Comparison of traffic forecasting methods in urban and suburban context

Julien Salotti, Serge Fenet, Romain Billot, Nour-Eddin El Faouzi, Christine Solnon

Comparison of traffic forecasting methods in urban and suburban context

Julien Salotti*, Serge Fenet*, Romain Billot†, Nour-Eddin El Faouzi‡, Christine Solnon*
*Univ Lyon, INSA de Lyon/Univ. Lyon 1, LIRIS, F-69622, Lyon, France
firstName.lastName@liris.cnrs.fr
†IMT Atlantique, Lab-STICC, F-29238, Brest, France
romain.billot@imt-atlantique.fr
‡Univ Lyon, ENTPE, IFSTTAR, LICIT UMR-9401, F-69675, Lyon, France
nour-eddin.elfaouzi@ifsttar.fr

***Abstract*—In the context of Connected and Smart Cities, the need to predict short term traffic conditions has led to the development of a large variety of forecasting algorithms. In spite of various research efforts, there is however still no clear view of the requirements involved in network-wide traffic forecasting.**

In this paper, we study the ability of several state-of-the-art methods to forecast the traffic flow at each road segment. Some of the multivariate methods use the information of all sensors to predict traffic at a specific location, whereas some others rely on the selection of a suitable subset. In addition to classical methods, we also study the advantage of learning this subset by using a new variable selection algorithm based on time series graphical models and information theory. This method has already been successfully used in natural science applications with similar goals, but not in the traffic community.

A contribution is to evaluate all these methods on two real-world datasets with different characteristics and to compare the forecasting ability of each method in both contexts. The first dataset describes the traffic flow in the city center of Lyon (France), which exhibits complex patterns due to the network structure and urban traffic dynamics. The second dataset describes inter-urban freeway traffic on the outskirts of the french city of Marseille.

Experimental results validate the need for variable selection mechanisms and illustrate the complementarity of forecasting algorithms depending on the type of road and the forecasting horizon.

***Index Terms*—Traffic Forecasting, Time Series, Variable Selection, k-NN, SVR, Lasso, VAR, ARIMA.**

I. Introduction and Context

The growing need for short-term forecasting of traffic conditions has led to the development of a large variety of forecasting algorithms. Real-time Intelligent Transportation Systems and Connected and Smart Cities are the driving force behind this renewal of interest. To measure traffic conditions, sensors are embedded in roads and measure traffic flow (*i.e.*, number of vehicles per time unit) and density (*i.e.*, number of vehicles per length unit). Collected data are aggregated with respect to fixed-length time steps to obtain time series. To forecast traffic conditions, the goal is to estimate these measures at time $t+h$ based on historical data, *i.e.*, time-series up to time t. In our case, we focus on short-term forecasting of traffic flow measures, with a prediction horizon $h \leq 30$ minutes.

Road traffic exhibits several features that are common among many complex systems: self-organization, emergence of transient space-time patterns based on local and global feedback loops, hardness of predictability, etc. Unlike some other complex systems, however, road networks are characterized by a huge variability at small time scales [1] that prevents efficient forecasting at small to medium time scales [2]. If traffic forecasting is difficult for freeway traffic (which has attracted the majority of current scientific work), it is even more difficult for city center traffic. In this latter context, the sensitivity to highly probable small non predictable events leads to an even higher variability [3]. Moreover, city centers are often equipped with a high number of sensors and, in this case, forecasting is subject to the curse of dimensionality that dilutes important information into a small dynamic subset of sensors. So, in this particular context, data availability creates new challenges. While in theory only a small subset of sensors is necessary to learn a reliable predictor, their identification is an essential step of the process. As pointed out in [4], [5], an analysis of space-time dependencies is necessary to identify critical sensors and reduce the dimensionality of the problem.

Our application is also challenging for forecasting algorithms which need a lot of training data to perform well. This is due to the fact that even if a large history of data is available (*e.g.*, several years), it may be meaningless to use the whole dataset to train the model because the road network is frequently modified: speed limits, one-way street directions, road signaling, etc, are often changed. Therefore, models are often trained on a recent data history (a few weeks) in order to keep the underlying phenomenon as stable in time as possible.

The literature in short-term traffic forecasting is extensive [6] and involves non-parametric models [7], [8], Kalman filtering, auto regressive integrated moving average models, wavelets [9], fuzzy logic [10], traditional or deep neural networks [11], [12], or Bayesian models [13]. Nevertheless, few contributions analyze whether the non linear characteristics of urban traffic allow for a relevant application of these methods to multivariate traffic flow time series. [14] investigated the

This work is supported by the LABEX IMU (ANR-10-LABX-0088) of Université de Lyon, within the program "Investissements dAvenir" (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR)

deterministic properties of freeway traffic flow using a non linear time series analysis technique and showed that these data exhibit strong chaotic properties. [15] used a dynamic Bayesian network to identify conditional independences and infer some causal drives embedded in traffic flow multivariate series.

More recently, Runge et al. [16]–[18] introduced a method that takes into account the specific characteristics of high-dimensional dynamical systems (time-delays, non-linearity, and strong autocorrelations) in the context of climate and geophysical data. The idea is to extract important regions supposed to participate in the propagation of perturbations through the system. This is done with a two-step method. First, a graphical model based on decomposed transfer entropy is built to detect causal interaction between variables of the system, as well as assert time delays between causes and effects. This leads to a causal time series graph structure that describes conditional dependencies between components at different time lags. In a second step, the strength of each interaction in this graph is evaluated with partial correlation and a regression measure. This reduces detection biases when strong autocorrelations are present, and allows the ranking of pairwise associations between sensors based on their causal strength. Applied to climate data, the method improves forecasting performance.

A question addressed in this paper is the applicability of this causality detection approach to improve urban traffic forecasting. The main challenges are coming from specificities of urban traffic. First, the number of sensors is much higher for urban traffic than for previous applications of this approach, and the construction of the graphical model may become computationally challenging. Also, urban traffic systems are more sensitive to perturbations than climate systems. For example, cascading failures (leading to traffic jams) spread through the system in minutes, rather than in months or years in climate data. Furthermore, in our case, we have a sampling frequency of 6 minutes for the Lyon dataset, which is much slower than some fixed forced regimes present in the system (*e.g.*, less than a minute for some traffic lights). Finally, causal dependencies change through time: in fluid traffic a sensor causally influences downstream sensors, whereas in a traffic jam (that may coalesce in a few minutes) causality is reversed as the jam wave-front travels backwards.

Contributions and organization of the paper

In this paper, we evaluate several state of the art methods for forecasting traffic conditions in two different urban datasets, and we investigate whether causal information can readily improve urban traffic forecasting. Throughout the paper, when referring to causality, we consider a definition similar to the idea of Granger Causality [19] for more than two variables: we call causality a dependency between two time series, where the past of the first series provides a unique information which is useful for predicting the future of the second series. The term "unique" means that this information is not present in the rest of the dataset. However, it should be emphasized that the presence of such a relationship in the data does not prove the existence of a causal relation in the real world.

Section II describes a set of ten forecasting methods. We consider three kinds of forecasting methods, *i.e.*, linear approaches, support vector regression, and nearest neighbor approaches. For each kind of approach, we consider univariate and multivariate settings (to evaluate the interest of exploiting information coming from other sensors), and we consider variable selection settings (to evaluate the interest of exploiting causality information).

In Section III, we experimentally compare these ten methods on two real-world datasets with different characteristics: the first one comes from sensors in the city center of Lyon whereas the second one comes from sensors on inter-urban freeways in the city of Marseille. Measures have been aggregated with a time step of 6 minutes for the Lyon dataset and 2 minutes for the Marseille dataset.

When forecasting traffic flow in Lyon city center, we show that the multivariate nearest neighbor approach outperforms other methods, and that causality information does not significantly improve the multivariate nearest neighbor approach.

When forecasting on Marseille urban freeway, we show that classic parametric methods (Lasso, SVR) give the best forecasts. The embedded variable selection mechanism of Lasso allows for a robust performance even when the forecasting horizon increases.

II. Description of forecasting methods

Sensors are denoted $x_1, \ldots, x_N$, the variable that gives the state of x_i at time step t is denoted $X_i(t)$, and its realization is denoted $x_i(t)$. Given a current time step t, a time horizon h, and a history length d, the goal is to produce at time t a forecast $\hat{x}_i(t+h)$ of $x_i(t+h)$ given the realizations of all variables between time steps $t-d+1$ and t.

In order to ascertain the predictive power of a wide range of forecasting schemes, we describe ten methods in this section. We consider three kinds of approaches: state-of-the-art linear and time series approaches (described in Section II-A), support vector regression approaches (described in Section II-B), and nearest neighbor approaches (described in Section II-C). For each kind of approach, we consider a univariate method (that only uses the past of x_i to compute $\hat{x}_i(t+h)$), and a multivariate method (that uses the past of all sensors to compute $\hat{x}_i(t+h)$). For each kind of approach, we also consider methods that select a spatio-temporal environment, for each sensor, and that use this selected environment to compute $\hat{x}_i(t+h)$. We consider two different approaches for selecting a spatio-temporal environment: a Lasso-based approach (described in Section II-D), and a Tigramite-based approach (described in Section II-E).

A. Linear and Time Series Methods

• ARIMA [6] is a classical univariate time series model. It has been widely used in traffic flow forecasting literature. The "AR" term is the autoregressive part, meaning that the future of the time series is regressed on its prior values. The

"I" for *integrated* indicates that we can predict the difference between consecutive values rather than the values themselves. The "MA" part (moving average) means that the current error value is a linear combination of past errors.

The ARIMA model has 3 hyperparameters: p the autoregressive order, q the moving-average order and d the degree of differencing. Let us define the difference operator Δ for a time series:

$$\Delta X(t) = X(t) - X(t-1)$$

We note $\Delta^d x(t)$ the difference of degree d of the time series. It is obtained by applying d times the difference operator to the series. The goal of this operation is to obtain a stationary time series. The ARIMA (p, d, q) model is defined by the following equation:

$$\Delta^d X(t) = \phi_0 + \sum_{i=1}^{p} \phi_i \, \Delta^d X(t-i) + \sum_{i=1}^{q} \theta_i \, \epsilon_{t-i} + \epsilon_t \quad (1)$$

where ϵ_t is white noise. A unique model is learned to forecast every time horizon h: when $h = 1$, we directly produce a forecast from past observations; when $h > 1$, we compute a forecast for $h = 1$ which is then used as a new observation to recursively produce forecasts for higher horizons. The setting of hyperparameters p, d, q is determined using Akaike information criterion (AIC) [20].

• VAR (vector autogression) [21] is a linear multiple regression on the past values of all sensors: a value observed on a specific sensor may be influenced by those observed in the past of all other sensors. The model has one hyperparameter (the autoregressive order p) which is determined using AIC. As for ARIMA, a unique model is used to produce forecasts recursively for all horizons starting with $h = 1$.

B. Support Vector Regression

SVR (Support Vector Regression) [22] is an adaptation of SVM (Support Vector Machine) for regression problems. They are kernel methods that project each example x into a new representation space $\phi(x)$, in which it will be possible to learn a linear model with good accuracy. In our experiments, we tried the linear kernel and the Radial basis function (RBF) kernel. We observed similar performances for both, and for readability we only present the results with the RBF kernel. The model has 3 parameters: γ is the parameter of the kernel, C and ϵ are two parameters of SVR controlling the complexity of the regression model and its number of support vectors. The parameters are learned by grid search, with logarithmic ranges centered around initialization values described in [23].

• SVR-RBF-*uni* denotes a univariate SVR model with an RBF kernel. A model is learned for each sensor and forecasting horizon. As for the ARIMA model, its inputs are the d last observations of this particular sensor.

• SVR-RBF-*multi* denotes a multivariate SVR model with an RBF kernel. A model is learned for each sensor and forecasting horizon. As for the VAR model, its inputs are the d last observations of all the sensors.

C. Nearest Neighbor Approaches

Nearest neighbor (K-NN) forecasting [7] is based on the idea that the future behaves like the past. It uses a description of the current state of the system, where a system is either a single sensor in a univariate set-up, or a set of sensors in a multivariate set-up. The idea is to search similar past states of the system in the history of observations, then combine the future of these states (which has been observed) in order to predict the future of the current state (which has not yet happened).

• K-NN-*uni* denotes the univariate version of K-NN. The state of sensor x_i at time step t is described by its d last values. To measure the similarity of the state of sensor x_i at two different time steps t_1 and t_2, we compute the Euclidean distance between its states at times t_1 and t_2. The predicted value at time step $t + h$ is the average of the values observed at time horizon h in the future of the k-nearest states of x_i in its past values.

• K-NN-*multi* denotes an extension of K-NN-*uni* to the multivariate setting. In this case, $\mathbf{x}(t) = (x_1(t), \ldots, x_N(t))$ is the value of the multivariate time series. The state vector $\mathbf{s}(t)$ is the matrix composed of the N state vectors associated with each sensor:

$$\mathbf{s}(t) = \big(s_1(t), \ldots, s_N(t)\big)$$

As for the univariate case, let h be the time horizon, and $t_1, \ldots, t_k \in [0, t-h]$ be the k time steps such that $\mathbf{s}(t_1), \ldots, \mathbf{s}(t_k)$ are the k nearest states of $\mathbf{s}(t)$. The predicted vector at time step $t + h$ is

$$\hat{\mathbf{x}}(t+h) = (\hat{x}_1(t+h), \ldots, \hat{x}_N(t+h))$$

with

$$\hat{x}_i(t+h) = \frac{1}{k} \sum_{j=1}^{k} x_i(t_j + h). \quad (2)$$

For all K-NN approaches, the value of the hyperparameter k is determined with cross-validation. K-NN-*uni* and K-NN-*multi* have another parameter d (the length of a sequence) which is also learned with cross-validation.

D. Selection with Lasso

In our multivariate methods (VAR, SVR-RBF-*multi*, and K-NN-*multi*), the same environment (composed of all sensors) is used to predict the value of each sensor: these methods assume that a sensor in the network may be impacted by every other sensor (as well as by its own past).

A goal of this paper is to study the interest of selecting more relevant environments. In this case, we assume that the state of a sensor at time $t+h$ is not impacted by the states of every other sensor at all time steps in $[t-d, t]$, but only by a subset of these variables. Selecting the most relevant subset of variables for each sensor is challenging since there is an exponential number of possible subsets: it is not possible to

evaluate each of these subsets in order to select the one that leads to the best prediction.

Lasso [4] is a linear multiple regression on past values of all sensors with an ℓ_1 regularization term. Its coefficients shrink to zero for variables that are not useful for prediction. This method therefore allows predictions with an embedded variable selection mechanism. To predict the future of a sensor y at horizon h, a regression is done on the past d values of all sensors:

$$\hat{Y}(t+h) = \sum_{i=1}^{N} \sum_{j=0}^{d-1} \beta_{i,j} X_i(t-j) \tag{3}$$

Let β be the flatten parameter vector. Its value is optimized by the following minimization:

$$\min_{\beta} \frac{1}{2(T-h)} \sum_{t=1}^{T-h} \left(\hat{y}(t+h) - y(t+h)\right)^2 + \lambda \|\beta\|_1$$

where T is the number of samples in the training set. In this approach, a different model is learned for every sensor x_i and horizon h. The value of the regularization parameter λ is learned by cross-validation.

Because this selection procedure is already embedded in a regression model, we can already produce forecasts with this model.

• Lasso denotes the prediction produced with this regression model.

We can also use this selection procedure with other forecasting methods: Lasso is used to select a subset of relevant variables (those with nonzero coefficients after the Lasso regression), and this subset of relevant variables is used to build a model with another forecatsing method.

• K-NN-*lasso* denotes a K-NN multivariate prediction scheme with such a lasso-based variable selection procedure. For each sensor i and each time horizon h, Lasso is used to select a subset of relevant variables (with nonzero coefficients), that we denote $P(i,h) \subseteq \{X_j(t-\tau) : j \in [1,N], \tau \in [0,d]\}$. Given this subset, the state $s_i^h(t)$ of the system at time t used to predict $x_i(t+h)$ is the vector of the realizations of the variables in $P(i,h)$. We can then search for the k-nearest states as for K-NN-*multi* and obtain the prediction $\hat{x}_i(t+h)$ using equation (2).

• SVR-RBF-*Lasso* denotes a multivariate SVR model with an RBF kernel. A model is learned for each sensor and forecasting horizon. The difference with SVR-RBF-*multi* is the Lasso processing step. As for the K-NN-*lasso* model, its inputs are the variables with nonzero coefficients $P(i,h)$.

E. Selection with Tigramite

TiGraMITe [24] is a time series analysis method for causal discovery. This algorithm can quantify information transfer along causal pathways in complex systems: it restricts its results to optimal causal drivers and excludes the effects of common drivers and indirect influences. Hence, it seems well suited for urban traffic forecasting. More precisely, given a training dataset, TiGraMITe builds a causal dependence graphical model linking sensor states at different time delays: An edge $X_i(t-\tau) \rightarrow X_j(t)$ models the fact that the state of x_i has a causal influence on the state of x_j with a time delay τ[1]. A threshold parameter controls the density of the learned graph: depending on its value, we obtain graphs with different densities. We note $\mathcal{G} = \{g_1, \ldots, g_m\}$ the set of these graphs with m the number of graphs. For a given graph g_j, $P(i,h)$ is derived from the set of causal parents of $X_i(t+h)$ in g_j as illustrated in Fig. 1. A key point is to choose the graph $g_j \in \mathcal{G}$ used to define causal parents. The best graph is chosen by performing the prediction for each graph of $\mathcal{G}$ on a validation set, and keeping the one leading to the best prediction on this set. As for all the evaluated methods, it is evaluated on separated training and validation sets (see III-D).

• K-NN-*tig*SB denotes a K-NN multivariate prediction scheme with a variable selection procedure where $P(i,h)$ is computed with the causal algorithm (see Fig. 1). This set of causal parents determines our description of the state of the system (used to select the nearest neighbors), and it depends on the sensor x_i for which we want to perform the prediction and on the time horizon h of this prediction. The SB part of the name K-NN-*tig*SB means *Single Best* because we select only the best causal graph.

We did not use TiGraMITe for variable selection of the linear regression model, because Lasso is the classical and well-established approach for this type of model. We also did not use TiGraMITe for the SVR approach, because the time and computing resources needed to learn an SVR model for each possible graph g_j is not really suited to our application, where models are retrained frequently. Furthermore, we performed the experiments with K-NN first, showing that TiGraMITe was not able to learn a pertinent subset of variables. We describe this in section III.

III. Experimental Comparison

We compare the forecasting methods introduced in the previous section on two real datasets coming from two French conurbations: Lyon and Marseille.

A. Missing Values

Loop sensors are not very reliable, leading to datasets with a lot of missing data, both for Lyon and Marseille. When there were less than 3 successive missing values, we applied imputation with a simple linear interpolation. For larger gaps of missing values, we could have used the historical median or mean value, but we would have then compared our forecasting methods on data contaminated by our imputation method instead of raw data. It would therefore have been difficult to quantify the effect of such a contamination on the relative

[1]In our case, we may also observe a contemporary (non directed) dependency $X_i(t) \leftrightarrow X_j(t)$ due to the fact that our raw data are the results of the aggregation of a measure per minute over multiple time steps.

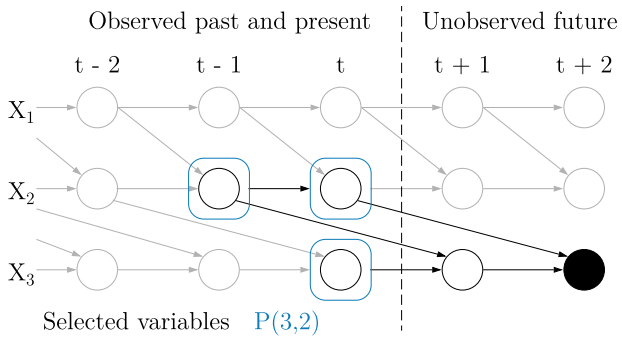


Fig. 1: Selection of relevant variables from a `TiGraMITe` graph. The predicted variable is $X_3(t+2)$ (black node). Its parents are $X_2(t)$ and $X_3(t+1)$. Because $X_3(t+1)$ is still in the unobserved future, it cannot be used to produce a forecast. We have to go back to its own parents, $X_2(t-1)$ and $X_3(t)$, which are already known at time t. Therefore, we can include them in the set of selected variables $P(3,2)$ – (in blue). This figure is adapted from Fig.1 in [24].

performances of our forecasting methods. For this reason, we chose to keep a subset of sensors with no large missing data gaps, even though this meant learning with less sensors.

B. Sensor geographical location

The Lyon dataset comes from 600 sensors embedded in the streets downtown. In this work, we focus on a cleaned subset of 44 sensors which are displayed in Figure 2. It shows us that the 44 selected sensors are grouped into 3 clusters such that sensors in different clusters are sufficiently far apart to rule out direct causal relationships between them, at least in the 3 first time delays (*i.e.*, 18 minutes). Therefore, a good variable selection algorithm should discover and use this structure of almost independent clusters.

The Marseille dataset comes from 24 sensors embedded in an urban freeway which are displayed in Fig. 3.

C. Aggregation steps

One time step is equal to 6 minutes in the Lyon dataset and 2 minutes in the Marseille dataset. In Lyon, we had no control on this parameter, as the data collected were already aggregated. For Marseille, the aggregation time has been chosen to be consistent with the average time one vehicle needs to go from one sensor to the next. It is important to keep this in mind when analyzing figure 5, where $h = 1$ corresponds to a two minute horizon for Lyon and a six minute horizon for Marseille. Note also that the speed limits are very different in the two datasets: this limit varies between 30 and 50 km/h in the Lyon dataset, and between 90 and 110 km/h in the Marseille dataset. As a consequence, the length traveled by a vehicle during a same duration is usually much higher in the Marseille dataset than in the Lyon dataset. This counterbalances the fact that time steps are smaller.

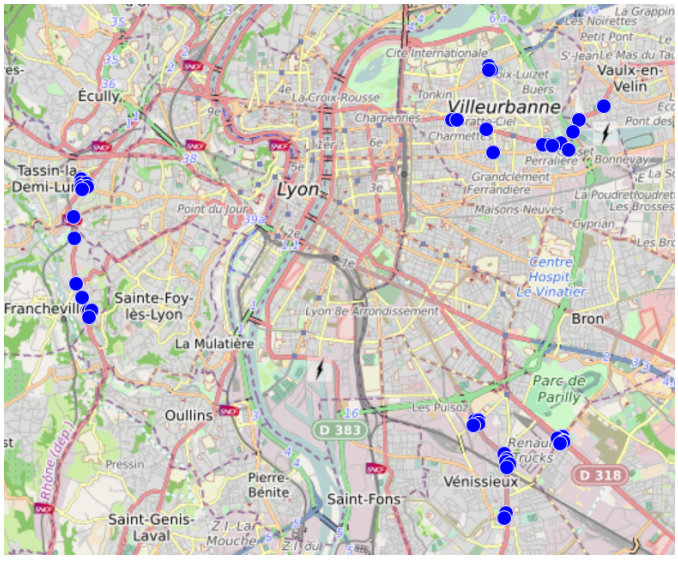


Fig. 2: Location of the 44 sensors in Lyon conurbation. The three clusters are chosen in such a way that sensors in different clusters are far enough to exclude causal dependence between them, whereas sensors within a same cluster may be dependent. A dependence structure learned with a good variable selection algorithm should reflect this.

Fig. 3: Geographical location of the 24 sensors in the Marseille dataset. The sensors follow each others on an urban freeway and detect the traffic in both directions.

We display in Fig. 4 an example of one day of flow measures (aggregated by time steps) on two road segments in Lyon and Marseille. Both plots have high frequency variations, but the amplitude of these variations is larger for Lyon than for Marseille, leading to more difficult forecasting.

D. Experimental setup

For both dataset, models are trained on a training set of 20 week days (excluding Saturdays and Sundays because they have different dynamics) and the prediction is made on the next 4 week days. For the Marseille dataset, we have also made experiments on vacation days. As results were stable on both sets and for brevity, we only display results for the four week day datasets.

Different scientific software libraries were used for this experiment. The Python library *scikit-learn* [25] was used to learn Lasso and `SVR` models. The Python module *statsmodels* [26] was used to learn `VAR`. For `ARIMA` model, we used the implementation of the R package *forecast* [20]. K-`NN`

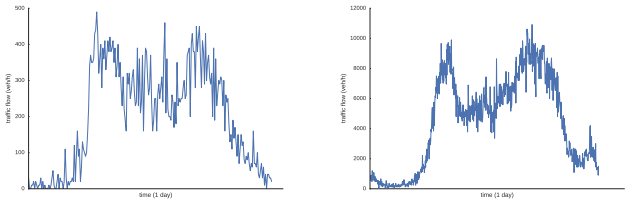


Fig. 4: Example of raw data for one sensor of the Lyon dataset (left) and the Marseille dataset (right): the x-axis corresponds to time (24 hours of measures) and the y-axis to flow (number of vehicle per hour).

approaches were reimplemented, but the causal graph for TiGraMITe was learned with *tigramite* Python API [27].

All experiments were performed with the computing resources of the CNRS/IN2P3 Computing Center (Lyon/Villeurbanne - France).

We use the Mean Absolute Scaled Error (MASE) [28] to evaluate forecasting accuracy, because it is scale invariant. The time series can have very different amplitudes due to the difference of traffic volume on different roads. Using RMSE would not allow us to reason about the aggregated performance over all the time series, because it is not scale invariant.

E. Univariate vs Multivariate

The different linear or time series models are compared in Fig. 5a, K-NN approaches in Fig. 5b and SVR approaches in Fig. 5c. As expected, multivariate methods (K-NN-*multi*, VAR, and SVR-RBF-*multi*) almost always perform better than their univariate counterparts (K-NN-*uni*, ARIMA, and SVR-RBF-*uni*) for all prediction horizons, because the latter cannot use the information of nearby road segments.

The only exception occurs on Lyon dataset for large prediction horizons, where SVR-RBF-*multi* has a very bad prediction performance, while SVR-RBF-*uni* has a slightly better performance but is still not competitive with methods from other families.

F. Evaluation of the interest of selection methods

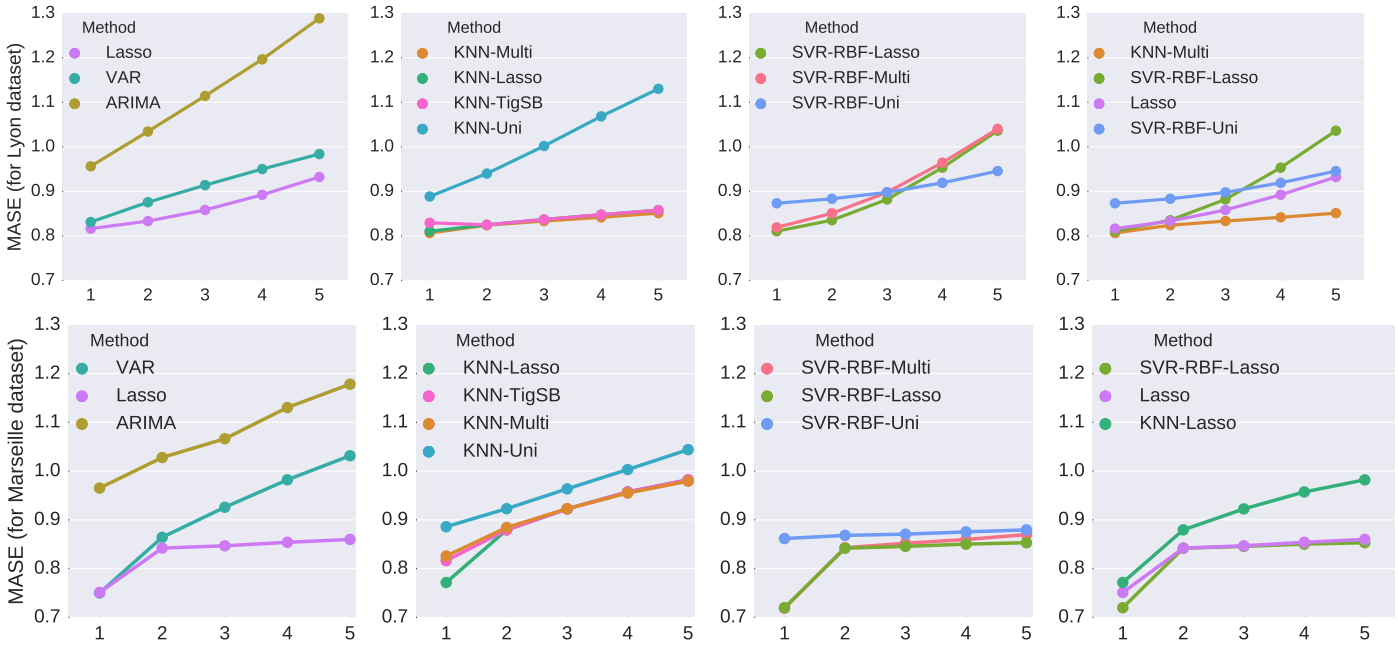
a) Linear and time series approaches: On both datasets, Lasso performs better than VAR. While the difference is subtle in the urban context, it is clearer for the freeway traffic. We will show in the next paragraph that this is coherent with the observations made on K-NN methods, that it is easier to find a relevant set of predictors for freeway data. The generalization advantage of Lasso becomes larger with the forecasting horizon.

b) K-NN approaches: On Lyon dataset, we observe that the best K-NN approach for all horizons is K-NN-*multi*. This means that the variable selection performed by K-NN-*lasso* and K-NN-*tig*SB does not improve the forecasting accuracy of the algorithm. On the Marseille dataset, the K-NN-*lasso* is the best for one-step horizon and performs as well as K-NN-*multi* for the other horizons. This suggest that the dependence graph learned with the K-NN-*lasso* is relevant. However, K-NN-*tig*SB is again not able to select a suitable set of predictors. The TiGraMITe graph with the best performance on the validation set is a very dense graph, meaning that the variables used for the prediction are almost the same as those used in K-NN-*multi*.

The difficulty to learn a dependence structure for the Lyon dataset which improves the prediction performance may be surprising, especially given the nature of the urban traffic phenomenon. A possible explanation is the fact that the causal relationship between road segments depends on their traffic state: a segment with fluid traffic causally influences downstream segments, whereas a segment with congested traffic rather impacts upstream segments (as traffic waves of congestion propagate backwards relative to vehicle directions). This property is difficult to capture if we learn a single dependence graph. Hence, we have segmented the day into five distinct regimes using expert knowledge about traffic flow (calm night period, morning and afternoon peak and intermediate regime in between) [2]. However, this was not sufficient to learn a proper dependence structure. This is probably due to the fact that in urban areas, the global traffic state in the network is composed of local traffic conditions on every segment. Different road segments can switch from one regime (fluid/congested) to another at different time steps, and there is no reason why this switch should occurs simultaneously on every road segment. Therefore, throughout the day, at each time step, it is possible that some road segments switch from one regime to another, changing locally the dependence structure with its neighbors and therefore requiring a different dependence graph. This helps us understand why segmenting the day in five different periods (identified by traffic experts) is still insufficient to capture the very unstable dependence structure. This very high dynamics makes the forecasting task very difficult.

The fact that K-NN-*lasso* performs well on the Marseille dataset and that K-NN-*tig*SB does not indicates that a simple linear and parametric approach is better suited to the modeling of freeway traffic than the information theory approach, with very few hypotheses about the nature of the dependence. The parsimony of lasso is an advantage for better generalization, and having good restricting assumptions on the dependence structure reduces the number of observations needed to converge to a good model.

We can observe that the performances of K-NN-*tig*SB and K-NN-*lasso* tend towards that of K-NN-*multi* when the horizon is large. This can be explained by the fact that when the horizon increases, the number of parents used in prediction also increases, and therefore we are closer to a setting where the set of selected variables is the full network. For example, if we want to forecast x_i at horizon $h = 3$, we search in the graph the direct parents of $X_i(t+3)$, that we will note $\mathcal{P}_{X_i(t+3)}$. We want to select them as predictors but some of them may still be in the unobserved future, for example if $X_j(t+1) \in \mathcal{P}_{X_i(t+3)}$, so their information is not available at time t. Therefore, in order to retrieve this information, we replace them by their parents, for example $\mathcal{P}_{x_j(t+1)}$ and do



(a) linear / time series (b) k-NN (c) SVR (d) best methods

Fig. 5: Comparison of forecasting methods: (top) Lyon dataset; (bottom) Marseille dataset; (left) k-NN approaches; (middle left) time series/linear approaches; (middle right) SVR approaches ; (right) best approaches. The y-axis gives the MASE and the x-axis gives the time horizon $h \in \{1, 5\}$.

this recursively until the selected variables are all evaluated at time t or prior. This procedure gives us the set of selected variables $P(i, 3)$ used to predict $X_i(t + 3)$. At each step of this recursion, a variable is replaced by its set of parents, which may contain a lot of variables. This is the reason why the number of selected variables increases with the forecast horizon, and tends towards selecting the whole network if the dependence graph is connected. The denser the graph, the sooner this happens. This outlines the need to keep the graph as sparse as possible.

c) SVR approaches: SVR-RBF-*Lasso* seems slightly better than SVR-RBF-*multi* on both datasets, but the difference is too small to be considered significant.

On the Lyon dataset, SVR-RBF-*multi* and SVR-RBF-*Lasso* have a good performance for small forecasting horizons ($h \leq 3$). This performance collapses for larger horizon, meaning the the model was not able to capture the increasingly complex dynamics.

G. Best methods for both datasets

In Fig. 5d, we show the methods which are at least once the best in their category (Linear, K-NN or SVR). On the Lyon dataset, K-NN-*multi* has the best performance, and the difference with other methods increases with forecasting horizon. Lasso performs better than SVR. On the Marseille dataset, K-NN-*multi* is clearly beaten by Lasso and SVR, whose performances are very close. SVR is better for one-step horizon and Lasso is slightly better for larger horizon, but this is not enough to draw significant conclusions.

The superiority of the non-parametric K-NN approach for urban traffic forecasting can be interpreted as the difficulty to learn a parametric model which is globally valid on all the test set. The advantage of K-NN is to find, for each step of prediction, some past conditions similar to those of the time of prediction without the hypotheses of a constant structure or model. In Fig. 6, we provide an example of one day of prediction with the best method for one sensor of each dataset.

Let us conclude this experimental evaluation with a comparison of computational aspects. In this application, models must be trained frequently (*e.g.*, every day), in order to incorporate recent data and to be robust to changes in the network. They should also be able to produce forecasts in real-time to be used by practitioners. A brute-force K-NN does not require offline training, but online forecasts may become expensive depending on the size of history. We may speed-up online forecasting by creating offline a tree-based data structure. Lasso and SVR both require offline training. Lasso is much faster than SVR both for offline learning and online forecasting.

IV. Conclusion

We presented several contributions to short-term urban road traffic forecasting. First, we evaluated a benchmark of ten forecasting methods on real-world city traffic data in two different contexts. We showed that multivariate approaches are essential. In the particular city centre context, we showed that the nonparametric approach (K-NN) always outperforms other parametric methods and that variable selection algorithms do

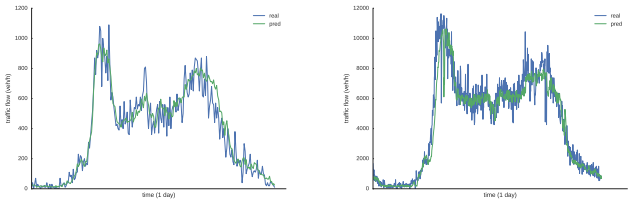


Fig. 6: Example k-NN forecasts of urban traffic in Lyon (left) and Lasso forecast of freeway traffic in Marseille (right), for $h = 5$ in both cases.

not improve the predictions: capturing relevant causal drivers is too difficult a task because of the very high dynamics. In the freeway context, we showed that parametric approaches (Lasso, SVR) perform better, probably due to simpler traffic dynamics, and Lasso variable selection is useful (even for K-NN) especially for large horizons. This is surprisingly not the case for TiGraMITe, maybe because it does not make assumptions about the nature of the dependence (as Lasso does) that would avoid overfitting. We advise practitioners to prefer Lasso to SVR because it is faster for learning and forecasting and is easier to interpret.

REFERENCES

[1] E. Vlahogianni, M. Karlaftis, and J. Golias, "Short-term traffic forecasting: Where we are and where were going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, 2014.

[2] Y. Kamarianakis, W. Shen, and L. Wynter, "Real-time road traffic forecasting using regime-switching space-time models and adaptive lasso," *Applied stochastic models in business and industry*, vol. 28, no. 4, pp. 297–315, 2012.

[3] B. Pan, U. Demiryurek, and C. Shahabi, "Utilizing real-world transportation data for accurate traffic prediction," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, 2012, pp. 595–604.

[4] L. Li, X. Su, Y. Wang, Y. Lin, Z. Li, and Y. Li, "Robust causal dependence mining in big data network and its application to traffic flow predictions," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 292–307, 2015.

[5] J. Haworth, J. Shawe-Taylor, T. Cheng, and J. Wang, "Local online kernel ridge regression for forecasting of urban travel times," *Transportation Research Part C: Emerging Technologies*, vol. 46, pp. 151–178, 2014.

[6] B. Smith and M. Demetsky, "Traffic flow forecasting: comparison of modeling approaches," *Journal of transportation engineering*, vol. 123, no. 4, pp. 261–266, 1997.

[7] B. Smith, B. Williams, and K. Oswald, "Comparison of parametric and nonparametric models for traffic flow forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 4, pp. 303–321, 2002.

[8] G. Davis and N. Nihan, "Nonparametric regression and short-term freeway traffic forecasting," *Journal of Transportation Engineering*, vol. 117, no. 2, 1991.

[9] X. Jiang and H. Adeli, "Dynamic wavelet neural network model for traffic flow forecasting," *Journal of transportation engineering*, vol. 131, no. 10, 2005.

[10] Y. Zhang and Z. Ye, "Short-term traffic flow forecasting using fuzzy logic system methods," *Journal of Intelligent Transportation Systems*, vol. 12, no. 3, pp. 102–112, 2008.

[11] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Trans. on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[12] E. Vlahogianni, M. Karlaftis, and J. Golias, "Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 3, pp. 211–234, 2005.

[13] S. Sun, C. Zhang, and G. Yu, "A bayesian network approach to traffic flow forecasting," *IEEE Transactions on intelligent transportation systems*, vol. 7, no. 1, pp. 124–132, 2006.

[14] A. Nair, J. Liu, L. Rilett, and S. Gupta, "Non-linear analysis of traffic flow," in *Intelligent Transportation Systems, 2001*, 2001, pp. 681–685.

[15] C. Queen and C. Albers, "Intervention and causality: forecasting traffic flows using a dynamic bayesian network," *Journal of the American Statistical Association*, vol. 104, no. 486, pp. 669–681, 2009.

[16] J. Runge, J. Heitzig, N. Marwan, and J. Kurths, "Quantifying causal coupling strength: A lag-specific measure for multivariate time series related to transfer entropy," *Physical Review E*, vol. 86, no. 6, 2012.

[17] J. Runge, J. Heitzig, V. Petoukhov, and J. Kurths, "Escaping the curse of dimensionality in estimating multivariate transfer entropy," *Physical review letters*, vol. 108, no. 25, p. 258701, 2012.

[18] J. Runge, V. Petoukhov, J. Donges, J. Hlinka, N. Jajcay, M. Vejmelka, D. Hartman, N. Marwan, M. Paluš, and J. Kurths, "Identifying causal gateways and mediators in complex spatio-temporal systems," *Nature communications*, vol. 6, 2015.

[19] C. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: Journal of the Econometric Society*, pp. 424–438, 1969.

[20] R. J. Hyndman and Y. Khandakar, "Automatic time series forecasting: the forecast package for R," *Journal of Statistical Software*, vol. 26, no. 3, pp. 1–22, 2008. [Online]. Available: http://www.jstatsoft.org/article/view/v027i03

[21] H. Chandra, S.R. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[22] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[23] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural Networks*, vol. 17, no. 1, pp. 113–126, Jan. 2004. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0893608003001692

[24] J. Runge, R. Donner, and J. Kurths, "Optimal model-free prediction from multivariate time series," *Physical Review E*, vol. 91, no. 5, 2015.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[26] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with python," in *9th Python in Science Conference*, 2010.

[27] J. Runge, D. Sejdinovic, and S. Flaxman, "Detecting causal associations in large nonlinear time series datasets," *ArXiv e-prints*, Feb. 2017.

[28] R. Hyndman and A. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.