28	21.44
May/Jun			12.26	12.17	18.22	18.16	19.68	22.39
Jun/Jul				9.82	21.97	15.48	15.45	16.39
Jul/Aug					20.91	19.36	19.68	21.07
Aug/Sep						20.57	21.6	22.76
Sep/Oct							18.8	20.72
Oct/Nov								20.54

CLUSTER-BASED (k = 2)

training set \ test set	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	11.89	20.81	12.41	11.98	17.23	17.62	18.46	19.96
Feb/Mar	12.67	21.47	12.49	11.52	17.27	15.73	15.58	15.55
Mar/Apr	12.79	20.13	12.3	11.62	17.26	15.79	15.64	15.70
Apr/May		23.51	11.6	12.29	17.31	18.04	19.29	20.76
May/Jun			11.41	12.32	18.26	19.09	20.06	21.22
Jun/Jul				10.96	22.1	16.82	16.38	16.25
Jul/Aug					20.5	20.5	19.73	19.24
Aug/Sep						21.64	21.3	20.97
Sep/Oct							19.36	19.44
Oct/Nov								19.59

GLOBAL

training set \ test set	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Jan/Feb	11.82	20.86	12.24	11.88	17.23	17.6	18.41	19.99
Feb/Mar	12.52	20.03	12.3	11.34	17.4	15.64	15.49	15.39
Mar/Apr	12.56	21.47	12.28	11.36	17.47	15.76	15.57	15.58
Apr/May		23.26	11.58	12.28	17.33	18.07	19.4	20.78
May/Jun			11.4	12.18	18.51	19.33	20.12	21.65
Jun/Jul				11.09	22.14	16.91	16.39	16.29
Jul/Aug					20.46	20.64	19.78	19.47
Aug/Sep						21.76	21.08	20.89
Sep/Oct							19.26	19.45
Oct/Nov								19.63

Table 9: EQ3, MSE yielded by the *local* (top table), *cluster-based* (with k = 2, middle table), and *global* (bottom table) approaches with respect to a dynamic sensor network.

and cluster-based approaches achieve comparable results w.r.t. 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 performance of the global and cluster-based approaches is determined by its resilience to model aging and to structural changes affecting dynamic sensor networks, and we believe that these results have the potential to impact smart cities and current data management practices in the field of speed prediction.

One line of future research is towards the improvement of the clustering approach, as a compromise between local and global. Alternative similarity measures have to be tested to better group sensors data with similar behaviour.

Another line 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*, i.e., road networks where the costs of edges can vary over time.

Acknowledgments

This work is partially supported by the FUNCAP SPU 8789771/2017, the UFC-FASTEF 31/2019, BIGDATAGRAPES (EU H2020 RIA, grant agreement Nº780751), MASTER (H2020, MSCA grant agreement 777695) and the OK-INSAID (MIUR-PON 2018, grant agreement NºARS01_00917) projects.

[1] Arthur, D., Vassilvitskii, S., 2007. k-means++: The advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, pp. 1027–1035.

[2] Breiman, L., 2001. Random forests. Machine Learning 45 (1), 5–32. URL http://dx.doi.org/10.1023/A:1010933404324

[3] Castro-Neto, M., Jeong, Y.-S., Jeong, M.-K., Han, L. D., 2009. Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions. Expert systems with applications 36 (3), 6164–6173.

[4] Chan, K. Y., Dillon, T. S., Singh, J., Chang, E., 2012. 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 13 (2), 644–654.

[5] Chen, C., 2003. Freeway performance measurement system (pems). California Partners for Advanced Transit and Highways (PATH).

[6] Chen, C., Skabardonis, A., Varaiya, P., 2003. A system for displaying travel times on changeable message signs. University of California, Berkeley 94720, 1720.

[7] Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, pp. 785–794.

[8] Chen, T., He, T., 2015. Xgboost: extreme gradient boosting. R package version 0.4-2.

[9] Clark, S., 2003. Traffic prediction using multivariate nonparametric regression. Journal of transportation engineering 129 (2), 161–168.

[10] Freedman, D. A., 2009. Statistical models: theory and practice. cambridge university press.

[11] Friedman, J., Hastie, T., Tibshirani, R., 2001. The elements of statistical learning. Vol. 1. Springer series in statistics New York.

[12] Friedman, J. H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189–1232.

[13] Friedman, J. H., 2002. Stochastic gradient boosting. Computational Statistics & Data Analysis 38 (4), 367–378.

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

[15] Hoogendoorn, S. P., Bovy, P. H., 2001. State-of-the-art of vehicular traffic flow modelling. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 215 (4), 283–303.

[16] Huang, W., Song, G., Hong, H., Xie, K., 2014. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. IEEE Transactions on Intelligent Transportation Systems 15 (5), 2191–2201.

[17] Klein, L. A., Mills, M. K., Gibson, D. R., 2006. Traffic detector handbook: Volume ii. Tech. rep., United States. Federal Highway Administration.

[18] Li, Y., Yu, R., Shahabi, C., Liu, Y., Jul. 2017. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. ArXiv e-prints.

[19] Lippi, M., Bertini, M., Frasconi, P., 2013. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. IEEE Transactions on Intelligent Transportation Systems 14 (2), 871–882.

[20] Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.-Y., 2015. Traffic flow prediction with big data: a deep learning approach. IEEE Transactions on Intelligent Transportation Systems 16 (2), 865–873.

[21] Min, W., Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transportation Research Part C: Emerging Technologies 19 (4), 606–616.

[22] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al., 2011. Scikit-learn: Machine learning in python. Journal of Machine Learning Research 12 (Oct), 2825–2830.

[23] Rzeszótko, J., Nguyen, S. H., Aug. 2012. Machine learning for traffic prediction. Fundam. Inf. 119 (3-4), 407–420. URL http://dl.acm.org/citation.cfm?id=2385625.2385635

[24] Schmitt, E. J., Jula, H., 2007. On the limitations of linear models in predicting travel times. In: 2007 IEEE Intelligent Transportation Systems Conference. IEEE, pp. 830–835.

[25] Shuai, M., Xie, K., Pu, W., Song, G., Ma, X., 2008. An online approach based on locally weighted learning for short-term traffic flow prediction. In: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems. ACM, p. 45.

[26] Sun, S., Zhang, C., Yu, G., 2006. A bayesian network approach to traffic flow forecasting. IEEE Transactions on Intelligent Transportation Systems 7 (1), 124–132.

[27] Van Der Voort, M., Dougherty, M., Watson, S., 1996. Combining kohonen maps with arima time series models to forecast traffic flow. Transportation Research Part C: Emerging Technologies 4 (5), 307–318.

[28] Vlahogianni, E. I., Golias, J. C., Karlaftis, M. G., 2004. Short-term traffic forecasting: Overview of objectives and methods. Transport reviews 24 (5), 533–557.

[29] Vlahogianni, E. I., Karlaftis, M. G., Golias, J. C., 2014. Short-term traffic forecasting: Where we are and where we're going. Transportation Research Part C: Emerging Technologies 43, 3–19.

[30] Wang, D., Zhang, Q., Wu, S., Li, X., Wang, R., 2016. Traffic flow forecast with urban transport network. In: Intelligent Transportation Engineering (ICITE), IEEE International Conference on. IEEE, pp. 139–143.

[31] Wojnarski, M., Gora, P., Szczuka, M., Nguyen, H. S., Swietlicka, J., Zeinalipour, D., Dec 2010. Ieee icdm 2010 contest: Tomtom traffic prediction for intelligent gps navigation. In: 2010 IEEE International Conference on Data Mining Workshops. pp. 1372–1376.

[32] Yu, D., Liu, Y., Yu, X., 2016. A data grouping cnn algorithm for short-term traffic flow forecasting. In: Asia-Pacific Web Conference. Springer, pp. 92–103.

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

[34] Zhang, Y., Haghani, A., 2015. A gradient boosting method to improve travel time prediction. Transportation Research Part C: Emerging Technologies 58, Part B, 308 – 324, big Data in Transportation and Traffic

Engineering.
[35] Zhang, Y., Zhang, Y., 2016. A comparative study of three multivariate short-term freeway traffic flow forecasting methods with missing data. Journal of Intelligent Transportation Systems 20 (3), 205–218.