Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2018) 000–000

Transportation Research Procedia

www.elsevier.com/locate/procedia

World Conference on Transport Research - WCTR 2019 Mumbai 26-31 May 2019

Toward Reliable Estimations of Urban Traffic Flows from Machine Learning and Floating Car Data

Jinjian LI*, Jacques Boonaert, Arnaud Doniec, Guillaume Lozenguez

IMT Lille Douai, Univ. Lille, Unité de Recherche Informatique et Automatique, F-59000 Lille, France

Abstract

This work presents an analysis of applying Floating Car Data (FCD) from Google Maps for reconstructing urban traffic flows with a Gaussian Process Regressor (GPR) tuned using machine learning method. Firstly, the travel duration on roads is requested through the Google Maps API. Then, based on this travel duration, the proposed method estimates the traffic flow by applying the GPR machine learning approach. This is an innovative work to estimate and reconstruct the traffic flow in urban transportation street networks with machine learning method from aggregated FCD. A series of experiments are conducted to compare the estimated traffic flow calculated by using GPR machine learning method and the real one from actual sensors data. The obtained results show that, based on aggregated FCD data, the proposed method can always reproduce and capture the tendency of real traffic flow. Furthermore, the performance can be improved by regulating parameters in GPR machine learning model, such as half width of sample window and sample size (a whole week or only weekdays). Therefore, the proposed GPR machine learning and FCD based new method can replace those traditional loop detectors for the measurement of traffic flow.

Keywords: Estimation of urban traffic flows; Gaussian Process Regression (GPR); Machine Learning; Floating Car Data (FCD)

1. Introduction

With the rapid growth of urban centers during the last decades, the development of efficient public transportation services has become a central issue. The resulting increasing demand in terms of transportation flows has to cope with the difficulty to adapt existing or create new transportation networks. In this context, exploiting the corresponding infrastructures as efficiently as possible is a mean to provide a better quality of service while minimizing the requirement of costly and time-consuming investments. One of the major basements on which such an optimization process can be though is the ability to produce models (and specially simulation models) representing the way that transportation flows evolve with time, depending on the traffic demands, the operational conditions and particular events occurrence (accidents, traffic signals malfunction, etc.). Such models, in return, can be exploited in many different ways: they can be used to exhibit some of the transportation networks' weaknesses or lack of robustness, they can also simulate the effects of changes in the operational conditions (change of the bus lines schedules or paths, local modification of the network, etc.) and they can greatly contribute to efficiently monitor the traffic in real time, using as few sensors as possible.

Whatever the final application of these models are, a prior step to the construction of such a model is to collect (often large) datasets. Our motivation is to be able to simulate the urban traffic phenomena for a mid-sized city. To do so, we have to provide to the simulation a realistic traffic demand at each entrance of the city (i.e. the flow of incoming vehicles). On the one hand, even in mid-sized cities, continuously gathering the required sensors data can be too time consuming or costly, making the mentioned simulation process out of reach. On the other hand, there are already different accessible databases that can provide useful information regarding the transportation condition (travel duration) at a given location and at a given time. Those platforms generally provide only aggregated data like average travel duration more than the initial raw data. At this step, a central point is to know if these aggregated data are precise enough to replace the "in site" gathered data. If so, it should be possible to build or, at least, update traffic models from these aggregated data with a sufficient confidence instead of organizing in site and necessarily time limited collects. In this context, this paper presents a machine learning based approach aimed to reconstruct "sensor like traffic flow data" from available aggregated information.

In this work, instead of using stationary devices (such as loop detectors (Cheung, 2005) or video cameras (Coifman, 1998)) to measure traffic flows, we attempt to estimate these data according to Floating Car Data (FCD) from Google Maps only on the basis of Regressor trained using machine learning techniques. To the best of our knowledge, this is an innovative approach that could help minimizing the use of stationary sensors while providing a good enough estimation of traffic flows. The main contributions of this paper are as follows: (I) Introduction of an innovative method for the reconstruction of traffic flows based on the Google's FCD. (II) Demonstration of the capability of the Gaussian Process Regression (GPR) machine learning method to deal with this problem. Experiments are conducted by comparing estimated flows with real one provided from induction loop sensor. The results we obtain seem promising enough to say that correct urban transportation flows models could be obtained with a very light use of real traffic sensors.

This paper is organized as follows: the next section describes related works and the different usages of aggregated FCD in the context of urban transportation networks modeling. The third section focuses on the problem we choose to address, that is building sensor like data flow measurements from aggregated data. In this section, we also provide details regarding our problem formulation on the standpoint we took for solving it. The fourth section presents the GPR machine learning method for the estimation of traffic volume. The fifth section deals with the experimental site, the results we obtained, the comparison between estimated traffic flow and real observed data prior to the discussion. The last section concludes this paper and presents some perspectives and further works based on it.

2. Related work

The successful wide scale deployment of the Advanced Traveler Information Systems (ATIS) and Advanced Traffic Management Systems (ATMS) highly relies on the capability to perform accurate estimation of the real traffic states on road networks. Therefore, the use of real-time Floating Car Data (FCD), based on traces of Global Positioning System (GPS) positions of vehicles, is emerging as a reliable and cost-effective way to collect accurate traffic data for a wide area road network. Unlike other traffic data collection techniques (e.g. traffic cameras, induction loops embedded in the roadway, radar-based sensors), floating cars act as moving sensors traveling in a traffic stream and do not require additional instrumentation to be set up on the roadway.

The main communication architecture for FCD is based on the exchange of information between a fleet of floating cars traveling on a road network and a central data operation system. The floating cars periodically send their positions (latitude, longitude and altitude) and instantaneous velocity thanks to GPS receiver and Global System for Mobile communications (GSM) or General Packet Radio Service (GPRS) transmitter. While the central data operation system tracks the received FCD along the traveled path by matching the related trajectories data to the corresponding real road network. The frequency of sending/reporting is generally determined by the required resolution of the data and the performances of the available communication channels, for example, bandwidth. Therefore, FCD is an effective approach to determine the real traffic speed on the road network, based on gathering of data such as localization, speed, direction of travel and time information from the mobile phones of drivers and passengers of the vehicle. In other words, every vehicle with an active mobile phone acts as a sensor for the network.

Using FCD for estimating travel duration and traffic states has received high attention over the years from the scientific community. The most common and useful information provided by FCD is travel duration and speeds along road links or paths (Miwa, 2018; Turksm, 2000; Yoon, 2007). Works such as (DeFabritiis, 2008; Nanthawichit, 2003) calculate the travel duration on roadside according to the speed of the floating car on that road. Event detection, such as congestion and accidents has been discussed by using the trajectories of floating cars (Asakura, 2015). The percentage of floating cars required for the estimation of travel duration is presented in the works like (Dai, 2003; Hong, 2007). Reconstructing the traffic states from FCD has been previously addressed in papers such as (Fabritiis, 2008; Hong, 2007; Kerner, 2005; Sunderrajan, 2016).

In the work (Fabritiis, 2008), the authors attempt to estimate and predicte the travel speed with the real-time FCD based on traces of GPS positions. Another work like (Hong, 2007), both spatial and temporal characteristics to the domain of floating car sampling is introduced. And the analysis of this work can provide an insight for floating car based traffic state system designers on the transmitting period, sampling interval and penetration of floating car that are desirable in a traffic network in order to get certain coverage and accuracy in traffic state estimation. An interesting method is proposed in (Kerner, 2005), whose purpose is obtain a high quality on the reconstruction of travel times in the net with smaller percentage of FCD vehicles and amount of FCD messages. The most recent research about traffic state reconstruction using FCD is presented in work (Sunderrajan, 2016). The speed is estimated based on FCD. Then the authors make use of the R^2 *Statistic* to build the function between average speed ($\overline{V}$) and density (ρ). Finally, the traffic flow is calculated according to the Fundamental diagram ($Q= \rho * \overline{V}$). Furthermore, the floating car penetration required for accurate traffic state estimation is also discussed.

In fact, the above works focus on two main paths: (I) method of collecting FCD; (II) estimation of traffic state by building traditional function model. However, in this work, at the one hand, the aggregated FCD in Google Maps is applied directly, because it is the most widespread and best-known system from a more practical and industrial point of view. At the other hand, an estimation of the traffic flow is done with machine learning method. In fact, since 2007, real-time traffic information are available from Google Maps, but unfortunately Google does not provide access to the raw data but only aggregated information. The Google Maps users can either view on web site and app the speed of traffic through colored (4 colors are available: green for normal speed of traffic, orange for slower conditions, red for congestion and dark red for stopped traffic) road section or request estimated travelling time duration by using the dedicated Application Programming Interface (API).

3. Problem Description

In this section, we present the problem to be solved, that is, for a given lane, finding the relationship between vehicles' travel duration and traffic flows, as illustrated at Fig.1. On the one hand, travel duration can be "easily" obtained using the API provided by Google Maps. On the other hand, flow velocity and traffic flow are linked considering the Fundamental diagram of traffic flow. However this function relationship should be nonlinear and difficult to formulate (Wu, 2006; Lam, 1992). Thus, our idea is to train a Gaussian Process Regression (GPR) to establish a relation between travel duration and traffic flow. The Fig.2 illustrates this idea.

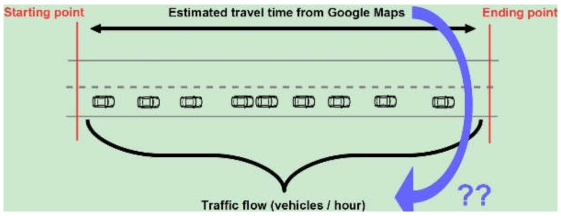
Starting point
Estimated travel time from Google Maps
Ending point
Traffic flow (vehicles / hour)
??

Fig. 1. Illustration of our problem

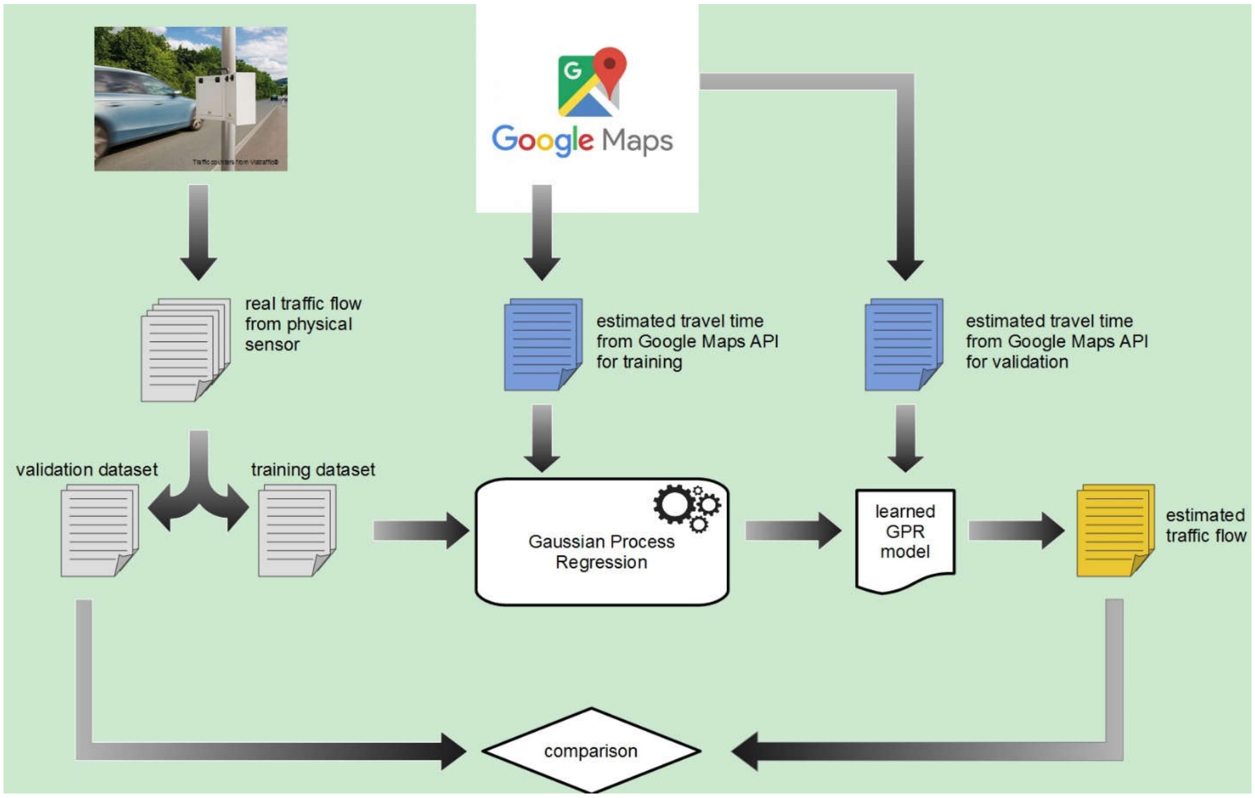
Google Maps
real traffic flow from physical sensor
estimated travel time from Google Maps API for training
estimated travel time from Google Maps API for validation
validation dataset
training dataset
Gaussian Process Regression
learned GPR model
estimated traffic flow
comparison

Fig. 2. System overview

Considering a given route, we start by measuring traffic flows over a certain period of time (say 7 days for example) using a traffic sensor. The obtained data are split in two sets: a training set and a validation set. Over this same period of time, we request Google servers to collect every 10 minutes estimated travel duration. The training set and the corresponding estimated travel duration are used by the GPR. At the end of the learning process, we obtain a model that can now take estimated travel duration as input in order to predict traffic flow as output. To validate our approach, we compare the validation set to the flow estimated by our model. In the following subsections, we present the type of data we use and how we state our regression problem.

In the context of this work, both real traffic flow measurements for a given circulation lane and the corresponding travel duration obtained by FCD from the Google Maps API are used. Let symbol $f(t)$ denotes the real traffic flows (*veh/h*) measured by a traditional sensor on the given lane, like loop detector, and $d(t)$ the travel duration (s) estimated by Google Maps for passing the given lane.

In practice, traffic flow signal $f(t)$ is sampled and calculated with a period T_{flow} corresponding to one hour with traditional sensor. However the travel duration signal $d(t)$ is sampled in Google Maps at a period T_e, which is equal to 10 minutes. To cope with these different sampling periods, data are linearly interpolated so that we consider that both $d(t)$ and $f(t)$ are sampled with sample period T_e . The data received from each floating car is shown as following, which are used to calculate the travel duration on a given lane by Google Maps with a complicated algorithm:

1. Lane name for starting and ending locations. This element is applied to locate the observed lane on Google Maps.
2. GPS place for starting and ending location of given lane. This element is used to calculate the observed distance on the lane.
3. Distance (m). The length between starting and ending locations on the Google Maps.
4. Starting time. The beginning of observation sample, whose interval is T_e (10 minutes).
5. Ideal travel duration. The travel time between the starting and ending locations.
6. Real travel duration for a floating car. The travel time for a floating car considering real time traffic state.

For the sake of simplicity, in the following, we denote $d(k\cdot T_e) \equiv d_k$ and $f(k\cdot T_e) \equiv f_k$, which means that the travel duration or travel flow can be represented as d_k or f_k, respectively, for k^{th} sample at time $k\cdot T_e$.

The assumption made in this work is as followings: for a given time-step k, the traffic flow in point f_k can be approximated from a $2\cdot n + 1$ sample window of traffic duration. The variable n ($n \in N^*$) refers as half width of sample window acquired from $d(t)$, centered at the time step k, as shown in the Fig.3 with a red rectangle. Therefore, the input vector X_k for machine learning is then built as $X_k = \{d_{k\text{-n}} \dots d_{k\text{-}1} \;\; d_k \;\; d_{k+1} \dots d_{k+n}\}$. The output is traffic flow f_k at k^{th} sample time step. In other words, the task is to estimate the traffic flow at k^{th} sample time step f_k by applying machine learning method, based on a sequence of travel duration samples X_k from FCD in Google Maps.

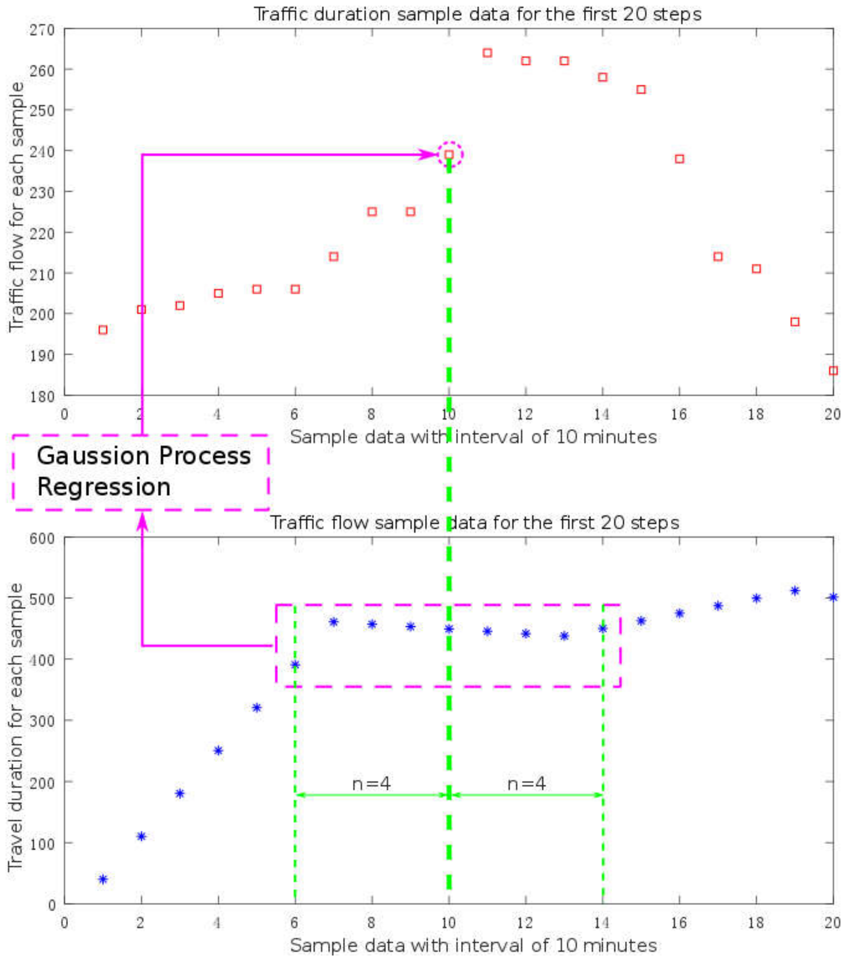

Figure 3: Window sample used for approximation

4. Gaussian Process Regression machine learning for traffic volume estimation based on the FCD

This section presents the application of the Gaussian Process Regression (GPR) (Rasmussen et al. 2004) machine learning method for the estimation of traffic volume from the Floating Car Data (FCD) of Google Maps. Firstly, the background of GPR machine learning method is introduced. Then the algorithm of applying the GPR for the estimation of traffic flow is explained.

The GPR (Xie et al. 2010) is a supervised machine learning method that offers mapping function between input and (continuous) output data. The Gaussian Process framework is used in different areas extended from classification to regression problems, including speed estimation (Hu et al. 2015), travel duration estimation (Idé, 2009), time-varying systems (Hu et al. 2014) etc. In this work, a GPR model was adopted to model and estimate traffic volume from the Google aggregated FCD. Because the traffic volume is expected to have some complex relationship with the travel duration on the same road, simple parametric models such as linear or polynomial functions is inappropriate for this task (Wu et al. 2006).

4.1 Gaussian Processes (GPs) formulation

Generally, GPs offer a Bayesian paradigm to learn an implicit functional relationship $\hat{y} = \hat{f}(x)$, according to a given training data set, $D = \{(X_i^d, y_i) | \, i \in \mathrm{N}\}$. The variable N is the size of the data set. The symbol X_i^d represents a vector for the i^{th} observed input variable (also named predictor, regressor, control, or independent) in a d-dimensional feature space. And y_i is a one-dimensional observed target value (also named predicted, regresses, response, or dependent), which is either continuous or discrete. However, unlike most classical Bayesian models (Jensen, 1996), GPs directly infer a prior distribution on the whole function $\hat{f}(X)$. In other words, function $\hat{f}(X)$ is treated as a random field and is assumed to be a GP a prior, as the Eq.(1) shows.

$$(\hat{f}(X)|\theta \propto GP\,(m(X), k\,(X, X')) \quad (1)$$

Where, the prior GP is fully defined by a mean function $m(X)$ and a covariance function $k(X, X')$. The notation θ means the prior's hyper-parameters applied to parameterize the covariance function, as follows:

$$K\,(X, X') = k(X, X'; \theta\,) \quad (2)$$

Strictly speaking, a GP model can also be treated as a probability distribution, which is defined over the following functions:

$$E[\hat{f}(X)] = m(X) \quad (3)$$

$$Cov[\hat{f}(X), \hat{f}(X')] = k(X, X') \quad (4)$$

where $\hat{f}(X), \hat{f}(X')$ are random variables that are indexed by any pair of X and X'. Then, a GP prior can be roughly considered as a probability distribution for an infinite number of random variables. Furthermore, a collection of function values, which are indexed by any finite number of $X = [x_1, x_2, ..., x_n]^T$, e.g., $F(X)=[f(x_1), f(x_2), ..., f(x_n)]$, supposes a multivariate normal distribution in Eq.(5).

$$p(\hat{f}(X)\,|) = N(m(X), k(X, X')) \quad (5)$$

Where the average vector $m(X)$ and covariance matrix $k(X, X')$ are determined directly based on $m(\bullet)$ and $k(\bullet, \bullet)$, as following:

$$m(X) = [m(x_1), m(x_2), ..., m(x_n)]^T \quad (6)$$

$$K_{i,j} = k(x_i, x_j) \qquad i, j = 1, ..., n \quad (7)$$

For the sake of simplicity and without loss of generality, $m = (x) = 0$ is assumed, since the data can always be centered
by the sample mean.

With the machine learning term, $k\,(x, x')$is often named as a kernel function or simply a kernel instead of a covariance function. As detailed later, kernel functions generally take certain forms which are parameterized by one or several parameters θ. Therefore, a GP prior can be specified by determining a specific type of kernel function (also
named covariance function) and the associated θ values.

Once a GP prior $p(f\,|\theta)$ and a "noise" model $p(y\,|f)$ are determined, $p(f\,|D,\theta)$ the posterior distribution of f can be easily obtained by updating the prior $p(f\,|\theta)$ based on the Bayes theorem with the training data set D, as shown in Eq.(8).

$$p(f\,|D,\theta) = \frac{p(y\,|f)\;p(f\,|X,\theta)}{p(\,D|,\theta)} \quad (8)$$

where the input variables X should be made explicit in the prior and term $p(D|,\theta)$ is called Marginal Likelihood, since it is a function of variable θ and given data set D. The noise model \$p(y|f)\$ is also a likelihood, for the reason that it is a function of f for a fixed set of observations y. Here, the $p(y\,|f)$ is introduced, since y_i is a corrupted version of $f\,(x_i)$. Therefore, the estimation distribution for a new input x_{new} is achieved by using the Eq.(9) with the posterior $p(f\,|D,\theta)$

$$p(f_{new} \mid x_{new}, D, \theta) = \int p(f_{new}, f\,|D, \theta)\,df \quad (9)$$

by the combination with Eq.(9) and the noise model, the predictive distribution for y_{new} is achieved in the Eq.(10)

$$p(y_{new} \mid x_{new}, D, \theta) = \int p(y_{new}, f_{new} | D, \theta)\, df_{new} \tag{10}$$

From the Eq.(10), not only the estimated average values but also the associated uncertainty (error-bar) could be calculated. In the GP modeling, it is as collection of function values $f(x)$ needed to be Gaussian instead of variables x itself, which are assumed to be distribution-free. Therefore, the GP model theoretically can handle data with any kinds of distributions. For a more detailed presentation, interested readers can refer to (MacKay et al. 1997, Williams et al. 2006) for more information.

4.2. Gaussian Process Regression machine learning method

The GP model presented in the above subsection can solve non-linear regression problems, if the observed target value y_i is continuous, and the noise model $p(y \mid f)$ is assumed as a normal distribution. Then the GPR model can be expressed in the Eq.(11).

$$y_i = f(x_i) + \theta_i \qquad \theta_i \sim N(0, \sigma^2) \tag{11}$$

In this case, the inference of GPR model becomes analytically tractable, as a result of the Gaussianity of $p(y \mid f)$. Accordingly, for a new input x_{new}, the predictive mean and variance associated with $\hat{f}_{new} = f(x_{new}) = f_{new}$ are defined in Eq. (12)-(13), respectively (Rasmussen et al. 2004).

$$\mu\,(f_{new}) = k(x_{new}, X)\,[K(X, X) + \sigma^2 I]^{-1}\, y \tag{12}$$

and

$$Var(f_{new}) = k(x_{new}, x_{new}) - K(x_{new}, X)\,[K(X, X) + \sigma^2 I]^{-1}\, K(X, x_{new}) \tag{13}$$

Where X and y mean the observed predictors and observe target value. I is defined as the identity matrix.

4.3. Measurements of effectiveness

The Root Mean Square Error (RMSE) is a commonly used criterion to evaluation and compare different machine learning estimation methods. Here, it is defined below:

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N} (f(k) - \hat{f}(k))^2} \tag{14}$$

where, $f(k)$ is the observed traffic flow at time step $\hat{f}(k)$ is the corresponding estimated traffic flow. The time step in this work is 10 minutes. N is the total number of time steps for the testing data in training process.

5. Experiment and validation from the real data

In this section, we conduct a series of experiments to evaluate the proposed algorithm and compare the results with real data. Firstly, the protocol of our experiments are presented. Then, the comparison of RMSE among the four typical kernel function in Gaussian Process Regression (GPR) is shown for n (half width of sample window) from 4 to 24. Next, experimental performance during a whole week is presented in detail for the kernel function, named rational quadratic, because it can help us to achieve the lowest RMSE value, compared with others. Finally, an experiment with only weekdays, which performs better than the above case, is executed, for the reason that the profile of traffic flow exists big difference between weekdays and weekends.

5.1. Protocol and experiments description

The experiment is executed on the road "86 Boulevard de la République, Douai, France" for the validation of the proposed machine learning method. The data of travel duration and real traffic flow on this road for a whole week from Monday to Sunday is respectively acquired from Google Maps, as shown in Fig. 4 and City Hall of Douai in France, as Fig. 5 shows. The GPR parameters are as follows: 50 percentage of sample data is extracted as new data to test the performance of the trained GPR model, and the remainder is used to train and validate the GPR model. This work applies the 5-fold Cross Validation method to avoid the over-fitting. The following four different types of kernel function are applied in the GPR machine learning method: rational quadratic, squared exponential, Matern

5/2 and exponential. The range of *n* (half width of sample window) considered in this work is from 4 to 24, which means that the sample time interval is from 40 to 240 minutes, because each sample time step is 10 minutes. The units of parameters are as follows: traffic flow (*veh/h*), travel duration (*s*).

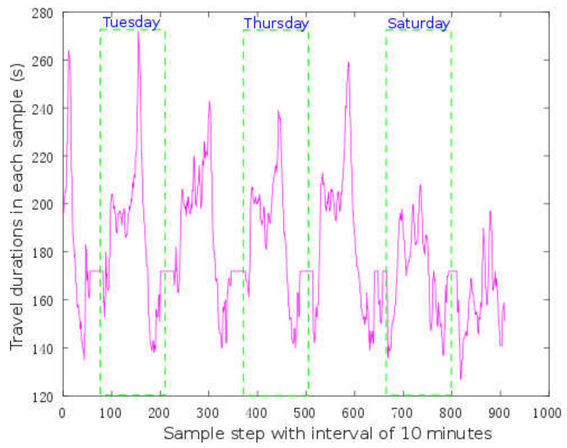

Figure 4: travels durations from Google FCD along a week

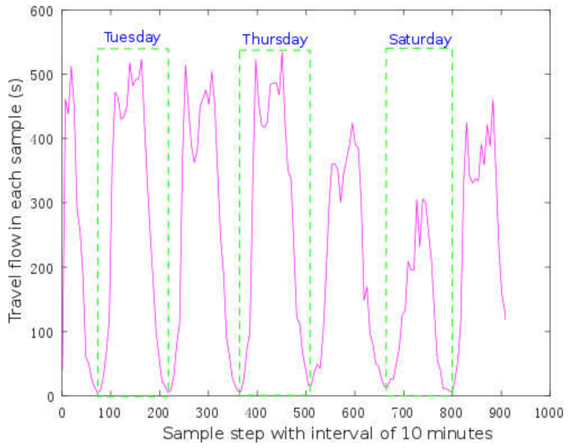

Figure 5: traffic flows from actual sensors measurements along a week

5.2. Comparison of RMSE among different kernel function in GPR

The experimental performances are compared among the four proposed kernel function in GPR under the n from 4 to 24 to find the most suitable kernel function. As shown in the Fig. 6, the GPR with kernel functions of rational quadratic can always achieve the best performance with lowest RMSE error under different half width of sample window than those with other kernel functions. The worst performance happens with the GPR under the kernel functions of Squared exp. (when *n* from 4 to 9) and Exp. (when *n* from 10 to 24). Globally, for the GPR with kernel functions of rational quadratic, when *n* increases from 4 to 23, the RMSE value accordingly decreases, and the lowest one achieved at the point (*n*=23) is 18.9 veh/h. Then RMSE value lightly increases, for the reason that when the *n* is too big, some data far away from the center point is not so related and give some extra noise influence to the estimation. However, the rate for reducing the RMSE is extremely difference when the *n* is augmented. For *n* from 4 to 15, the RMSE is reduced rapidly with an improvement of 65.3 veh/h. Then the RMSE is reduced slowly with an improvement of 10.3 veh/h for *n* from 16 to 24. Therefore, the best choice for *n* is 23 when the RMSE performance is the only criterion. Nevertheless *n* with 15 is the best choice if the RMSE performance and length of *n* should be considered together, because the bigger *n* is, the more data is needed to estimate traffic flow, as shown in the Fig. 3.

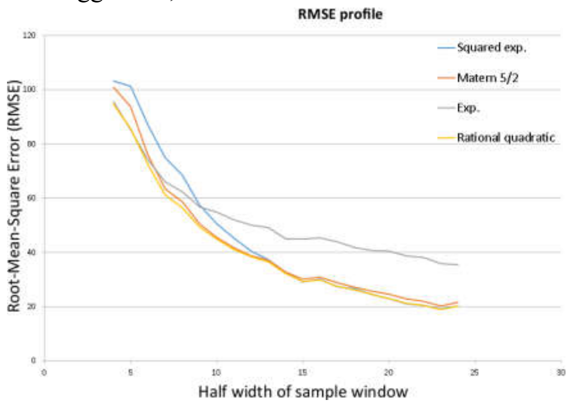

Figure 6: RMSE profile for GPR under different n

5.3. Experimental performance for a whole week

The comparison between real and estimated traffic flow and the distribution of errors is shown in detail with *n* as: 4, 8, 16, 24, as shown in Fig. 7-14. Firstly, with *n*=4, the trained GPR model can only capture the traffic flow tendency, because the n is too small and more information should be needed, as shown in the Fig. 7. As a result, for the corresponding distribution of error in the Fig. 8, only 45 percentage of errors locates between the zone [-50, 50] (veh/h). The biggest error happens in 300 veh/h and the RMSE is 92 veh/h. However, when *n* is increased to 24, estimated traffic flow is more similar to the real one than that with *n* = 4, as the Fig. 13 shows, which means that the increase of n can help to model the profile more exactly. Therefore, 86 percentage of errors locates in the zone [-50, 50] (veh/h), which is almost two times bigger than that with *n* =4. The RMSE is 20.18 veh/h. Third times lower than the one with *n* = 4. The biggest error is 170 veh/h. However, in Fig. 13, most of obvious error happens in weekends, because the profile's shape of traffic flow is very different between workdays and weekend, which motivates us to build a GPR model only for weekdays to improve the performance. This subject is discussed in detail in the following subsection.

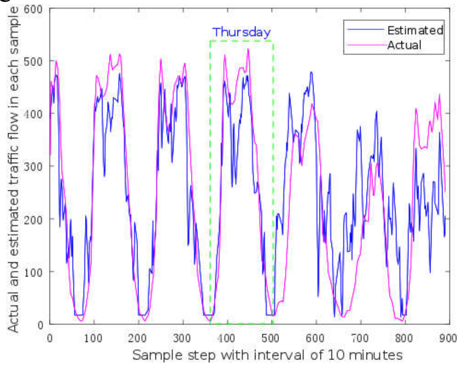

Figure 7: actual VS estimated traffic flow with n=4 for a week

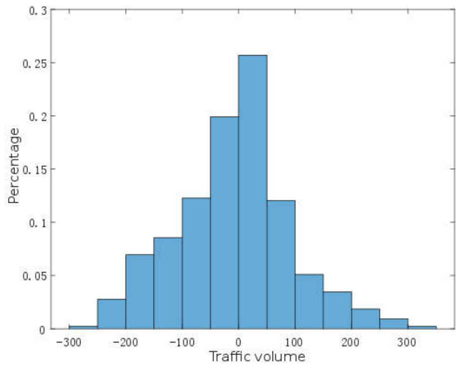

Figure 8: normalized estimation error with n=4 for a week

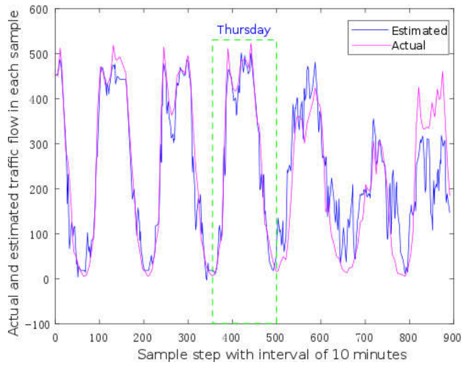

Figure 9: actual VS estimated traffic flow with n=8 for a week

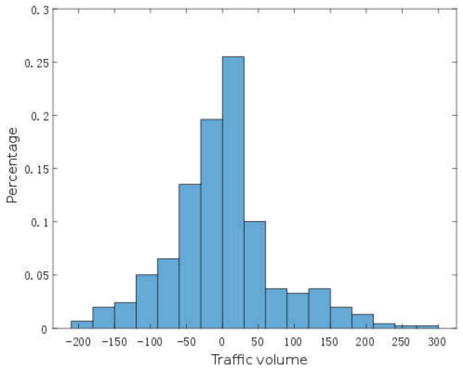

Figure 10: normalized estimation error with n=8 for a week

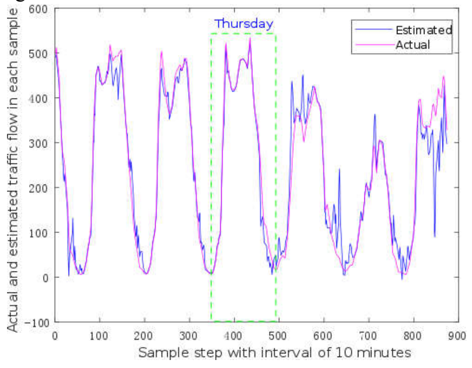

Figure 11: actual VS estimated traffic flow with n=16 for a week

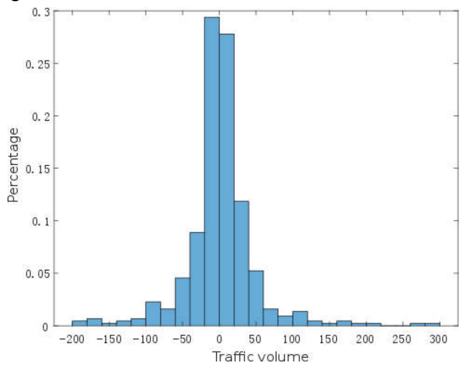

Figure 12: normalized estimation error with n=16 for a week

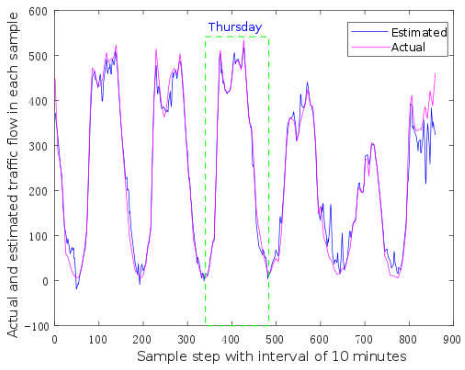

Figure 13: actual VS estimated traffic flow with n=24 for a week

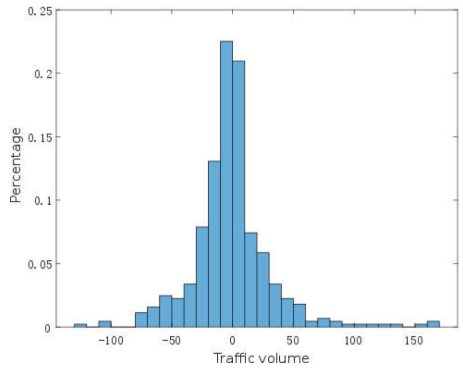

Figure 14: normalized estimation error with n=24 for a week

5.4. Experimental performance for only weekdays

The performance are compared between GPR models for whole weeks and only weekdays. For n=4, compared with the GPR model for whole week (refers to Fig. 7), the GPR model for only weekdays (refers to Fig. 15) can capture the real data more precisely, and can get higher percentage of errors (62.8%) locating between the zone [-50, 50]. Furthermore, when the n is increased to 24, as shown in the Fig. 22, 97 percentage of errors in in the zone [-50, 50] , which is 11% higher than that in the GPR model for whole week, as Fig. 14 shows. Therefore the performance can be improved by building a GPR model with smaller time zone, for example, only the weekdays instead of whole week.

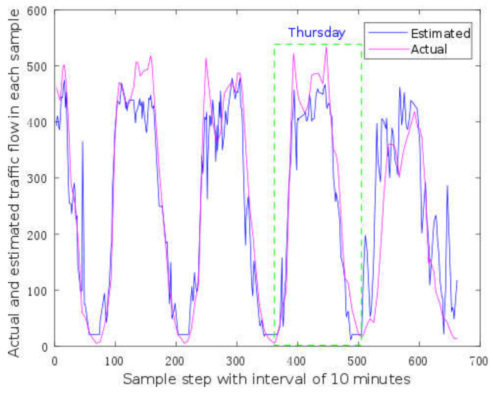

Figure 15: actual VS estimated traffic flow with n=4 for weekdays

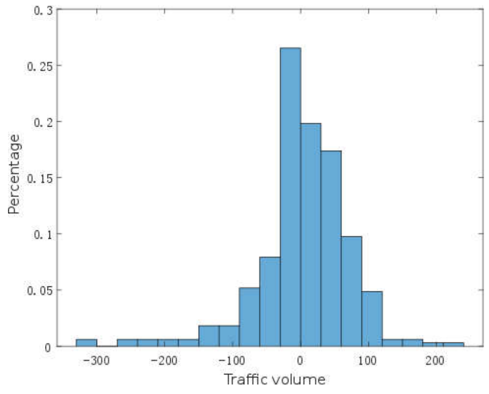

Figure 16: normalized estimation error with n=4 for weekdays

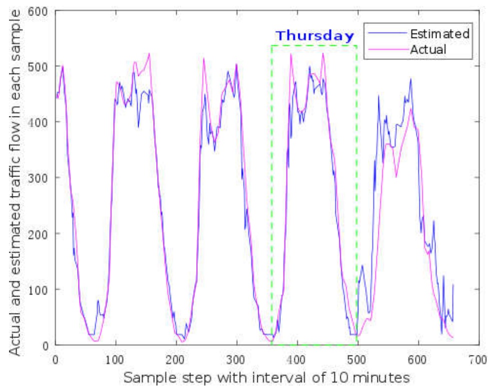

Figure 17: actual VS estimated traffic flow with n=8 for weekdays

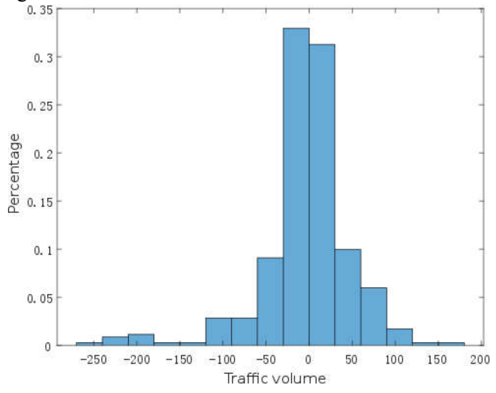

Figure 18: normalized estimation error with n=8 for weekdays

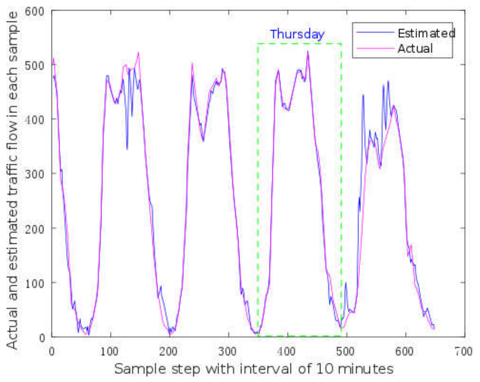

Figure 19: actual VS estimated traffic flow with n=16 for weekdays

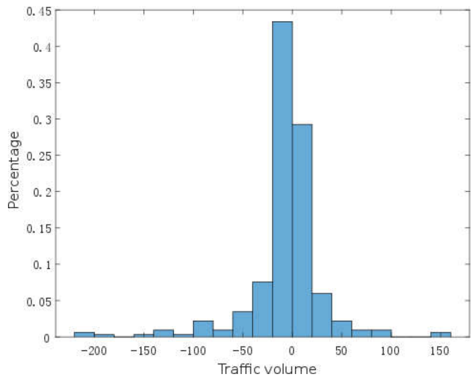

Figure 20: normalized estimation error with n=16 for weekdays

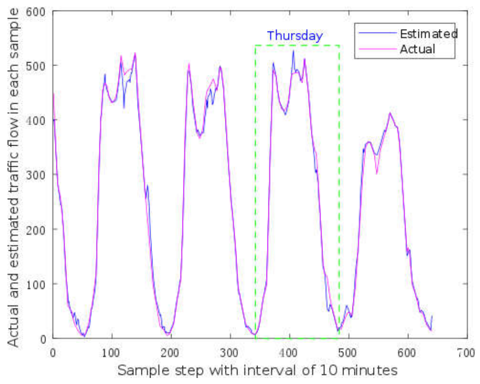

Figure 21: actual VS estimated traffic flow with n=24 for weekdays

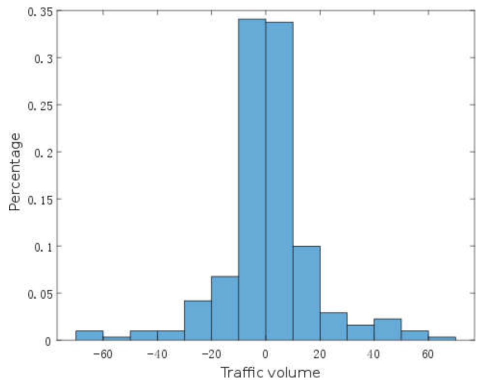

Figure 22: normalized estimation error with n=24 for weekdays

6. Conclusion and future works

In this paper, we propose a new approach with Gaussian Process Regression (GPR) machine learning for the estimation of traffic volume based on the Floating Car Data (FCD) from Google Maps. The travel time on road is the input for the estimation system. The GPR is applied to get the travel volume. The simulation results shows the efficiency of the proposed method. The performance with *n* (half width of sample window) from 4 to 24 (40 minutes to 240 minutes, because each sample interval is 10 minutes) are shown and compared.

From our experiments, the best result happens with n as 23, leading to a 15% rate for the maximum error. Then, the produced estimation seems to be precise enough to be used instead of static sensors based measurements. Practically speaking, this could have very interesting consequences: once the GPR model has been learned for a given road (on the basis of a relatively brief in site data acquisition process), the FCD are sufficient for continuously providing a good enough estimation of the vehicles flow along that road, using our GPR Regressor. Thus, as an alternative approach to investing in a costly static sensors system for each of the roads requiring a flow estimation, we could rather use dismountable measurements units to acquire the data for training the GPR, and then move this system to another place and repeat the process until each model of these "strategic" roads was learned. Finally, a GPR Regressor combined with its corresponding FCD stream would act as a kind of virtual sensor, cost effective vehicles flow sensor. This is exactly what we are currently developing in the scope of the ORIO project (refers to the Acknowledgment section).

An important point regarding the proposed approach is that the input vector of the GPR Regressor is a sequence of consecutive FCD values centered at the **sample step** for which the flow estimation has to be produced. To make it clear, the proposed GPR does not directly take into account the "**time**" (says hours, minutes and seconds), neither does it use the name of the day itself while it produces a rather good estimation of the flows. It is no secret telling that traffic flows evolve with the time of the day and the day of the week (the traffic during the week end has sometimes nothing to do with what it is on business days!). Then, instead of feeding our GPR learning process with data including a complete week, we plan to build sets of days related GPR Regressor in replacement of "weekly" GPR Regressor. This partition can also be made at an even lower level if we split the day related Regressor into sets of "period of the day" Regressor (for instance: night, morning, midday, afternoon and evening). This approach corresponds to a multi-melization technique inspired from Automatics: a complex nonlinear model of a system is replaced by a set of simpler models whose outputs are combined a way the resulting combination stay close enough the the system's actual output. This multi-modelization approach could be a valuable alternative to the use of a larger amount of FCD data we need to handle some of the local behaviors of the vehicles flow evolution: as we discussed in the section presenting our experimental results, when we use a weekly GPR Regressor, the value n of the FCD data half width window has to be enlarge to cope with the specific trend of the flow during the week-end. Using "daily" GPR Regressor or, at least, week-end specific Regressor may help keep n small while still producing a good enough flow estimation.

Another important aspect we did not address yet is about the prediction of the vehicles flows. As stated at the beginning of this paper, we want to produce a kind of virtual sensor that can replace actual in site measurements with a Regressor fed by aggregated FCD. We then propose to solve an **estimation** problem. One next interesting point is to check how the GPR Regressor can be used to predict (at a given time horizon) the traffic flows, what is part of our further works.

To be exhaustive enough, different other Machine Learning methods should be tested on the floating car data set (e.g. support vector machine, Regression Tree, linear Regressor, and Ensemble learning) so that their performances can be compared and explained. In relation with this last point, we also have to check the genericity of the robustness of the proposed Regressor be applying them to other roads and transportation networks. As we can see, these offer a very wide and interesting playground for Machine Learning related techniques with many possible applications.

Acknowledgment

This work was supported by the project ORIO (Observing the peRformances of urban Infrastructures and mobility / preventing collisions with vulnerable people using Opportunistic radar). The project ORIO is done within the framework of ELSAT2020, which is co-financed by the European Union with the European Regional Development Fund, the French state and the Hauts de France Region Council. The real traffic flow data from induction loops on the road 86 Boulevard de la République, Douai, France" was provided by the town hall of Douai (mairie de douai).

References

[1] S. Cheung, S. Coleri, B. Dundar, S. Ganesh, C.-W. Tan, P. Varaiya, Tracffic measurement and vehicle classification with single magnetic sensor, Transportation research record: journal of the transportation research board (1917) (2005) 173–181.

[2] B. Coifman, D. Beymer, P. McLauchlan, J. Malik, A real-time computer vision system for vehicle tracking and tracffic surveillance, Transportation Research Part C: Emerging Technologies 6 (4) (1998) 271 – 288.

[3] T. Miwa, Y. TAWADA, T. Yamamoto, T. Morikawa, En-route updating methodology oftravel time predictionusing accumulated probe-car data.

[4] S. Turksma, The various uses of floating car data, in: Tenth International Conference on Road Transport Information and Control, 2000. (Conf. Publ. No. 472), 2000, pp. 51–55.

[5] J. Yoon, B. Noble, M. Liu, Surface street traffic estimation, in: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services, MobiSys '07, ACM, New York, NY, USA, 2007, pp. 220–232.

[6] C. De Fabritiis, R. Ragona, G. Valenti, Traffic estimation and prediction based on real time floating car data, in: Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on, IEEE, 2008, pp. 197–203.

[7] C. Nanthawichit, T. Nakatsuji, H. Suzuki, Application of probe-vehicle data for real-time traffic-state estimation and short-term travel-time prediction on a freeway, Transportation Research Record: Journal of the Transportation Research Board (1855) (2003) 49–59.

[8] Y. Asakura, T. Kusakabe, N. X. Long, T. Ushiki, Incident detection methods using probe vehicles with on-board gps equipment, Transportation Research Procedia 6 (2015) 17–27.

[9] X. Dai, M. A. Ferman, R. P. Roesser, A simulation evaluation of a real-time traffic information system using probe vehicles, in: Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE, Vol. 1, IEEE, 2003, pp. 475–480

[10] J. Hong, X. Zhang, Z. Wei, L. Li, Y. Ren, Spatial and temporal analysis of probe vehicle-based sampling for real-time traffic information system, in: Intelligent Vehicles Symposium, 2007 IEEE, IEEE, 2007, pp. 1234–1239.

[11] C. de Fabritiis, R. Ragona, G. Valenti, Traffic estimation and prediction based on real time floating car data, in: 2008 11th International IEEE Conference on Intelligent Transportation Systems, 2008, pp. 197–203.

[12] B. S. Kerner, C. Demir, R. G. Herrtwich, S. L. Klenov, H. Rehborn, M. Aleksic, A. Haug, Traffic state detection with floating car data in road networks, in: Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., 2005, pp. 44–49.

[13] A. Sunderrajan, V. Viswanathan, W. Cai, A. Knoll, Traffic state estimation using floating car data, Procedia Computer Science 80 (2016) 2008–2018

[14] J. Hong, X. Zhang, Z. Wei, L. Li, Y. Ren, Spatial and temporal analysis of probe vehicle-based sampling for real-time traffic information system, in: 2007 IEEE Intelligent Vehicles Symposium, 2007, pp. 1234–1239.

[15] J. H.Wu, M. Florian, S. He, An algorithm for multi-class network equilibrium problem in pce of trucks: application to the scag travel demand model, Transportmetrica 2 (1) (2006) 1–9.

[16] W. H. Lam, H.-J. Huang, Calibration of the combined trip distribution and assignment model for multiple user classes, Transportation Research Part B: Methodological 26 (4) (1992) 289–305.

[17] C. E. Rasmussen, Gaussian processes in machine learning, in: Advanced lectures on machine learning, Springer, 2004, pp. 63–71.

[18] Y. Xie, K. Zhao, Y. Sun, D. Chen, Gaussian processes for short-term traffic volume forecasting, Transportation Research Record: Journal of the Transportation Research Board (2165) (2010) 69–78.

[19] J. Hu, J. Wang, Short-term wind speed prediction using empirical wavelet transform and gaussian process regression, Energy 93 (2015) 1456–1466.

[20] T. Id´e, S. Kato, Travel-time prediction using gaussian process regression: A trajectory-based approach, in: Proceedings of the 2009 SIAM International Conference on Data Mining, SIAM, 2009, pp. 1185–1196.

[21] J. Hu, X. Li, Y. Ou, Online gaussian process regression for time-varying manufacturing systems, in: Control Automation Robotics & Vision(ICARCV), 2014 13th International Conference on, IEEE, 2014, pp. 1118–1123.

[22] F. V. Jensen, An introduction to Bayesian networks, Vol. 210, UCL press London, 1996.

[23] D. J. MacKay, Gaussian processes-a replacement for supervised neural networks?, Citeseer, 1997.

[24] C. K. Williams, C. E. Rasmussen, Gaussian processes for machine learning, the MIT Press 2 (3) (2006) 4.

revision letter

review 1

This is a good paper in general since it provides a novel approach to measure the traffic flow on roadways without using traditional loop detectors or video cameras. The use of GPR is effective and makes the model straightforward to understand and apply. I just have one suggestion: In the future, maybe you could try to figure out an even easier method to predict traffic flows by simply accumulating and analyzing the number of FCD senders within a specific road segment in a given time period.

Reply: Thanks for your nice suggestion. Firstly, the Google map does not provides crude data to us, such as number of FCD senders and only offers travel duration within a specific road segment in a given time period. Furthermore number of FCD senders does not equal to number of vehicles, because onew vehicle can have several passengers, like bus and sedan car. Therefore, we can not measure the traffic flow by simply accumulating and analyzing the number of FCD senders within a specific road segment in a given time period.

review 2

This manuscript developed a method to estimate traffic flow parameters based on floating car data. The topic is new and results are interesting. However, it still needs further expoloring.

Reply: Thanks for your nice comment.