DEGREE PROJECT IN THE BUILT ENVIRONMENT,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2017

Spatio-temporal Traffic Flow Prediction

MESELE ATSBEHA GEBRESILASSIE

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ARCHITECTURE AND THE BUILT ENVIRONMENT

ROYAL INSTITUTE
OF TECHNOLOGY

Spatio-temporal Traffic Flow Prediction

Master's Degree Thesis

Mesele Atsbeha Gebresilassie
mageb@kth.se

Division of Geoinformatics
Department of Urban Planning and Environment
Schools of Architecture and the Build Environment
KTH - Royal Institute of Technology

Stockholm, 2017

Abstract

The advancement in computational intelligence and computational power and the explosion of traffic data continues to drive the development and use of Intelligent Transport System and smart mobility applications. As one of the fundamental components of Intelligent Transport Systems, traffic flow prediction research has been advancing from the classical statistical and time-series based techniques to data–driven methods mainly employing data mining and machine learning algorithms. However, significant number of traffic flow prediction studies have overlooked the impact of road network topology on traffic flow. Thus, the main objective of this research is to show that traffic flow prediction problems are not only affected by temporal trends of flow history, but also by road network topology by developing prediction methods in the *spatio-temporal*.

In this study, time–series operators and data mining techniques are used by defining five partially overlapping relative temporal offsets to capture temporal trends in sequences of non-overlapping history windows defined on stream of historical record of traffic flow data. To develop prediction models, two sets of modeling approaches based on *Linear Regression* and *Support Vector Machine for Regression* are proposed. In the modeling process, an orthogonal linear transformation of input data using Principal Component Analysis is employed to avoid any potential problem of multicollinearity and dimensionality curse. Moreover, to incorporate the impact of road network topology in the traffic flow of individual road segments, shortest path network–distance based distance decay function is used to compute weights of neighboring road segment based on the principle of *First Law of Geography*. Accordingly, (a) Linear Regression on Individual Sensors (LR-IS), (b) Joint Linear Regression on Set of Sensors (JLR), (c) Joint Linear Regression on Set of Sensors with PCA (JLR-PCA) and (d) Spatially Weighted Regression on Set of Sensors (SWR) models are proposed. To achieve robust non-linear learning, Support Vector Machine for Regression (SVMR) based models are also proposed. Thus, (a) SVMR for Individual Sensors (SVMR-IS), (b) Joint SVMR for Set of Sensors (JSVMR), (c) Joint SVMR for Set of Sensors with PCA (JSVMR-PCA) and (d) Spatially Weighted SVMR (SWSVMR) models are proposed. All the models are evaluated using the data sets from 2010 IEEE ICDM international contest acquired from Traffic Simulation Framework (TSF) developed based on the NagelSchreckenberg model.

Taking the competition's best solutions as a benchmark, even though different sets of validation data might have been used, based on k–fold cross validation method, with the exception of SVMR-IS, all the proposed models in this study provide higher prediction accuracy in terms of RMSE. The models that incorporated all neighboring sensors data into the learning process indicate the existence of potential interdependence among interconnected roads segments. The spatially weighted model in SVMR (SWSVMR) revealed that road network topology has clear impact on traffic flow shown by the varying and improved prediction accuracy of road segments that have more neighbors in a close proximity. However, the linear regression based models have shown slightly low coefficient of determination indicating to the use of non-linear learning methods. The results of this study also imply that the approaches adopted for feature construction in this study are effective, and the spatial weighting scheme designed is realistic. Hence, road network topology is an intrinsic characteristic of traffic flow so that prediction models should take it into consideration.

Key words: ITS, principal component analysis, spatio-temporal traffic flow, spatially weighted regression, traffic flow prediction, support vector machine for regression

Acknowledgments

First and foremost thanks to the almighty God. My appreciation also goes to my family who have been always supportive. Next, I would like to thank my master thesis adviser Györö Gidófalvi (PhD.). The door to Györö Gidófalvi's office was always open and his invaluable advise, support, patience, feedback, ideas, critiques and curious evaluations of progresses to this research development has been amazing. He consistently allowed this paper to be my own work, but steered me in the right direction whenever he thought I needed it. The Swedish Institute (SI) also takes my sincere gratitude for their financial support during my study through the SI Scholarship program. I would also like to thank all the KTH community and my friends for their support during my study.

List of Abbreviations

ATR	Automatic Traffic Recordings
ARMA	Autoregressive Moving Average
ARIMA	Autoregressive Integrated Moving Average
ATIS	Advanced Travelers Information System
ATMS	Advanced Traffic Management Systems
BFE	Backward Feature Elimination
DTA	Dynamic Traffic Assignment
ERM	Empirical Risk Minimization
FFC	Forward Feature Construction
GBM	Gradient Boosted Machines
GIS	Geographic Information System
GPS	Global Positioning System
GWR	Geographically Weighted Regression
iid	Independently Identically Distributed
ITS	Intelligent Transport System
ISAD	Iterative Single Data Algorithm
JLR	Joint Linear Regression for set of Sensors
JLR-PCA	Joint Linear Regression with Principal Component Analysis
JSVMR	Joint Support Vector Machine for Regression
JSVMR-PCA	Joint Support Vector Machine for Regression with Principal Component Analysis
KKT	Karush–Kuhn–Tucker
k-NN	K - Nearest Neighborhood
LR-IS	Linear Regression for Individual Sensors
LSSVM	Least Square Support Vector Machine
LSSVM - PSO	Least Square Support Vector Machine with Particles Swarm Optimization
MSRARMA	Multivariate Spatio-temporal Auto-Regressive Moving Average
OLS	Ordinary Least Square
PCA	Principal Component Analysis
RBF	Radial Basis Function
RMSE	Root Mean Square Error
RFID	Radio Frequency Identification
RBM	Restricted Boltzman Machine
SARIMA	Seasonal Autoregressive Integrated Moving Average
SMO	Sequential Minimal Optimization
STARIMA	Spatio–Temporal Auto–Regressive Integrated Moving Average
STRE	Spatio–Temporal Random Effect
SVM	Support Vector Machine
SVMR	Support Vector Machine for Regression
SWR	Spatially Weighted Regression
SVMR - IS	Support Vector Machine for Regression for Individual Sensors
SWSVMR	Spatially Weighted Support Vector Machine for Regression
VIPS	Video Image Processing Systems

Contents

List of Figures

List of Tables

1 Introduction

Traffic information such as flow, volume, speed, occupancy, travel time, density, vehicle classification, emission level etc. along road networks is important for planning, control and management of transport systems. Traffic information can be revealed using sensor technologies (in real time) or by computational estimation of historical traffic data (i.e. data-driven) or from the combination of both. Sensor technologies that measure traffic characteristics have been commonly used in major urban areas globally [29]. These technologies provide traffic information, such as traffic flow in real time. Traffic flow is defined as the number of vehicles passing through a specific point on a road segment in unit time, expressed in terms of vehicles per unit of time [61]. Traffic flow data is important to drive and forecast other relevant traffic characteristics along road networks. For example traffic flow estimation and forecasting aims at helping the understanding and development of optimal operation of road networks leading to efficient mobility.

Some of the traffic sensor technologies commonly used include Inductive–loop Detectors, Video Image Processing Systems (VIPS), Radio Frequency Identification (RFID) based systems, Pneumatic Tubes etc. [52]. According to the Federal Department of Transportation [25], using these technologies requires road pavement cut and they have high cost of installation and maintenance. Moreover, their operations face frequent disruption due to weather anomalies. Many of these technologies also lack comprehensiveness in terms of the traffic parameters they measure. Some of them are also pron to physical camera shake related problems.

Nevertheless, the long time use of these physical sensor technologies in major urban areas produced large volume of historical traffic data. Such historical traffic data continues to get larger which is then termed as *transportation big data* [32]. Coupled with the high cost of sensor technologies, advancement in computing and computational intelligence, as early as three decades ago, research interests in the area of traffic forecasting continued shifting towards data–driven approaches [41]. Thus, research on traffic prediction, modeling and algorithm development approaches continued to advance. Consequently, contemporary modeling of traffic flow prediction research expands from classical time series and univariate modeling techniques into data mining and machine learning algorithms in support of the development of advanced *Intelligent Transport Systems (ITS)* applications [26]. Traffic flow prediction is a salient feature of ITS.

1.1 Traffic flow prediction

Proactive traffic flow prediction is key to the development of ITS [56], which depends on the timely and accurate forecasting of the spread of traffic to support the control, management and improvement of traffic conditions. According to the IEEE Transaction on Intelligent Transportation Systems, ITS is defined as those systems utilizing synergistic technologies and system engineering concepts to develop and improve transportation systems. The EU Directive also defined ITS as systems in which information and communication technologies are applied in the field of transport, including infrastructures, vehicles and users, and in traffic and mobility management [11]. Traffic flow prediction is thus a key element of Advanced Travelers Information System (ATIS), Advanced Traffic Management System (ATMS) and Dynamic Traffic Assignment (DTA); these are in turn functional components of ITS.

The ability to accurately predict traffic flow on a specific road segment ahead of time has multifaceted advantages for individual travelers, traffic controllers, transport planners, managers, businesses and government agencies [1,56,67]. For example, to estimate congestion level which is a phenomenon where vehicles travel at slower speed due to demand for road space exceeding the capacity of the road [54]. Congestion is a common problem and it is increasingly inducing problems to the socio-economies and well being of mainly the urban ecosystem. Thus, effective traffic flow prediction can support better decision to reduced/alleviate congestion, reduce emission, improve traffic operations and management. Accurately predicting traffic situation on road segments ahead of time, and communicating it in an effective way and in a timely manner to travelers can influence to change the behavior of travelers. These behavioral changes such as route change, trip cancellation and modal change of travelers depending on the traffic situation of road network is critical for transportation management and optimization.

Generally, traffic flow prediction research aims at developing methodologies to support development of smart transportation systems and their management mainly for the rapidly urbanizing world where traffic related problems have severe consequences. Development of accurate and effective traffic flow prediction algorithms is thus, one of the main advancements in ITS research [42, 56]. In this regard, data–driven traffic flow prediction research has been getting momentum. Four important phenomena can be identified for the growing focus on developing data–driven traffic flow prediction algorithms to support the realization of ITS. These are: (1) high cost of construction, operation and maintenance of real–time physical traffic sensor technologies; (2) ineffectiveness of traffic sensor technologies due to environmental effects and their operational limitations; (3) the proliferation of large accumulation of traffic data and (4) the advancement in computational power and computational intelligence.

The first two phenomena can be considered as challenges in the implementation of physical traffic sensor technologies. The cost of implementing traffic sensor devices is not economically sound and feasible for many cities. Even if it could be possible, many of the sensor technologies have inherent limitations to address critical problems; such as resisting effects of weather anomalies. They also have limitations in their ability to measure as many traffic parameters as required by ITS application. Moreover, real–time sensors have limitations to provide accurate and timely short–term traffic information due to time required for data processing and communication. On the other hand, the proliferation of traffic data from various sensors cater for opportunities in advancing data–driven research horizon for the realization of ITS. The advance in computing power and computational intelligence to be able to effectively exploit historical traffic data also helps to develop robust prediction algorithms. As a result, data–driven short–term traffic prediction aims at increasing operational efficiency in traffic control and management.

Since the emergence of data–driven traffic flow prediction research, different studies have attempted to develop traffic flow models from different perspectives [8, 34, 38, 49]. However, prediction problems vary in their length of prediction horizon, types, size, frequency of data, etc. Moreover, a single model does not excel all other methods for all prediction scenarios; and a single method does not solve all kinds of prediction tasks [21, PP. 188]. Thus, traffic flow prediction is a problem specific task and there does not exist a universal model that fits all kinds of prediction problems.

From the *First Law of Geography* [53], which states *"all things are related, but near things are more related to each other than far things"*, it is clear that traffic flow at near by road segments affect each other's flow as they feed traffic to each other. In a similar manner, studies such as [1, 26, 57, 62, 64] have indicated that not only temporal relationship of data but also geographic proximity among road segments has various degrees of impact on traffic flow among connected road segments. The knowledge of *road network topology* and *temporal* distribution of traffic information is important for nearly all transportation planning and design strategies [19]. Therefore, while time–series approaches can cater for ways to model temporal dimensions, spatial auto–correlation based approaches such as *Geographically Weighted Regression (GWR)* methods can reveal the spatial dependency of traffic flow in a network of roads.

1.2 Background of the study

The task of traffic flow prediction is highly complex, and many physical traffic sensor technologies hardly address traffic flow prediction in an effective and accurate manner [25]. However, data–driven, spatial and temporal analytics, data mining techniques and machine learning algorithms could bear better prediction performance. In relation to this, in 2010, IEEE sponsored by TomTom, the world's leading provider of portable GPS, fleet management and navigation systems, location–based and mapping products had organized a global research contest. The competition was organized into three major areas; namely (1) Traffic congestion prediction *(Traffic)*, (2) Modeling process of traffic jam formation *(Jam)* and (3) Traffic reconstruction and prediction based on real–time information from individual drivers *(GPS)* [22]. In the competition, researchers were asked *to devise the best possible algorithm that tackles problems of traffic flow prediction, for the purpose of intelligent driver navigation and improved city planning based on simulated historical traffic information (i.e. Traffic)*. The main question of the research was to devise an algorithm for predicting Automatic Traffic Recordings (ATR) on ten selected bidirectional road segments in Warsaw City, Poland based on the synthetic data. For the research contest various solutions were developed. Now, the competition is concluded, but the challenges and the data are available for further research. This study, emanated from the contest, aims to develop an Automatic Traffic Recording system (i.e. traffic flow prediction model); which is the first task of the competition using a combination of spatial and temporal analytics, data mining techniques and machine learning algorithms in the *spatio–temporal domain*.

1.3 Problem Definition

Let the time domain be denoted by $\mathbb{T} \equiv \mathbb{N}_0$ and represent minutes. For simplicity, let the geographical road network that confines the movement of vehicles be modeled as a weighted directed graph $G = (V, E)$, where V is a set of vertices such that each vertex $v_i \in V$ is a point in the 2D space, i.e., $v_i \in \mathbb{R}^2$, and where E is a set of directed edges such that there is a directed edge e_{ij} from vertex v_i to vertex v_j if and only if a road connects the two vertices and vehicles can move from vertex v_i to v_j on this road. Furthermore, let a directed edge e_i be associated with the following three attributes: number of lanes nr_l, maximum speed limit max_s, and the edge length(distance) $dist\_ln$. Let also $S = \{s_1^{\rightarrow}, s_1^{\leftarrow}, \ldots, s_n^{\rightarrow}, s_n^{\leftarrow}\}$ be a set of sensors which are subset of the directed edges $E^S \subseteq E$ measure the flow of vehicles.

Let $Q_{S_i}^{\Delta t_{hist}}$ be *the whole historical vehicle count data* of all road segments and q_{s_i} be the *vehicle count of individual road segments* and $q_{s_i} \subseteq Q_{S_i}^{\Delta t_{hist}}$. In particular, without limitation, let $q_{s_k}^{\rightarrow}(t)$ and $q_{s_k}^{\leftarrow}(t)$ denote the flow (i.e. count) of vehicles that pass through the directed edge $e_k^{\rightarrow} = e_{ij}$ in the forward direction and the directed edge $e_k^{\leftarrow} = e_{ji}$ in the backward direction during the time period $[t, t+1)$.

Then flow prediction task, for a given *prediction time* t_p, *prediction horizon* t_{ph}, *prediction window* Δt_{pw}, and *history window* Δt_{hist} is to estimate for each sensor $s_i \in S$ the flow of the vehicles through s_i during time period $[t_p + t_{ph}, t_p + t_{ph} + \Delta t_{pw})$, $\widehat{q_{s_i}^{pw}}$, based on all the sensor readings during the time period $[t_p - \Delta t_{hist}, t_p)$ such that the sum of squared error of the flow estimates is minimized, i.e.:

$$\sum_{s_i \in S} \left[\widehat{q_{s_i}^{pw}} - \sum_{t=t_p}^{t_p+t_{ph}+\Delta t_{pw}} q_{s_i}(t) \right]^2$$

An alternative evaluation of a proposed solution is to compare its prediction performance relative to a baseline (*BL*) solution which has been provided as:

$$BL = [t_p + t_{ph}, t_p + t_{ph} + \Delta t_{pw}] = \left(\sum_{t=t_p-\Delta t_{pw}}^{t_p} q_{s_i}(t) \right)$$

where BL stands for *Baseline* which is the period $[t_p + t_{ph}, t_p + t_{ph} + \Delta t_{pw}]$ for which prediction was made and $q_{s_i}(t)$ is the total count recorded at the specified time from $t_p - \Delta t_{pw}$ up to t_p. An illustration of a pair of consecutive history windows with the t_p, Δt_{pw} and Δt_{hist} is shown on Figure 1.

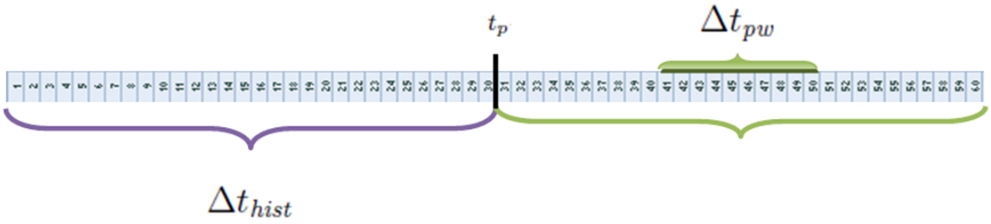


Figure 1: Pair of consecutive history windows in the time-series

1.4 General Objectives

The main objective of this study is to develop a prediction model that makes a short-term traffic flow prediction from historical traffic data using the concepts and methods in spatial and temporal analysis, data mining techniques and machine learning algorithms.

Specific objectives

1. To identifying and analyze relevant spatio-temporal analytics and data mining concepts and methods for short–term urban road traffic flow prediction.

2. To develop traffic flow prediction models based on existing spatial and temporal analysis, data mining techniques and machine learning concepts.

3. To evaluate the prediction models' performance.

1.5 Limitations and delimitation

In this research out of the potentially many ways one can follow in features construction, few statistical properties were identified based on the preliminary assessment of the historical traffic flow data. Therefore, there is no intentions to evaluate each potential data engineering technique. Moreover, it is assumed that the ten different simulation resulted in the training data are simply appended one after the other in the order they are given to us. The intention of this research is not to develop as many models as possible neither to propose and investigate internal optimization of each proposed models. It is delimited to develop models in such a way that both spatial and temporal dimensions of traffic flow are incorporated by designing appropriate input features from the historical traffic flow data.

1.6 Disposition

The rest of this paper is organized as Section 2 explores overall research in the area of traffic flow prediction spanning classical time series models to advanced machine learning algorithms including a brief assessment of the contest results and provides a glimpse of the technical details of the general modeling approaches employed in this study. Section 3 gives brief description of the methodology adopted. Section 4 presents the empirical evaluation and performance analysis and discussion of the models and finally 5 presents some concluding remarks as well as proposed future works.

2 Related Work

The literature on traffic flow prediction and modeling is concentrated on the classical time series modeling techniques. But recent development are shifting focus towards data–driven non-parametric and machine learning algorithms. As basic background for this study, a concise review of traffic flow forecasting studies is presented followed by the review of the top four approaches (solutions) selected in the 2010 IEEE ICDM international traffic prediction contest which is the basis for this study and some overview of the general modeling approaches adopted in this study.

2.1 Traffic flow modeling

Data–driven traffic flow prediction research has been going on for more than three decades. Different studies have approached the problem of traffic flow prediction from different dimensions. Traffic forecasting has been studied from time–series, pattern recognition, non-parametric regression, and a combination of of several of them [44]. The literature provides a wide range of methodological approaches for traffic flow forecasting heavily on the basis of classical time–series modeling techniques as being the foundation of time–series forecasting.

The classical and popular statistical modeling approach *Autoregressive Moving Average (ARMA)* and its wide range of extensions has been used as a baseline method for developing and evaluating other models for traffic flow forecasting [56]. ARMA is a generalized model of the *Box-Jenkins Autoregressive and Moving Average* models, which assumes that the time–series data is *stationary* (i.e. the constant nature of mean, variance and correlations over time). However,the major criticism toward using ARMA and its extensions is concerning their tendency to concentrate on the *mean values* and their inability to predict *extremes values* in time–series [58]. Moreover, the stationarity of time–series may not always truly exist.

On the other hand several studies have approached the problem of traffic flow forecasting using non–parametric modeling techniques. For example, non–parametric regression methods rely on data describing the relationship between dependent and independent variables deep rooted in pattern recognition [24]. As non–parametric models, advanced Neural Network and Bayesian Network modeling techniques have also been popular traffic flow forecasting methods [1,7,34]. Generally, non–parametric models are data–driven that imply, their successful implementation is related to the characteristics and quality of the available data. The two types of non–parametric techniques that have got significant popularity in short–term traffic forecasting research are non–parametric regression and neural networks. One of the main features of Neural Network models is their ability of *learning*, *memorizing* and *predicting*. These features are important for non–linear, uncertain and complex prediction problems.

Moreover, recent research developments such as in [32, 35, 43, 48] considered the problem of traffic flow prediction as a problem domain in the area of *transportation big-data* and approached it using machine learning algorithms. In a similar vein support vector regression and deep learning techniques, both supervised learning based techniques are becoming alternative tools to capture the complicated, non–linear and voluminous nature of traffic data [1, 4, 32]. Therefore, in general, traffic flow prediction literature is continuously shifting its focus onto data–driven approaches. While the classical statistical methods are still in use mostly as benchmarks for other models, non–parametric methods and machine learning algorithms are under intensive use in the area.

2.2 Comparisons of traffic flow models

Different studies have attempted to compare traffic flow modeling and forecasting approaches. The most common approaches used to compare forecasting models are *univariate and multivariate* modeling techniques [23] and *time–series (i.e. ARMA modelig family) and artificial intelligence* based modeling techniques [50]. Moreover, *parametric* and *non-parametric* nature of modeling techniques have also been commonly used [13, 49].

Beyond comparing group of methods based on individual modeling characteristics, several studies have also been trying to compare selected modeling approaches aiming at identifying a modeling approach or a model based on forecasting performance. However, the comparison of different forecasting methods come with their own pitfalls. Main problems associated with looking to find out the best performing method are, use of different operational setting of comparison, use of heterogeneous data, and linearity and non–linearity nature of the data sets and patterns in data etc. [56]. Thus, such comparison of models cannot help to explicitly identify a single modeling approach that can perform forecasting task best for all situation.

Some examples of such comparisons were the performance comparison of the classical statistical time–series ARIMA model, with artificial neural network and Non-parametric regression [44] which suggested that non-parametric regression significantly outperforms the other models. It was proposed to examine the classical parametric statistical model with time-series Seasonal Auroregressive Integrated Moving Average (SARIMA), and a non-parametric regression for an application to a single point short-term traffic flow forecasting [49]. This study aimed to examine prior claims about the superior performance of SARIMA model. An extensive experimental comparison of forecasting of two support vector machine models [30] also concluded that SARIMA model coupled with Kalman Filter as the most accurate model and support vector regressor as a highly competitive model for prediction of traffic flow during highly congested periods. The non-parametric model based Classification And Regression Trees (CART) model that works by classifying the historical traffic record and by applying the linear regression model to build corresponding traffic state pattern [66] and prediction through clustering shown that K-NN model and the Kalman filter parametric model as having better prediction accuracy. The non-parametric and data-driven methodology based on identifying similar traffic patterns using an enhanced K-NN algorithm, with weighted Euclidean distance and has more weights for recent measurements compared with SARIMA and Adaptive Kalman Filter models reported a better forecasting performance [18]. Moreover, the heuristic techniques used to determine the values of two parameters, Least Squares Support Vector Machine (LSSVM) with Fruit Fly Optimization Algorithm (FOA) claim a superior performance when compared with RBF neural network and LSSVM combined with particles Swarm Optimization Algorithms(LSSVM-PSO) was presented [9]. A combination of auto–regressive integrated moving average (ARIMA), Kalman Filter (KF) and Back Propagation Neutral Network (BPNN) and incorporated linearly into the Bayesian Combination Method (BCM) to take advantage of each of the models were applied [59]. The result of the BCM was reported as having achieved a better performance when compared with the traditional Bayesian Combination methods in terms of both prediction accuracy and stability. In Geberal, traffic flow prediction in relation to the challenges of big–data is to process the raw big–data into compact time–series in order to make them suiting for models of choices [35].

2.3 Spatio-temporal traffic flow prediction

An important feature in traffic flow prediction and the main interest of the current study is weather or not and how the effect of road network topology can be incorporated in modeling process. A study based on Kalma Filter algorithm aimed at short–term traffic flow prediction is considered as one of the earliest research works that attempted to take into account the effect of nearby links' traffic conditions to predict the flow in neighboring road links [41]. On the basis of the *First Law of Geography*, there has been also studies that have indicated the potentials of incorporating spatial dependency of road networks in modeling short–term traffic flow based on historical and real–time traffic data. Accordingly, studies such as [26] applied statistical models using time–series analysis and geometric correlation approaches which designed 3D heat map to describe traffic conditions between roads and the relationship between adjacent roads on spatio–temporal domain represented by cliques in MRF (Multiple Reference Frame). With the notion of big–data, data–driven traffic state identification and prediction, using spatial and temporal contexts of historical data combined with dynamic real–time traffic data aims to identify correlation between historical and real–time traffic for congestion prediction [33]. A combination of the Moving Average based time series and multivariate spatial–temporal autoregressive (MSTAR) using large traffic data set [38] is also one of the spatio–temporal models in the traditional time–series models. Consequently, Multivariate Spatio–temporal Autoregressive Moving Average(MSTARMA) model was developed aimed at speed and traffic flow prediction. Moreover, univariate historical average and ARIMA and two multivariate VARMA (i.e. vector autoregressive moving average) and STARIMA (spacetime ARIMA) models [23] are also examples in the spatio-temporal domain for traffic flow prediction using time–series methods. A comparison of the forecasting performance of these models was undertaken with data sets from loop detectors located in major arterial.

Another recent study developed a spatial–temporal Weighted K–Nearest Neighbor model on a Hadoop platform aiming at enhancing short-term traffic flow forecasting using a state vector proximity measure [65]. Furthermore, a spatial and temporal correlation together with big–data deep learning approach for traffic flow prediction [32] was develped and claimed that their approach is innovative for the fact that they have applied deep learning technique in traffic flow prediction for the fist time. A spatio-temporal based traffic flow prediction model for a freeway, [15] that showed incorporating road topology effect when compared with models such as ARIMA which does not consider the road topology effect has higher performance in prediction accuracy. They also compared with the linear regression model with only spatial contributions, and they claimed that the average prediction error of the proposed model performed better.

A model based on the K–Nearest Neighbor by formulating the weighted distance metric and state vector which incorporate both temporal and spatial information into their model [15] claimed provides better accuracy when compared with only temporal models (i.e. historical average and artificial neural network) models. Another novel Spatio-Temporal Random Effects (STRE) model has also reduced computational complexity due to mathematical dimension reduction [64]. It was claimed that the results shown the STRE model not only effectively predicts traffic flow but also outperforms the well–established models such as enhanced versions of ARMA and spatio–temporal ARMA, and artificial neural network models. To examine the spatio–temporal auto–correlation structure of road networks and to determine likely requirements for building a suitable space–time forecasting model and exploratory analysis in space–time auto–correlation through both global and local auto–correlation measures were also made [8]. It was found that instead of global, dynamic local structures are better for space–time modeling and forecasting.

2.4 ICDM 2010 data mining research contest review

As described in Section 1.2, in the 10^{th} IEEE International Conference on Data Mining (ICDM2010) held on $Dec.14-17, 2010$, in Sydney, Australia, three top winning solutions selected for the IEEE ICDM contest and presented their solutions. According to the descriptions, the first winning solution, used *Supervised SVD-like factorization (Singular Value Decomposition)* and *Restricted Boltzmann Machine(RBM)* modeling as well as a *Least Square Estimation* approach was used for parameter estimation. The second top solution applied a combination of *Random Forest* and *k–Nearest Neighbors* modeling approach. The third top solution adopted different re–sampling technique on the training data set and applied *12–Tree Random Forest* modeling approach.

Generally, the solutions provided to the traffic prediction problem during the contest did not investigate the impact of the spatial locations of sensors in the prediction. On the other hand, the First Law of Geography, and the literature on that basis have shown us that spatial and temporal correlation based methods could reveal traffic characteristics better in general and traffic flow prediction in particular. Moreover, traffic flow prediction problems depend on the types of data and on the temporal and spatial distribution of observations. Thus, traffic flow prediction is a problem specific research. Moreover, modeling techniques that take into account the effect of temporal dimension of data and road network topology on traffic flow have the potential to forecast more realistic traffic flow and could bear better accuracy. In addition, most literature are limited to expressways, day hours and many of the literature never considered the road network effect on the prediction. Limited amount of information exist in the literature that incorporate road network effect through upstream and downstream connections in models of traffic flow prediction. Another drawback of the literature is the temporal resolution of traffic data is in most cases is 5-min data [38], which in cases of short distance road segments may not represent the reality as shorter prediction horizons may give more realistic traffic situation.

The ever growing accumulation of traffic data and the growing demand for robust and accurate prediction models that can support ITS applications; the need for assimilating both spatial and temporal characteristics of traffic information in data–driven traffic flow modeling, as well as the development computational power and computational intelligence are some of the motivating factors behind this study. Hence, prior to the modeling process, a general overview of some of the related approaches are highlighted in the subsequent subsections.

2.5 Overview of related general modeling approaches

This study mainly employs linear regression and Support Vector Machine for Regression (SVMR) modeling to examine the fundamental relationship between historical traffic flow patterns to the future traffic situations. Thus, dimensionality reduction and data transformation, spatial dependency in traffic flow as an application of Geographically Weighted Regression as well as Support Vector Machine for Regression and linear regression are highlighted in this section.

2.5.1 Linear regression

Linear Regression model examines the relationship between dependent (response) and independent (predictor) variables. Linear regression generally is formulated as:

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon_i$$

where y_i is a response variable i.e. $y_i \in \mathbb{R}$; $x_i : i = 1, 2, \ldots, n$ is a series of independent variables or matrix columns in case of high dimensional data i.e. $x_i \in \mathbb{R}^n$; ϵ_i are independently and identically normally distributed (i.i.d) random error terms i.e. $\epsilon_i = \mathcal{N}(0, \sigma^2)$; and $\beta_i : i = 1, 2, \ldots, n$ are the model parameters with β_o being the model constant (i.e. intercept). The values of the parameters can be determined among others using *Ordinary Least Square (OLS)* estimator as in the following.

$$\hat{\beta}_i = (x_i^T x_i)^{-1} x_i^T y_i$$

One of the important characteristics of OLS is If $\beta_i = 0$ and the only independent variable is the intercept, then this is the same as regressing y on a *column of ones*, and hence $\beta = \bar{y}$ would mean the observations per se.

Linear regression comes with four fundamental assumptions [40]. These assumptions are 1) the relationship between the dependent and independent variables is linear and additive. By additive means that the effects of each independent variable on the values of the dependent variable is additive. 2) Errors are statistically independent such as there is no correlation between consecutive errors in cases of time series data. 3) Homoscedasticity is that the errors should have constant variance against time for time series, against predicted values and against independent values. 4) Errors are normally distributed. Thus, violations against these fundamental assumptions may result in inefficient, biased or misleading conclusions on the claim. Three common uses of linear regression are 1) Causal analysis; 2) forecasting an effect and 3) trend forecasting. The trend forecasting deals with predicting trends into the future and gets future values based on previous and/or current values.

2.5.2 Modeling impact of road network topology in traffic flow

The above notation of linear regression models and OLS parameter estimation on Subsection 2.5.1, does not take the geographic variability of observations into consideration. Thus, when applied onto location sensitive data, it only provides the same weight for all observations in a geographic region. On the contrary, variation of observations through space is evident [53]. Traffic flow at specific locations of urban road network can be affected by several factors. One of these factors is the road network topology; i.e. geographic proximity of road segments that feed traffic to each other in short time period. Thus, traffic flow at a road segment in a specific period of time would likely affect the flow at another segment. Therefore, in traffic flow modeling, the effects of road network topology should be an intrinsic component for a realistic prediction.

One of the approaches in modeling spatial dependency of traffic flow in a network of road segments is spatially weighted regression. Hence, in traffic flow prediction context, spatially weighted regression would imply exploring the interactions that would take place between road segments at certain distances (i.e. spatial dependency). Thus, spatially weighting techniques in regression can be utilized to model the varying relationships among sensors set up at different road segments in an urban road network. Application of geographically weighted regression techniques are useful to extend the traditional regression by allowing estimation of local rather than global parameters by running regression for each individual location [5, 6]. Thus, it helps in the assessment of the spatial heterogeneity in the estimation of relationships between independent and dependent variables of a regression model.

Spatially weighted regression in traffic flow prediction can be achieved by (1) explicit inclusion of a spatial independent variable in the regression model; or (2) using an internally estimated spatial parameter. By explicit inclusion of a spatial variable into a model, proximity of sensor location as in [46] or zonal areas as in [1]. To address problems of spatial non-stationarity of observations in space, Geographically Weighted Regression as proposed by [17], provides techniques to observe spatially varying observations through space–specific parameters estimation.

Geographically Weighted Regression
Geographically weighted regression model is given in the following form:

$$y_i = \beta_0(u_i, v_i) + \sum_{i=1}^{n} \beta_i(u_i, v_i)x_i + \epsilon_i$$

where y_i stands for the response variable; $x_i : i = 1, 2, \ldots, n$ stands for the series of independent variables, or matrix columns in case of a high dimensional data; $\beta_i(u_i, v_i)$ stands for the space-specific parameters of the independent variables measured at geographic coordinates of (u_i, v_i); and in a similar manner the ϵ_i is the regression error term. The parameters are determined using weighting schemes that take into account the geographic location of observations in the following form:

$$\beta(u_i, v_i) = (x^T w(u_i, v_i)x)^{-1} x^T w(u_i, v_i)y$$

$w(u_i, v_i)$ represents a matrix of geographic weights specific to each location (u_i, v_i) such that observations nearer to a location (u_i, v_i) are given higher weights than those farther away. According to [17], weighting scheme can use distance metric of any kind (i.e. Euclidean, Network, Manhattan etc.) distance as proximity measuring metrics. The two most common ways of computing weights between locations are in the form:

$$W_{ij} = exp\left[-\frac{1}{2}\left(\frac{d_{ij}}{h}\right)^2\right]; and; W_{ij} = exp\left[-\frac{d_{ij}^2}{h^2}\right]$$

Where w_{ij} is the weight matrix form; d_{ij} is the distance between locations $i\&j$; h is the bandwidth which defines the gradient of the kernel. Thus, the weights are produced from the distance between any two sensors. Optimal bandwidth (i.e. h) selection is a trade-off between the bias and variance; where too small bandwidth mean large variance and too large a bandwidth means large bias in the local estimation [14]. Two types of weighting mechanisms are common to compute the spatial weights in GWR, *Fixed kernel* and *Adaptive kernel*. Traffic sensor location at a single point in road segments are located on a zero-dimensional geometry (point feature). Thus, considering the fixed kernel and assuming the traffic flow readings through that point as constant throughout provides better interpretation; hence, fixed kernel.

2.5.3 Data dimenssionality reduction

Data-driven modeling processes require identifying relevant and uncorrelated model input feature, transformation of values, dimensionality reduction etc. especially when large input data sets exist. Data dimensionality reduction primarily concerns with the number of input features and potential collinearity between any pair of inputs features. Because, high dimensional data sets require large number of parameters to be estimated but not all features in a high dimensional data may be significantly relevant in the model. Dimensionality reduction speeds up models learning process, helps with prediction, classification or clustering accuracy, helps remove un-informative or disinformative feature, helps to match estimated coefficients with set of new input features etc. [45]. Dimensionality reduction also helps simplify problems, and optimize performance of algorithm [20]. Thus, dimensionality reduction in machine learning is a critical data pre-processing task that has to be investigated prior to modeling. Reducing collinearity between a pair of input features in a regression model also strengthens the model stability.

There exist several types of dimensionality reduction methods. The common ones include *missing value and low variance removal*. When large portion of input set contains missing values, it compromises model performance. Low variance variables also have little help in modeling because, variance indicates the measure of how much information variability exists within the values of an input feature. When variables assume constant value, the variance would be zero; thus, little ability to discriminate data. Therefore, low variance usually means no interesting patterns exist in the data. However, before removing low variance variables, conducting sensitivity analysis on the variables may provide insights on how it may impact the overall model performance. It should also be noted that *low variance* is a subjective term and a meaningful threshold should be defined based on the cotext. Another dimensionality reduction method is *correlated features removal* in that when variables are highly correlated, they contain similar information; and one can be derived from the other with a high level of accuracy [60]. Thus, they do not add much information to an existing pool of features and either variable should be dropped. Correlation of two variables can be identified from their linear correlation coefficient value. Another machine learning algorithm *Random Forest*, in addition to its capability for classification problems, is used as dimensionality reduction based on selecting a smaller subset of input features. *Backward Feature Elimination (BFE) and Forward Feature Construction (FFC)* are two techniques that use removing and adding variables one by one at each consecutive iteration respectively [28]. *Principal Component Analysis*, a statistical technique that uses orthogonal linear transformation of originally $n-$ dimensional data set into *smaller and uncorrelated* set of features keeping information lose minimal is one of the effective ways in dealing with reducing dimensions and in avoiding or reducing collieanrity [63].

Principal Component Analysis

Principal Component Analysis (PCA) is a statistical technique with applications in explanatory and predictive analytic modeling, pattern recognition, image compression, dimensionality reduction etc. The primary purpose of applying PCA as dimensionality reduction technique is to transform n features into a newly m *uncorrelated* input features such that m is less than n while keeping information loss minimal. The transformed values are linearly uncorrelated variables called *Principal Components* put in descending order of *component variance*. Specifically, PCA transforms $Y = XW$ by mapping X_i from an n-by-p variables into n-by-q variables in a descending order of components variance. By taking the first l dimension from q, the transformation results in the form $Y_l = W_l X$ matrix and W forms an orthogonal base for the l features. In the cases of the original vector, and the transformed vectors, the size of the examples (i.e. n) remains the same. The general notation of PCA is; given data points $(x_1, x_2, x_3, \ldots, x_n) \in \mathbb{R}^p$ construct the data transformation in $\mathbb{R}^p \rightarrow \mathbb{R}^q$; and this results in a reduced and uncorrelated l number of input features $(x_1, x_2, \ldots, x_l) \in \mathbb{R}^q$. Transforming data using PCA requires data to be standardized. When different units of measures exist in the variables, scaling the data may also be required. While applying PCA two closely related fundamental characteristics of matrix algebra need to be computed from the co-variance matrix. These are *eigenvectors* and *eigenvalues*. Eigenvectors are orthogonal to each other. Eigenvectors and eigenvalues comes together and eigenvalues use to rank eigenvectors. The eigenvector with the highest eigenvalue is the first principal component and it shows the most significant relationship in that direction [39]. Hence, PCA transformed data is one that is expressed in terms of the patterns found in the variables these patters are drawn from the covariance matrix. PCA's fundamental assumption is that the variables in the transformed matrix are *as uncorrelated as possible*; thus, their co-variance is close to zero.

2.5.4 Support Vector Machine for Regression (SVMR)

Support Vector Machine for Regression (SVMR) which is also known as Support Vector Regression, is an emerging popular Support Vector Machine variant used for regression problems. While Support Vector Machine is popular in *classification* problems, SVMR is trained to produce numeric values, thus for regression. The general formulation of both SVM and SVMR is very similar. In both SVMR and SVM, the basic idea is mapping data set X into a high dimensional feature space F via a mapping function called *kernel function* ϕ and to do a linear regression in F [37]. SVMR is essential in solving problems that require large number of parameter estimation using the classical statistical methods. The general formulation of SVMR is:

$$f(x) = y = (w \cdot \phi(x)) + b$$

$$\phi : \mathbb{R}^n \rightarrow F; w \in F$$

Where b is the *bias* that controls the displacement and w is the *norm* that controls the direction of the vector. Thus, linear regression in a higher dimensional space $\mathbb{R}^n$ corresponds to non-linear regression in low dimensional input space. The mapping from low dimensional non-linear problem to a high dimensional linear problem solving in both SVM and SVMR is achieved through kernel trick. Kernels are intrinsic components of applications of SVM and SVMR. Hence, kernel function ϕ helps the SVM to transform a function as:

$$\phi : \mathbb{R}^m \rightarrow \mathbb{R}$$

Through the theoretical ability of kernels to work in unlimited dimensional space, kernel functions are most effectively used when they replace an *inner product* function using a linear or non-linear kernel. SVM works with the *Empirical Risk minimization* [55] function using the ϵ – *insensitive* loss function.

According to Smola, A.J. and Scholkopf, B. [51], the algorithms used to train SVM is *convex/Quadratic programming* and SVM is firmly grounded in the framework of statistical learning theory. This type of learning enables them to be generalized well for unseen data points, hence for prediction. The parameters for SVMR are the regularization parameter C, the ϵ; and additional parameters of the kernel used chosen. The default kernel for SVM is the Radial Basis Function (i.e. Gaussian kernel). When one selects the Gaussian kernel, the parameter to be estimated is σ. However, there are other kernel types one can choose from. Kernels are functions also known as *similarity functions*, that transforms data into a higher dimensional feature space to make it possible to perform linear regression. Kernel makes transformation calculations faster and easier, especially when features vectors of high dimension exists. With SVMR, the kernel replaces the dot product $\phi(x_i).\phi(x)$ by a kernel $k(x_i, x')$ as in the following:

$$y = f(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b$$

There are several types of kernel functions one can choose from. The most commonly used kernels are **Linear kernel** $k(x, x') = x^T x'$, the **Polynomial kernel** given in the form $k(x, x') = (1 + x^T x')^d$; where any $d > 0$, which refers to the degree of polynomial and **Gaussian Radial Basis Function (RBF)** $k(x_i, x) = exp\left[\frac{-||x-x'||^2}{2\sigma^2}\right]$ for infinite dimensional space where $\sigma > 0$ [47]. However, understanding the data set and the underlying patterns in the data are important to choose an appropriate kernel function for a machine learning algorithm under consideration.

3 Methodology

Basic statistical data exploratory and data pre-processing techniques are conducted prior to modeling in order to examine if past traffic information can indicate future traffic situations and to be able to extract features. *testing data* are given separately, whereas data splitting techniques are employed to prepare *training* and *validation set* so that models are trained and evaluated on different data sets. Furthermore, input features are constructed using common statistical properties based on partially overlapping five different relative temporal offsets defined to capture the temporal trend of a *sequence of non-overlapping history windows* on streams of historical traffic flow data on each sensor. Following the feature construction, two sets of modeling approaches based on *(1) linear regression* and *(2) support vector machine for regression* are proposed. Based on the linear regression modeling approach, (a) Linear Regression for Individual Sensors **(LR - IS)**; (b) Joint Linear Regression for Set of Sensors **(JLR)**; (c) Joint Linear Regression for Set of Sensors with PCA **(JLR - PCA)**; (d) Spatially Weighted Regression for Individual Sensors **(SWR)** are designed. Based on the support vector machine for regression are, (a)Support Vector Machine for Regression for Individual Sensors *(SVMR-IS)*; (b) Support Vector Machine for Regression for Set of Sensors *(JSVMR)*; (c) Support Vector Machine for Regression with PCA *(JSVMR-PCA)* and (d) Spatially Weighted Support Vector Machine for Regression *(SWSVMR)* models are designed. While the *Ordinary Least Square* is used as learning method in the linear regression methods, the *convex/programming* is utilized as a learning method in the Support Vector Machine for Regression based models. Both in the linear regression and in the support vector machine for regression based models, a distance decay function based on shortest path network distance among each of the sensors is implemented to investigate the impact of road network topology on traffic flow. In each set of models, linear transformation of data is applied using principal component analysis aiming at reducing input feature dimensions and the impact of multicollinearity on the models. Features constructed as predictors and response variables are based on individual road segment level and on a combined contribution of set of road segments. The linear regression based models are evaluated against the fundamental assumptions of OLS learning. A general illustration of the research methodology adopted is indicated on Figure 2.

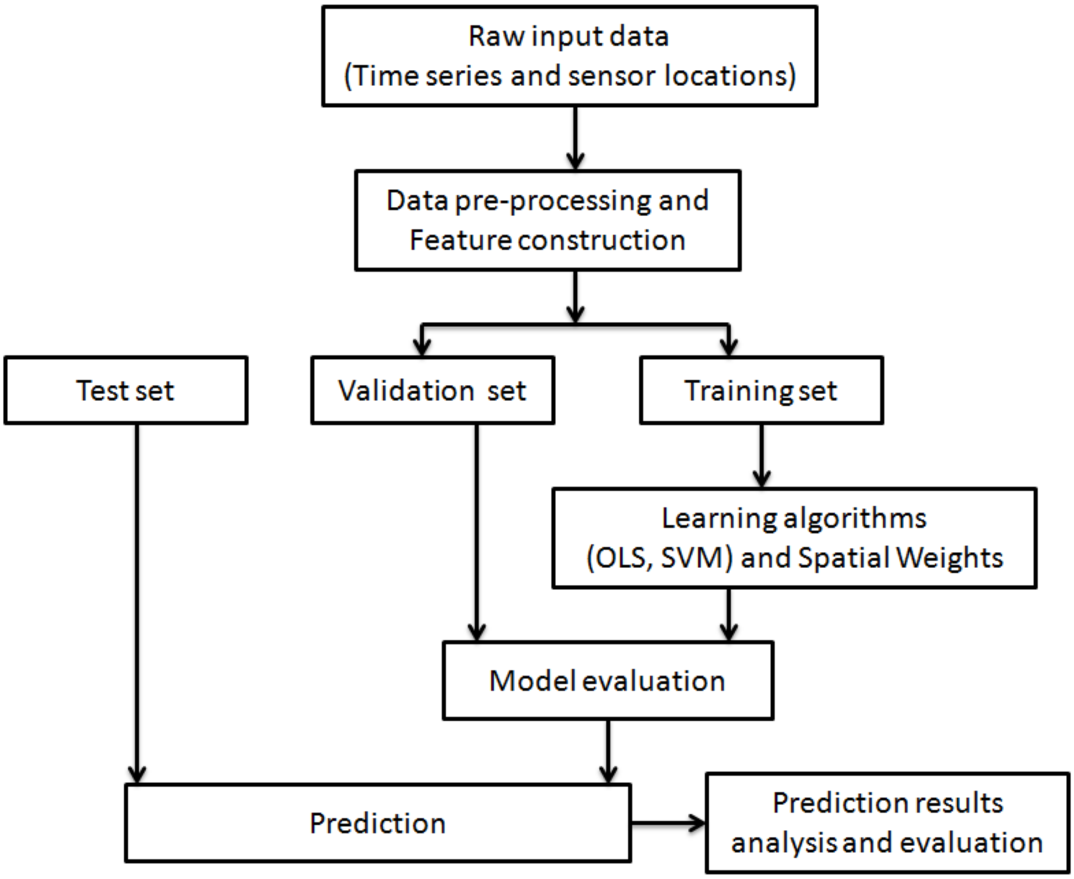


Figure 2: Research methodology adopted

3.1 Data pre-processing

Prior to features extraction visual examination on some statistical plots (i.e. scatter plotting, auto-correlation, cross correlation and box plots) are conducted to give a glimpse of the basic characteristics of the data. *Box plot* indicates the variability of data outside of lower and upper limits of the quartiles without assuming any underlying statistical distribution [36]. Figure 3 illustrates the box plot of each sensor's historical data. In this study outliers are ignored because, those larger values (more than the third quartile) are assumed as flows during certain period of rush hours not explicit outliers. Higher flows in all sensors can also be assumed as indications of the daily variability of traffic. Moreover, records of all sensors do not have any missing or negative values; zero value is a valid traffic count. Therefore, the box plot illustration of the streams of data of all sensors demonstrates that each sensor has reasonably proportional data records. That is there are no extremely high or extremely low values even though the distribution shows that certain records are slightly higher than the median and the standard deviation.

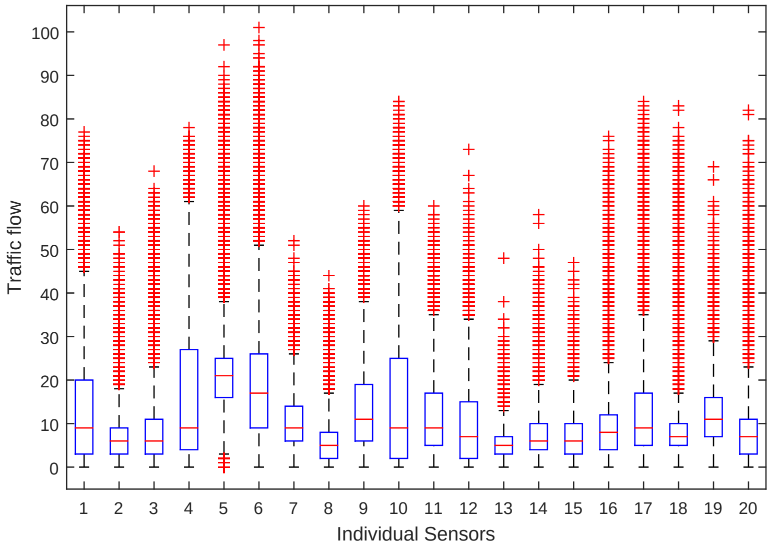


Figure 3: Box plot: traffic flow per minute of each sensor

To examine whether future traffic flow depend on past traffic situations, the historical traffic flow data is examined using auto-correlation and cross correlation techniques against its lagged values. In time series data the farther the history data is the less similarity exists with the future values and the more recent the history data is the more similar patterns exists [2]. Auto-correlation reveals linear dependence of a variable with itself that depends only on the time lag; i.e. between x and $x_{t-1}, x_{t-2}, ...$ for a time series values x and time scales $t = 1, 2, ...n$. Closely related to auto-correlation, *partial auto-correlation* gives the partial correlation of times series with its own lagged values [16]. Example of how lagged values of same sensor data is correlated, is shown on Figure 4.

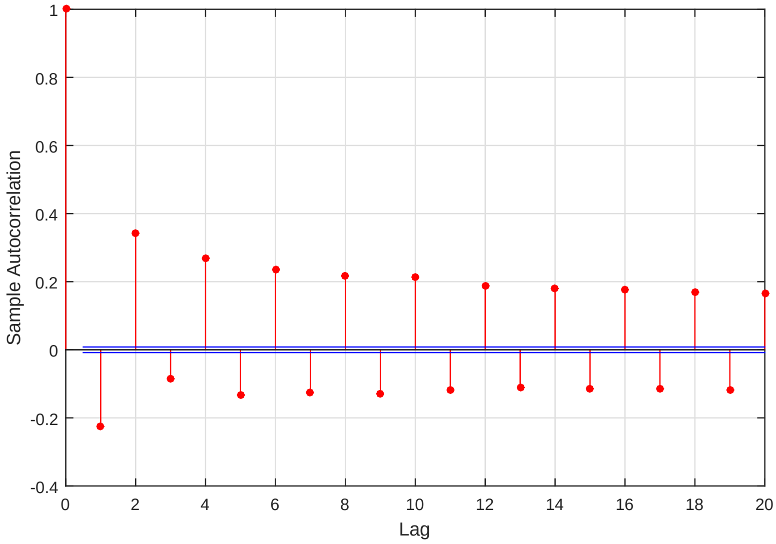


Figure 4: Partial auto-correlation: sensor - 13

Another statistical tool which describes the correlation between different sets of data (in this case across set of sensors) is *Scatter plot*. Scatter plots depict data from two variables and describe their *correlation* but not *causation*. In the case of Figure 4, it is well demonstrated that as an example for sensor - 13, that there exist significant correlation between consecutive values of for up to 20 lags (in minutes). Hence, this indicates, future values depend previous records to a certain extent. In addition, The scatter plot of the pairwise sensor data is examined mainly to help understand how data streams agree to each other in each sensors. The scatter plot of the training data set has shown no systematic patterns among data points of each sensor.

3.2 Data extraction and features construction

A relationship between sequence of past traffic flow patterns may indicate that future traffic situations can be derived from these historical records. In time series data, the situation is common and data engineering techniques rooted in time series operations as in [12] are employed to extract temporal trends in the data. To extract any temporal patterns that may exist in times series data, looking at smaller section of the data is crucial. But how small or how large section of time series data may contain specific patterns depends on the phenomena. One of the approach is using a non-overlapping fixed size history window that scans the entire series dividing the data into a sequence of $N = \{(x_1, t_1), (x_2, t_2), .\therefore, (x_n, t_n)\}$; where each (x_i, t_i) is consecutive fixed length sequence of smaller time series values and $t = 1, 2, ..., n$ are times sequences [31]. The values of the smaller sequences of series can be expressed using some derived values such as using statistical measures through which any trends and patterns would be sought. Standard time–series modeling techniques also use statistical properties together with time series operators such as *leads, lags and differencing* [49, PP. 70] for feature construction. Hence, important statistical properties such as *measure of central tendency*, *measures of statistical variation*, *inter-quartile range*, *measures of extreme values* etc. can be used to represent the values in each of the smaller series [2].

Accordingly, in this study, the above principles are applied in such a way that for each stream of historical traffic flow time series data, fixed size of 30 minutes long non-overlapping history window is defined. From each pair of consecutive history windows, a pair of (x_i, y_i) values are constructed in which while the leading makes up the x_i (i.e. independent variable), the succeeding makes up y_i (dependent variable)values. The sequence of x_i values are constructed based on a set of *smaller partially overlapping relative temporal offsets* that are defined on each leading history window on a *5, 10* and *30* minutes *temporal granularity*. Each of the relative temporal offset are represented by three selected statistical measures namely statistical mean (i.e. **mean**), rate of change (**roc**) and standard deviation (**std**) of the traffic flow in each temporal offset.

Considering the prediction time t_p, the temporal offsets are the most recent 5 minutes $(t_p - 5, t_p]$, the most recent 10 minutes $(t_p - 10, t_p]$, the middle 10 minutes $(t_p - 20, t_p - 10]$, the farthest 10 minutes $(t_p - 30, t_p - 20]$ and the whole history window $[t_p - 29, t_p]$; hence, a matrix of (x_i) where $i = 1, 2, ..., n$, is constructed as feature sets from each stream of time series data that comes from each sensor as illustrated on Figure 5.

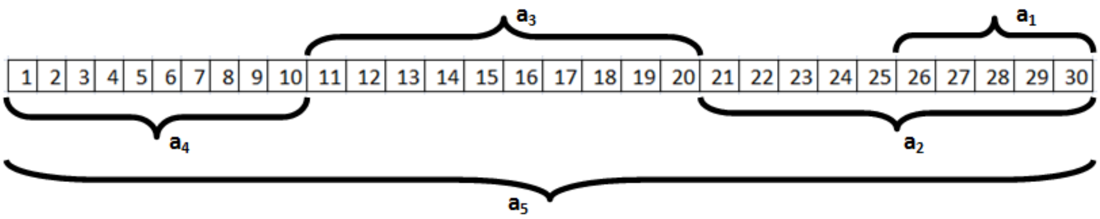


Figure 5: Partially overlapping temporal offsets in each preceding history window

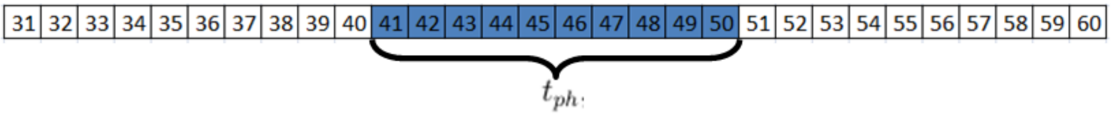


Figure 6: Prediction horizon in each succeeding window

Another vector of y_i, is defined as the summation of traffic flow values in each prediction horizon as illustrated on Figure 6; i.e. $\sum_{i=k}^{k+t_{\Delta ph}}(q_i)$ of each succeeding window. Based on the five temporal offsets and the three statistical measures, a total of 15 predictors (i.e. x_i) are constructed for each sensor's stream of data. Therefore, prediction models depending on such in put of large amount of data would likely give a potentially better forecasting capability. For easier notation, let's assume that $a_1 = [t_p - 5, t_p]$;

$a_2 = [t_p - 10, t_p]$, $a_3 = [t_p - 20, t_p - 10]$; $a_4 = [t_p - 30, t_p - 20]$ and $a_5 = (t_p - 30, t_p]$ be the most recent five minutes, the most recent ten minutes, the next most recent ten minutes, the farthest ten minutes and the whole history window (i.e. 30 minutes) respectively. Thus, for each sensor, the following feature vectors can be extracted from each stream of time series data.

$$x_i = \{mean(a_1), mean(a_2), ..., mean(a_5), roc(a_1), roc(a_2), ...roc(a_5), std(a_1), std(a_2), ..., std(a_5)\}$$

Based on the above notation, for each sensor, a total of *3 statistical properties* by *5 relative temporal offsets* produces *15 input features* and a response variable for 1000 hours that represent 1000 examples. These features are then used for training and validating to the *OLS* and *SVM* based learning algorithms in the proposed prediction model. As the historical traffic flow time series dataset is large in volume, *k–fold cross validation* technique where $k = 10$ is used to split into training and validation sets as in Kohavi, Ron. et.al. [27].

3.3 Main modeling techniques

The modeling techniques proposed are based on *OLS* for the linear regression and *SVM* for the support vector machine for regression models. The spatial weighting technique uses a *distance decay* function using *shortest path network distance* according to each sensor's *geographic proximity to the location of prediction*. The spatial weights are applied in both the *OLS* and *SVM* based algorithms.

The model input features constructed following the procedures described in Subsection 3.2 are organized in a systematic way to examine if (1) stream of data that come from individual sensors can help us predict feature traffic situation on same road segment; (2) stream of data that come from multiple neighboring sensors can help us predict with better accuracy to the future in a given sensor; (3) data streams that come from multiple neighboring sensors are weighted based on their geographic proximity and if they cater for better accuracy and more realistic prediction, so that if such kind of operational set of the models would reveal the impact of road network topology in traffic flow; and (4) if advanced and robust learning algorithms such as SVMRR would capture the patterns and trends of traffic flow so that more accurate predictions would be possible. Description of individual model design, assumptions made and the organization of input data is given in the subsequent Subsections.

3.3.1 Multilinear regression models

Based on the fundamental principles of OLS estimation, four slightly different models are designed. These models take into account only the individual sensor's historical data, historical traffic flow data combined from multiple neighboring sensors, as well as by introducing linear data transformation and spatially weighted inputs. Each of the models design is given as in the following.

Linear Regression for Individual Sensors (LR-IS)

LR-IS model meant to make traffic flow prediction based on a stream of time series data that come from a single sensor. In this model design, no neighboring road segments that might contribute traffic are considered. This model design is based on linear regression and is defined as:

$$y_i = \beta_o + \sum_{k=1}^{m} \beta_k x_{ik} + \epsilon_i$$

where $y_i, i = 1, 2, .\because, n$ is a vector of response values with n examples; β_o is a regression constant term (intercept); $\beta_k : k = 1, 2, .\because, m$ are set of m regression parameters estimated based on the OLS estimator; $x_{ik} : i = 1, 2, .\because, n$ is set of m variables with n number of examples. This model is designed in such a way that each sensor's data is used to train the model, to validate it and make the predictions using the test data of the same sensor.

Joint Linear Regression for Set of Sensors (JLR)

This model is designed aiming at improving the performance of the LR-IS model by adding more input features from other neighboring road segments, hence *Joint Regression*. JLR model is also design to investigate if there exist some kind of dependency among each of the road segments being studied without explicit inclusion of geographic proximity in the model. Thus, the JLR model is design as:

$$y_i = \beta_o + \sum_{k=1}^{l} \beta_k x_{ik} + \epsilon_i$$

where $\beta_k : k = 1, 2, .\because, l$ are l number of regression parameters determined by the OLS estimator; $x_{ik} : i = 1, 2, ...n$ is a matrix of n examples and l variables (features) constructed from the streams of data from all the sensors. JLR is deigned as a high dimensional model. Based on the current design, as the number of neighboring station increase on an interconnected network of roads, the dimension of l input features also increases.

Linear Regression for Set of Sensors with PCA (JLR-PCA)

Similar to JLR, JLR-PCA basically implements linear regression on input features constructed from all neighboring sensors. But JLR-PCA implements Principal Component Analysis with *Singular Value Decomposition* algorithm as a means of enhancing JLR in terms of the so called dimensionality curse and to reduce any potential collinearity problem among input features. Thus, the number of parameters to be estimated is smaller but with minimal information loss. *JLR-PCA* is designed by using p features sets after PCA is applied on l; where $p < l$. Principal component analysis is applied on the set of $xik : k = 1, 2, .\therefore, l$ variables, $x_{ik} : k = 1, 2, .\therefore, p$ number of top principal components can be selected for a model input.

$$x_{ik} \in \mathbb{R}^l \rightarrow \mathbb{R}^p$$

$$y_i = \beta'_o + \sum_{k'=1}^{p} \beta'_k x'_{ik'} + \epsilon_i$$

Therefore, when the *JLR* and *JLR-PCA* are compared, the latter is trained on p number of principal components unlike to the *JLR* model which is trained on m number of variables while the information loss is kept minimal. There is no clear cut as to what size should p be. However, by examining the information lose and the desired model performance, any size of p can be selected.

Spatially Weighted - Regression (SWR)

The *JLR* and *JLR - PCA* models above are meant to examine if there exist any interdependence among the road segments in traffic flow. In these models, any potential interdependence is not modeled based on the location of the road segments (i.e. sensors). However, spatially weighting the input features constructed from streams of data from all neighboring sensors based on their geographic proximity to the location of prediction is expected to appropriately reveal the interdependence. Spatially weighted regression model is designed to achieve this and aims to produce more realistic prediction on individual sensors. The important part of this model is dealing with the geographic distance metric and finding meaningful weights to variables and their values constructed from different geographic locations. The importance of giving more weight to some input features is to let them influence more on the parameters estimation, and by giving less weight to input features gives them less power to influence the parameters estimation [12]. Thus, a distance decay function is used to compute the weights given to each set of input features based on a shortest path network distance using the following function as implemented in ArcGIS.

$$w_{ij} = exp\bigg(- d_{ij}^2 / h^2 \bigg)$$

where w_{ij} weight of the variable constructed from sensor j while making prediction on sensor i. The two features in spatially weighted regression (i.e. distance and bandwidth) play critical role in revealing the semantics of *First Law of Geography*. In spatially weighted regression distance metric can be *Euclidean Distance*, which is a default distance metric for many GIS applications; or can be *Network distance* which gives better estimation of landscape heterogeneity especially with applications related to road transport. In this study, as vehicles only move along a predefined road networks, the network distance could provide meaningful weights than the Euclidean distance does. The variables constructed from the same sensor do have same weight, because all variables of same sensors are measured on same *geographic location*. Then the model is design to be trained on the weighted features constructed from each of the neighboring sensors. Therefore, the spatially weighted linear regression models is designed based on the network distance as in below.

$$y_i = \bar{\beta}_o + \sum_{k=1}^{l} \bar{\beta}_k X_{ik} + \epsilon_i$$

where: $X_{ik} = w_{ij} * x_{ik}$ which stands for the total number of features constructed from all sensors' features x_{ik}, weighted by the w_{ij} weights of each connected sensors.

3.3.2 Support vector machine for regressions models

Support Vector Machine for Regression is proposed aiming at achieving higher prediction accuracy with non–linear SVM learning algorithm. Similar to the OLS estimation based on linear regression, SVMR based models are designed and executed in four slightly different operational set ups. These SVMR based models are: (a) SVMR on individual sensors (SVMR-IS), which considers input features constructed from individual sensor's historical data to make prediction to the future on same sensor. (b) Joint SVMR (JSVMR), which combines input features from all other neighboring sensors to make prediction to the future on a specific sensor. In this model no means of weighting or geographic proximity of sensors is considered. (c) Joint SVMR using PCA (JSVMR - PCA), in which input features are constructed from all neighboring sensors and PCA, as a means of dimensionality reduction is applied before model training. (d) Spatially Weighted SVMR (SWSVMR), in which input features constructed from all neighboring sensors are weighted based on a distance decay function using the shortest path network distance from each individual sensor to the location of prediction. Thus, using spatial weights, features from near by road segments (i.e. sensors) would have higher influence on the prediction than those farther away. The SVMR model is designed as in below:

$$y_i = \sum_{i}^{n} (\alpha_i - \alpha_i^*).K(x_i, x) + b$$

where $y_i : i = 1, 2, .\because, n$ are n observed response values; $x_i : i = 1, 2, .\because, n$ are n training examples coming from all neighboring sensors; b is a model *bias*; $k(x_i, x)$ is a transformation function to be chosen from *linear*, *Gaussian* or *polynomial* kernel function described in Subsection 2.5.4. Further, Smola, A. et.atl., [51] provides a detailed derivation of Support Vector Machine for Regression.

However, the input (x_i, y_i) values are first weighted with the spatial weights similar to the spatial weights applied on the linear regression based models as in Subsection 3.3.1. Hence, spatially weighting and linear transformation of input features is accomplished as in below for the SVMR based models too.

$$x_{ik} = w_{ij} * X_{ij}$$

where X_{ij} are input values of the prediction location i and constructed from each neighboring sensors j; whereas the w_ij is spatial weights applied to values from sensor j.

$$X_{ij} \in \mathbb{R}^m \to \mathbb{R}^p : and; x_{ik} \in \mathbb{R}^p$$

is produced as a training set for the SVM algorithm with a specific kernel function.

4 Empirical Evaluation

Recalling back to the main task of this research as explicitly explained in the problem definition in Subsection 1.3 is, to predict to the future, the vehicle count of individual sensors based on historical records of set of sensors in the spatio-temporal domain. Consequently, data pre-processing techniques are employed mainly to extract temporal trends that can show future traffic situation and to construct features sets (random variables) using time series operators and data mining techniques. Accordingly, two different modeling approaches are proposed, designed and implemented. These modeling approaches are based on linear regression and Support Vector Machine for Regression. Both the sets of model are implemented based on input features constructed from individual sensors (i.e. road segments) traffic flow data and from a combination of neighboring sensors. The proposed models' results empirical evaluation and discussion are provided in the subsequent subsections.

4.1 Experimental set up

The proposed models are executed based on the MathWorks® MATLAB software standard implementation. The linear regression models are implemented in the MATLAB version *2015a* but the support vector machine for regression is only supported in MATLAB *2016* and latter versions. The learning algorithms used in the linear regression models is **Ordinary Least Square (OLS)**, and **Support Vector Machine** for the SVMR based models. Hence, the implementation is based on the **fitlm** and **fitrsvm** for the OLS and SVM respectively. All implementation codes used in this research are listed on the Appendix.

4.2 Data description

To evaluate each of the proposed models, the simulated traffic flow data set from the 2010 ICDM international research contest was used [22]. Training data set used contains $N = 60,000$ traffic flow records with a temporal resolution of one minute at 10 selected bi-directional road segments ($L = 20$) of Warsaw City, Poland a part of road network with $|V| = 18,716$ nodes and $|E| = 35,169$ edges. The training data set is collected over a total of 1000 hours of simulation, divided into a hundred of 10 hour long cycles. Test data sets are also provided covering different 1000 hours, split into 60 minute long window. From each 60 minutes long record only the first 30 minutes are available for this research, whereas the other 30 minutes of each hour were left secret for use in evaluation of the prediction task during the competition and never released thereafter. Thus, test data contains 30 lines of record for every window.

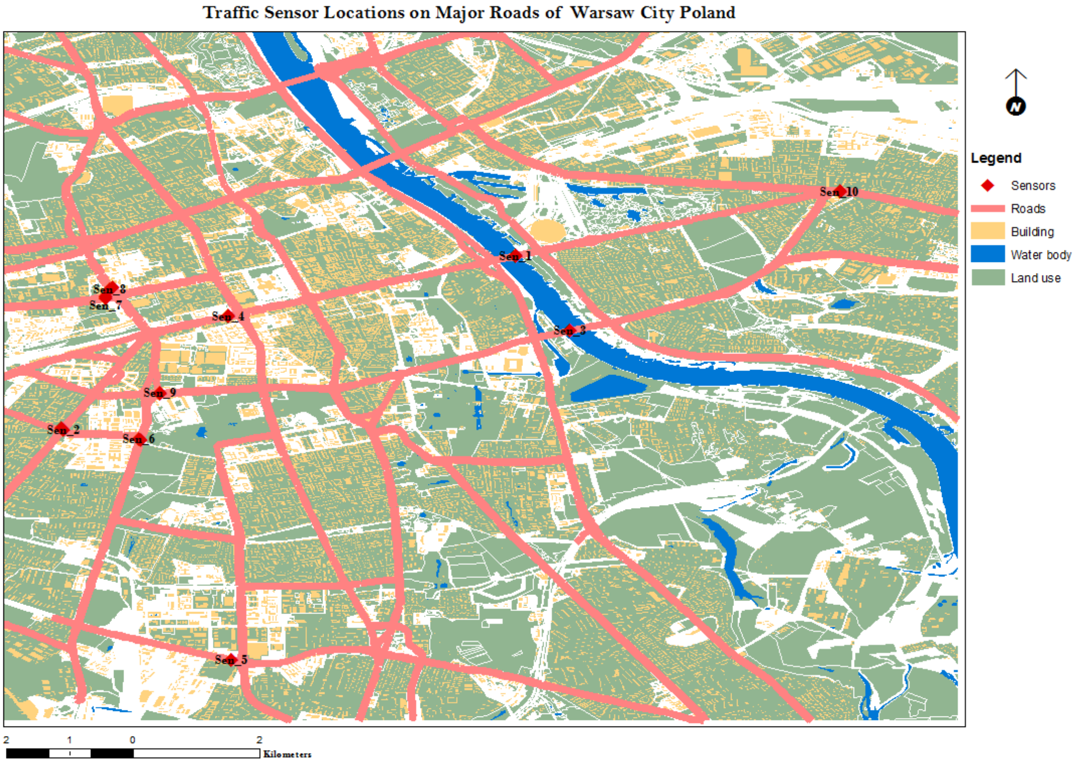


Figure 7: Geographic distribution and relative proximity of sensors

To guarantee the independence assumption of the commonly required *iid* distribution of many statistical learning methods (including *OLS*, *SVM*, etc) the flow data is divided into $n = N/30$ contiguous non-overlapping history windows of independent inputs each of which is associated with a 10 minute long prediction horizon into the future from each subsequent history window. The window splitting results into n number of *iid* observations of predictor / response (independent / dependent) sets of raw measurements. Due to the assumptions put forward in Section 1.5, any potential misalignment of the streams of data that come from 10 independent simulations, is ignored and the different simulation results are considered as continuous records of individual sensors. For each of the n number of *iid* observations of sets of raw measurements $m = 30$ temporal trend, input features and response values are extracted for the dependent and independent parts of the observation set as described in Subsection 3.2. To assess the generality of the models the data set is split into training and validation set using the *k-fold* cross validation, where $k = 10$ as described also in Subsection 3.2.

4.3 Results analysis and evaluation

The range of model performance obtained in terms of the RMSE values of all of the models and their performance on individual sensors and an overall average is given in Table 2 for the linear regression and in Table 3 for the SVMR based models. Furthermore, the coefficient of determination of the *OLS* based models (i.e. R^2) as a measure of the extent to which the independent variables explain the variation in the dependent variable is given in Table 4. The *OLS* based models are also examined against model proper specification through *residuals diagnosis* using *residual histograms* as in Figure 9 and using *normal probability distribution* of residuals as in Figure 10.

4.3.1 Results analysis

Before procedding to the models' performance analysis, it is important to first give a glimpse of the top solutions of the contest and associated RMSE score of each of the winning models as in Table 1. The entire list of submitted solutions during the contest is also available in [22]).

Table 1: RMSE of winning models of the 2010 ICDM contest

Model	RMSE Value
LLS, SVD-like factorization and RBM NN	25.2327
Combination of RF, kNN	25.4167
Random Forests	25.4337
Random Forest with GBM	25.5204

The accuracy assessment and reporting means during the contest was only using the RMSE metric. The result is reported based on the average overall performance of the models. Thus, there is no information on the prediction performance of each model on individual sensors level. Moreover, the validation set selection in the contest was neither restricted into a specific method nor to use a specific set of validation data. However, the models were finally evaluated based on a common test data that was kept secrete only for final evaluation of contest winners. These evaluation test data were kept secrete only for the contest award evaluation and they have never released thereafter.

Table 2: RMSE scores for the OLS based models

SID	Sen.Names	Direction	LR-IS	JLR	JLR-PCA	SWR
1	Most Poniatowsk.	E →W	33.2561	35.7988	31.6746	35.7988
2	Most Poniatowsk.	W →E	13.8143	11.8210	12.8760	11.8210
3	Grjecka	S →N	20.4509	19.2532	20.9334	19.2532
4	Grjecka	N →S	20.7974	20.8885	**19.4897**	20.8885
5	Most azienkowski	E →W	28.0027	26.3588	30.3226	26.3588
6	Most azienkowski	W →E	33.2805	32.7402	**31.7176**	32.7402
7	Aleje Jerozolimsk	E →W	19.5810	17.9776	19.6731	17.9776
8	Aleje Jerozolimsk	W →E	12.6397	10.7180	12.2877	10.7180
9	Marynarska	W →E	32.9484	29.3513	31.7242	29.3513
10	Marynarska	E →W	37.1645	39.1520	**37.0387**	39.1520
11	Zwirki i Wigury	S →N	32.2916	34.2579	**31.7085**	34.2579
12	Zwirki i Wigury	N →S	19.0246	18.0787	18.3920	18.0787
13	Prosta	W →E	10.4642	9.8422	10.9097	9.8422
14	Towarowa	S →N	17.5916	17.4191	**16.2101**	17.4191
15	Grochowska	W →E	15.2753	12.2435	13.0708	12.2435
16	Grochowska	E →W	16.2544	15.7768	**15.6138**	15.7768
17	Wawelska	W →E	36.6846	36.7293	**34.2724**	36.7293
18	Wawelska	E →W	17.7688	20.0970	**15.8075**	20.0970
19	Towarowa	N →S	21.3008	20.3925	23.3605	20.3925
20	Prosta	E →W	18.9332	20.5605	**18.3245**	20.5605
	Avg	-	**22.876**	**22.473**	**22.2343**	**22.4728**

Based on the overall average RMSE results, while the JLR-PCA has highest prediction accuracy, the LR-IS model performed least. On the other hand, the JLR was not influenced by the introduction of the spatial weights. Thus, the JLR and the SWR models produced no difference in terms of their RMSE scores. The JLR-PCA used less number of input features after the dimensionality reduction is applied. Therefore, out of the total number of input features constructed from all neighboring sensors, only 3.33%, which is only the top 10 principal components are used and produced the best RMSE result in the OLS based models. One of the most important assumption of linear regression modeling as described in 2.5.1 is, random variables should not be correlated to each other. Therefore, avoiding collinearity which results in *biased estimates* is effectively applied using orthogonal linear transformation of features by *principal component analysis* on *JLR-PCA* model. While the *LR-IS* model uses input variables constructed only from a single sensor, the *JLR, JLR-PCA* and *SWR* models use input variables constructed from all neighboring sensors combined.

Table 3: RMSE scores for the SVMR based models

SID	Sen. Names	Direc.	SVMR-IS	JSVMR	JSVMR-PCA	SWSVMR
1	Most Poniatowsk.	E →W	39.5874	37.1395	**36.6579**	37.2250
2	Most Poniatowsk.	W →E	14.4420	**12.4970**	12.8833	12.5208
3	Grjecka	S →N	21.0187	**19.2532**	20.8644	19.9493
4	Grjecka	N →S	23.2110	**20.8885**	22.5486	22.4576
5	Most azienkowski	E →W	32.4067	**29.7743**	30.3430	29.8226
6	Most azienkowski	W →E	41.1666	37.1160	**34.3848**	37.0919
7	Aleje Jerozolimsk	E →W	20.7779	18.8057	**18.7711**	18.7821
8	Aleje Jerozolimsk	W →E	12.2938	**10.8151**	11.2254	10.8496
9	Marynarska	W →E	34.0552	31.5396	**30.6008**	31.1411
10	Marynarska	E →W	45.0838	41.9519	**41.5577**	41.9196
11	Zwirki i Wigury	S →N	36.3169	36.0114	**35.2533**	36.0390
12	Zwirki i Wigury	N →S	20.6730	19.1598	19.826	**19.1057**
13	Prosta	W →E	10.4632	9.9350	10.4472	**9.9284**
14	Towarowa	S →N	18.9368	18.1109	18.1185	**18.1076**
15	Grochowska	W →E	13.9330	12.1056	12.4811	**12.0920**
16	Grochowska	E →W	18.2419	16.6868	**16.5972**	16.7011
17	Wawelska	W →E	45.9776	39.6157	39.8674	**39.2911**
18	Wawelska	E →W	25.9643	22.0123	**21.4320**	22.2020
19	Towarowa	N →S	22.1855	20.4658	21.4349	**20.3417**
20	Prosta	E →W	22.5018	21.6558	21.6350	**21.6028**
	Avg	-	**25.8962**	**23.8880**	**23.8465**	**23.8586**

While the SVMR-IS model uses variable constructed from a single sensor, the JSVMR, JSVMR-PCA and *SWSVMR* models use input variables constructed from all neighboring sensors combined. Similar to the linear regression based model on which dimensionality reduction was applied, the JSVMR-PCA used input features constructed from all neighboring sensors and an orthogonal linear transformation using principal component analysis was applied as a dimensionality reduction. As a result, based on the model stability and significance of the component variance in the principal components, only 14 principal components which is 4.7% of the total input features are used to train the JSVMR-PCA. Thus, comparing with the SVMR based models, JSVMR-PCA provides the highest accuracy based on the overall RMSE. Unlike the linear regression based spatially weighted regression–SWR model, the SWSVMR mode provides accuracy scores that are higher than the JSVMR.

Before explicitly including spatial weights into the traffic data in the neighboring roads of a prediction location, incorporating input features from them could give certain clue about any potential interdependence among them. Therefore, the *JLR* and *JSVMR* models are designed and implemented by taking input features from all neighboring sensors to make prediction on a specific sensor. These two models do not include any weighting. The results obtained from these two models is slightly higher than those models that do not consider any input from neighboring sensors. Therefore, the *JLR* and *JSVMR* models indicate that the traffic flow in the neighboring sensors has certain impact on the flow amount of a sensor's traffic flow in the prediction location.

To evaluate the *JLR* and *JSVMR* models' RMSE scores, it is important to take care of any potential multicollinearity in such a high dimensional set of input variables extracted from all neighboring sensors. Therefore, using an orthogonal linear transformation of the input features using principal component analysis, *JLR-PCA* and *JSVMR-PCA* models helped to avoid any potential threat of multicolinearity in the high dimensional input feature set. In addition, these two models also optimized the computing time.

To substantiate the spatial dependence of sensors indicated by the JLR and JSVMR models, explicitly incorporating measures for spatial relationship between each prediction location (sensor) and the locations of the set of neighboring sensors it is possible to reveal which neighboring sensors are impacting the flow to what extent. Accordingly, *SWR* and *SWSVMR* models take into account the contribution of each sensor based on individual sensor's geographic proximity to the prediction location. This is realized by using *Weighted least square* like technique that gives more weight to the nearby sensors and less to those farther based on a distance decay function using shortest path network–distance from the point of prediction. The relative proximity among each of the sensors computed using ArcGIS Network Analyst tool is illustrated on Figure 8.

Notably, from Tables 2 and 3, all of the proposed models but *SVMR-IS* performed better than the results of the top solutions selected during the contest based on the overall average prediction accuracy measured in RMSE. With regard to the support vector machine for regression based models, the SVMR-IS performed least even when compared to the winning models and the linear regression based models in this study.

Generally, one of the likely reasons to obtain these better and in some cases highly competitive prediction accuracy results is the effectiveness of the temporal trend of feature construction that is proposed in the present paper. Moreover, the use of input features constructed from all neighboring road segments to individual prediction location enhanced the accuracy both in the OLS and SVM based models. Further, the use of *SVMR* as a robust and in many ways a superior algorithm than the linear regression methods bridges the gaps observed in the linear regression models. By constructing features from neighboring sensors, not only enhances the prediction accuracy but also indicated the interdependence of road segments in traffic flow.

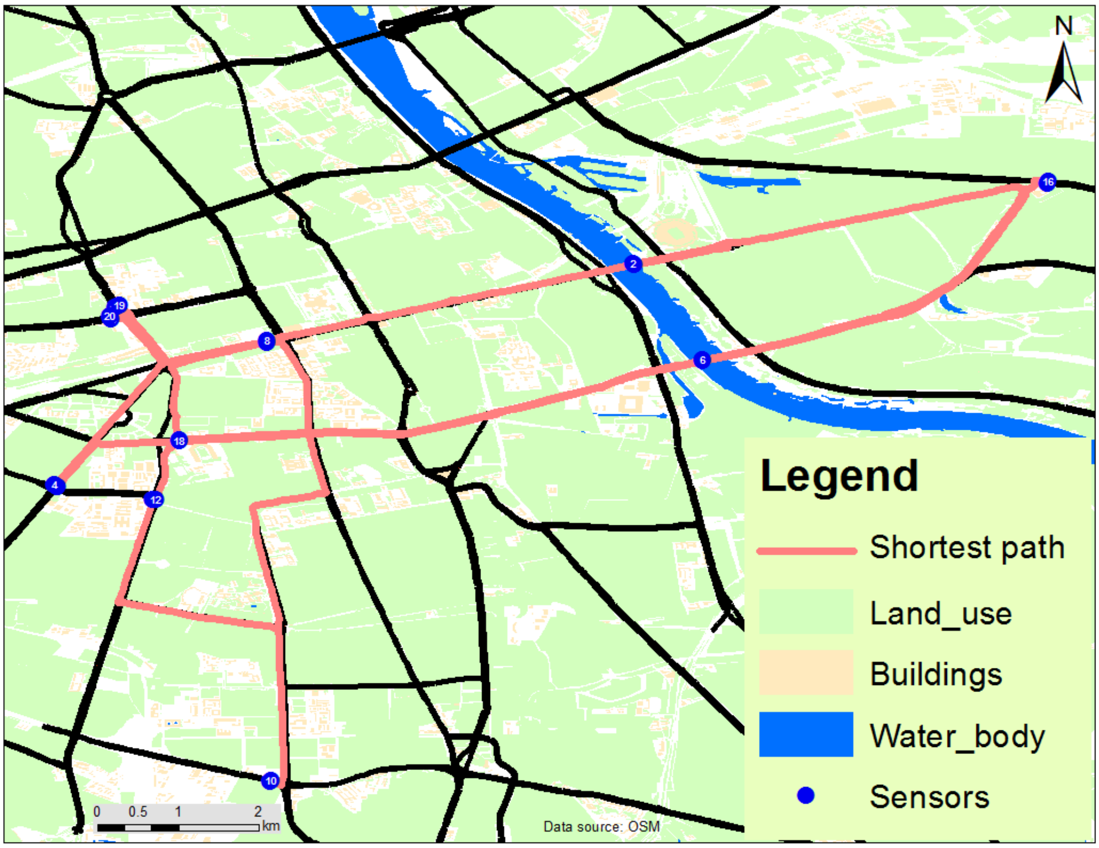


Figure 8: Spatial distribution and shortest path based network distance to sensor locations

In the linear regression based models, the impact of the spatial weights brings no change in the SWR when compared to the JLR model. However, the same spatial weighting approach has produced different prediction accuracy results in which in many cases with higher performance in the SWSVMR model. The spatial weights introduced in the spatially weighted support vector machine for regression clearly reveal the impact of road network topology on traffic flow. This is indicated by the *significant variation of RMSE* values at the individual sensors when compared with the models without any spatial weighting. As there was no results of individual sensors' prediction accuracy reported from the contest, the only way comparison is possible is using the average RMSE value over all sensors.

In comparing the results obtained in this paper, and that of the winning solutions of the contest, the main concern lies on the similarity of validation data sets. In this study, the validation set is selected from the training set based on the k-fold cross validation method, with $k = 10$. Therefore, there could be possibilities that the top selected solutions during the contest might have used different amount and techniques of selecting validation set. However, the validation data selection adopted in this study is commonly practiced. Moreover, since the training data is very large, *k-fold* cross validation is highly effective. To observe if any different validation set could result in different RMSE score, each model was repeatedly executed and no significant variation was recorded. Therefore, it is possible to conclude that the method adopted for validation set selection is appropriate and its results are acceptable. The main challenge with regard to validation is the portion of the test data kept secrete for evaluation during the contest is not available for us at this stage.

Even though there is not large RMSE value differences, the *SWSVMR* model produces variations among the different parameters estimated, the individual sensor's RMSE results as well as on the final prediction results of the testing data set when the spatial weighting is introduced. Specifically, by introducing the spatial weighting into the SVMR models, 7 out of 20 sensors (i.e. 35%) of the sensors have improved the prediction accuracy. All of the sensors but one are located in the sensors clustered in a nearby geographic proximity. In addition to the sensors with improved accuracy when spatial weighting is introduced, the sensors have shown certain level of variation in their RMSE score. Both the improvement and any change from the the other models, especially when compared with the JSVMR is result of the spatial weighting. Therefore, it is reasonable to conclude that, the introduction of spatial weights based on the shortest path network distance of each neighboring sensor to the location of prediction produces more realistic prediction results at individual sensor level in the SWSVMR model. One of the possible reasons as to why the SWR model does not vary when compared with the JLR could be the limitation of the linear learning process of OLS.

In addition to the RMSE measure the *OLS* based models can also be evaluated using *coefficients of determination* which explains how much of variability in the dependent variable is explained by the independent variables i.e. R^2 value. The R^2 values of the proposed linear regression based models is given in Table 4. Though coefficient of determination can be used to evaluate the models' effectiveness, it does not indicate whether a regression model per se is an adequate model for a specific situation. It is not uncommon to have a low R^2 value for a good model, or a high value but that does not fit the data [3]. High R^2 does necessary mean always a good fit and low R^2 as bad fit, because R^2 has its own limitations such as it cannot determine whether the coefficient estimates and predictions are biased or not [40]. But in general one opts to obtain higher R^2 value.

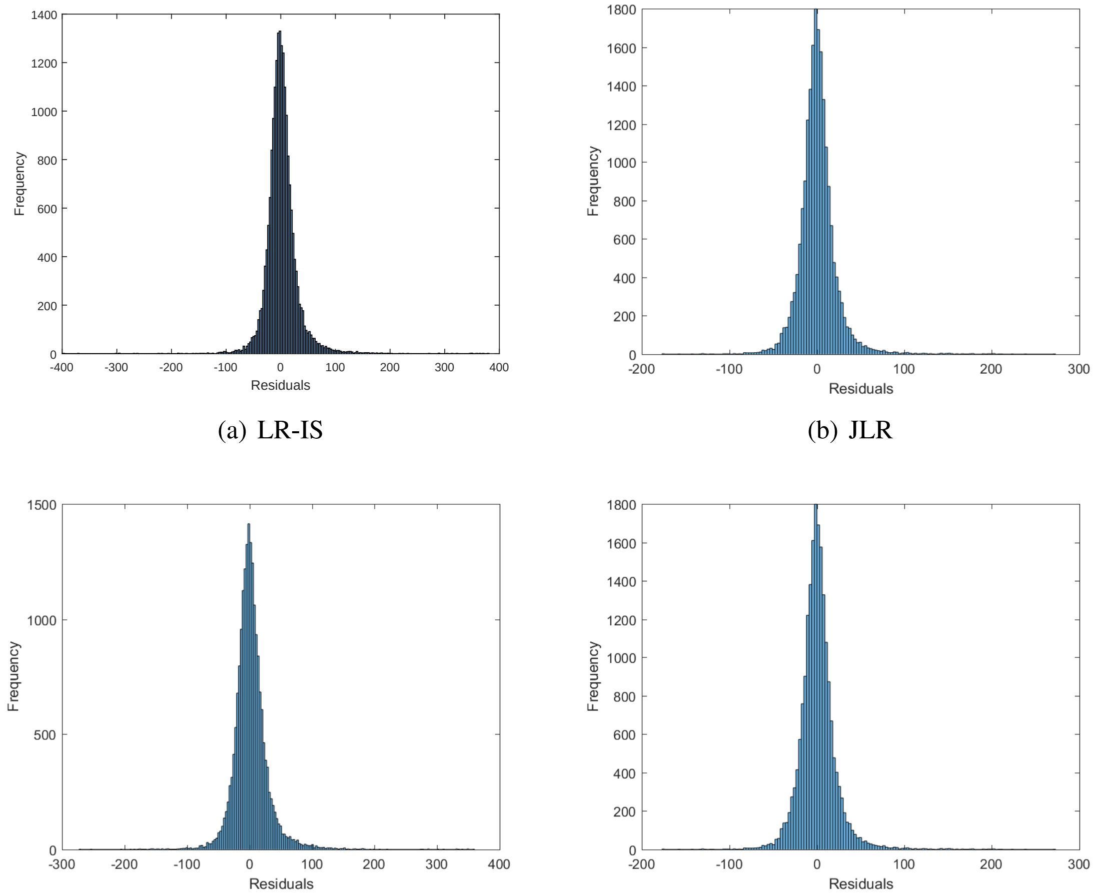
When the overall average of R^2 values of each model is examined, Table 4 shows that some sensors have as low coefficient of determination value as 5.686% in LR-IS, 37.045% in JLR, 7.232% in JLR-PCA and 7.046% in SWR models. On the other hand, there are sensors that have as high coefficient of determination as 91.642% in LR-IS, 94.459% in JLR, 90.519% in JLR-PCA and 91.7757% in SWR models.

Table 4: Individual sensors R^2 values

SID	Sen. Names	Direction	LR-IS	JLR	JLR-PCA	SWR
1	Most Poniatowskiego	E → W	18.875	43.875	26.518	19.7759
2	Most Poniatowskiego	W → E	54.729	75.966	60.858	54.6864
3	Grjecka	S → N	48.224	66.165	45.519	48.9581
4	Grjecka	N → S	9.469	42.900	20.298	12.4247
5	Most azienkowski	E → W	91.642	94.459	90.519	91.7757
6	Most azienkowski	W → E	5.686	63.305	7.232	7.0462
7	Aleje Jerozolimskie	E → W	22.204	52.042	28.253	23.1304
8	Aleje Jerozolimskie	W → E	55.600	76.736	61.801	56.2089
9	Marynarska	W → E	38.871	69.374	48.332	38.8650
10	Marynarska	E → W	71.115	75.468	67.065	70.8085
11	Zwirki i Wigury	S → N	43.668	56.642	44.069	43.566
12	Zwirki i Wigury	N → S	8.973	43.723	17.374	10.3241
13	Prosta	W → E	18.764	47.143	16.150	18.8083
14	Towarowa	S → N	37.073	59.545	46.770	38.7707
15	Grochowska	W → E	44.863	75.458	62.184	58.1283
16	Grochowska	E → W	33.556	62.768	35.693	34.9587
17	Wawelska	W → E	87.213	89.543	88.622	87.5806
18	Wawelska	E → W	53.641	72.752	58.738	53.1958
19	Towarowa	N → S	48.061	64.951	43.429	48.0570
20	Prosta	E → W	10.391	37.045	14.796	10.7657
Avg	-	-	**40.131**	**63.493**	**44.211**	**41.3918**

In general, in this study, the improvement from the *LR-IS*, *JLR*, *JLR-PCA* to *SWR* models clearly demonstrated that the prediction accuracy varies across models and prediction accuracy increases when combined input from neighboring sensors is used. Any potential risks of colliearity is avoided using PCA and improves the model's stability. Moreover, even though the RMSE metric does not show any variation or improvement in the SWR model's prediction accuracy, by introducing the spatial weighting, the coefficient of determination demonstrates that the road network topology has certain impact on the traffic flow measures by influencing the R^2 measure in the individual sensors. Hence, the impact of road networks topology in the prediction accuracy on individual sensors is reasonably recognized.

On the other hand, the appropriateness of the linear regression based models for the problem is important and need to be examined. *Residual analysis* is one of the common approache to examine if linear regression models are specified properly [10]. Residuals are the difference between observed *(y)* and predicted value $(\hat{y})$ of an OLS model also known as ϵ, characterized by $\sum \epsilon_i = 0$ and the $\bar{\epsilon} = 0$. The linear regression based models proposed in this paper are thus examined using *histogram plots* and the *Normal probability plot* of residuals as on Figures 9 and 10 respectively.



(a) LR-IS

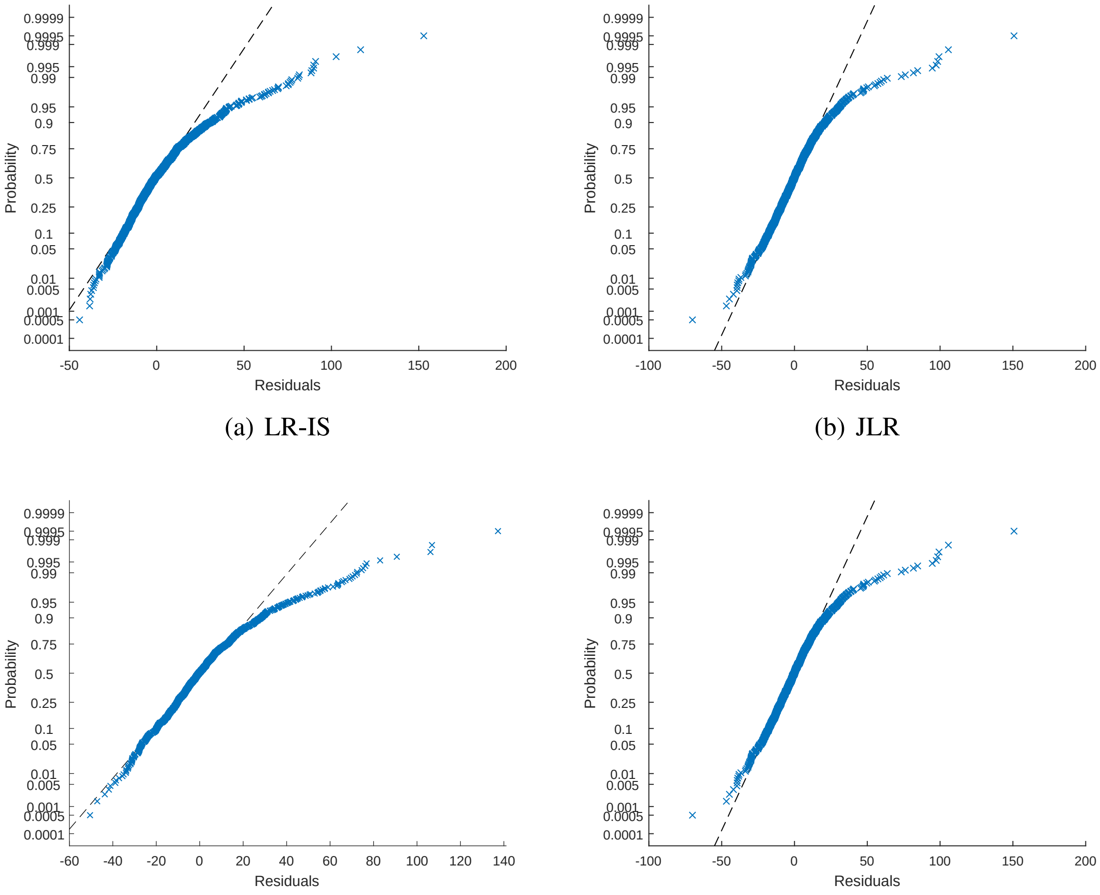
(b) JLR

(c) JLR-PCA

(d) SWR

Figure 9: Linear regression models: histogram of residuals

Based on a simple visual examination of the residuals histogram plots, all the plots are somehow perfect bell-shaped, which indicates that the error terms of the models are *normally distributed* with $\epsilon_i = N(\sigma^2, 0)$. However, a closer look into the histograms can give indications for a slight skewness in the histograms. But as the data set is large in volume, the skewness might not be clearly visible. Hence, further examination of the error terms may provide better insights into the appropriate specification of the models.



(a) LR-IS

(b) JLR

(c) JLR-PCA

(d) SWR

Figure 10: Linear regression models: residuals probability plots

Another technique to diagnoses the residuals of the linear regression models is using the normal probability plots where the straight line indicates a *normality line* and residual points aligned along that line are normally distributed. Hence, it can be concluded that, the improvement in the appropriateness of the *linear regression* slightly increased from the *LR-IS*, *JLR*, *JLR-PCA* to *SWR* as on Figure 10. In these figures, the misalignment is decreasing in both directions. The slight misalignment observed in the normal probability plot of the residuals could hint other models than the linear regression may fit to the prediction problem better. Therefore, the introduction of the support vector machine for regression modeling techniques which uses a *non-linear* learning approach could bridge these limitations of the OLS linear learning method.

4.3.2 Discussion

Data-driven traffic flow prediction problem has been attracting scientific focus mainly due to the advancement in computational power and intelligence as well as due to the ever growing accumulation of traffic flow data. Moreover, the operational limitations, associated costs and lack of comprehensiveness of physical traffic sensors have contributed to the emergence and popularity of data-driven traffic flow prediction as critical components of ITS. Hence, data mining and machine learning methods have been continuously in focus of the research community. Apart from the common approaches in traffic flow prediction that is time-series based temporal patterns analysis, spatial dimensions from the road network topology and its impact on flow measurement is becoming important in the area. The main objective of this research is to show that traffic flow prediction tasks are not only affected by temporal trends of flow history, but also by road network topology; thus, prediction methods in the *spatio-temporal* domain would produce more accurate and realistic results. Hence, this study is based on the traffic flow prediction problem from the 2010 ICDM international data mining research challenge organized by IEEE and TomTom.

To examine if past traffic flow data can help forecast the flow into the future, certain correlation between past and future traffic flow in the historical data itself is important. Thus, using auto–correlation methods based on time–series operators such as the temporal lags, flow trends are mined and causal relationship is formed. As a result, dependent variables using partially overlapping temporal offset are defined on sequence of non-overlapping 30 minutes long history windows defined on the historical data. Each history window is a history window for the observation set on the succeeding window. To represent traffic flow values in each temporal offset, statistical mean, standard deviation and rate of change in traffic flow are used. Hence, large set of input variables are constructed both linear and non-linear based learning modeling techniques are proposed, designed and implemented.

Model input features are constructed and designed in such a way that predictions are possible using historical data of a single sensor as well as input features combined from multiple of neighboring sensors, so that impact of nearby road segments would be examined. Moreover, the models design is based on several stages in which models be designed without any assessment of collinearity in the input variables followed by an intervention on collinearity through dimensionality reduction techniques. In addition, to demonstrate if spatial dependency exists in traffic flow, models are designed in such a way that spatial dimension is ignored followed by considering spatial dimension through spatially weighting input features based on the geographic proximity of sensors to the location of prediction.

The first set of models, LR-IS, JLR, JLR-PCA and SWR that are all based on the linear regression modeling technique produced prediction accuracy all higher than the winning solutions based on the RMSE score. Moreover, the second set of models, SVMR-IS, JSVMR, JSVMR-PCA and SWSVM that are all based on the emerging popular machine learning classification algorithm developed for regression problems using the Gaussian kernel function is designed for non-linear learning. With the same approach for feature construction and model learning to that of the linear regression models, the SVMR based models produced slightly less RMSE scores, but higher than the winning solutions during the contest with the exception of SVMR-IS. Moreover, in the JSVMR-PCA slightly higher number of principal components are used to train the model that the JLR-PCA.

To directly compare the models RMSE score was the only parameter given. However, it is essential that models be correctly specified, fundamental principles of linear regression and support vector machine be fulfilled as well as prediction results be investigated. Moreover, other prediction evaluation methods be introduced so that not only direct comparison with the winning solution but also the proposed models be evaluated in relative to one another. Hence, as a measure of the extent to which independent variables explain the dependent variable, coefficient of determination (i.e. R^2) values of the linear regression models are examined. Accordingly, the linear regression based models R^2 values range from as low as 5.686%, to as high R^2 value as 94.459%. This variation can given further insights into to what extent the R^2 value spreads and how effective the constructed variables are in explaining the response variable.

Looking at the variability of the coefficient of determination among the 20 sensors, while the LR-IS model R^2 values have a standard deviation of 25.02 and an average R^2 of 40.13%, the JLR model resulted in average R^2 of 63.49% and a standard deviation of 15.8. In a similar manner, the JLR-PCA resulted in average R^2 value of 44.2% and a standard deviation of 23.77 as well as the SWR model provided an average R^2 value of 41.39% and a standard deviation of 24.86. These figures can indicate that the coefficient of determination of each model varies significantly, and the average R^2 is relatively low with the exception of the JLR model. Moreover, the variation (i.e. high value of standard deviation), among the sensors R^2 values show that in general the prediction capability of the linear regression based models highly varies among each of the sensors.

As illustrated on Table 4, in the *LR-IS* and *SWR* models, while 30% (i.e. 6) sensors have R^2 value that is greater than 0.5 times the standard deviation from the mean, 35% (i.e. 7) sensors that have R^2 value less than -0.5 times the standard deviation less than the mean. On the other hand, in the *JLR* model, while 35% (i.e. 7) sensors have R^2 value that is greater than 0.5 times the standard deviation from the mean, 30% (i.e. 6) sensors that have R^2 value less than -0.5 times the standard deviation less than the mean. Similarly, in the *JLR-PCA* model, while 35% (i.e. 7) sensors have R^2 value that is greater than 0.5 times the standard deviation from the mean, 35% (i.e. 7) sensors that have R^2 value less than -0.5 times the standard deviation less than the mean. Thus, it is clearly indicated that the standard deviation of the R^2 of all models is relatively high and that means the explaining power of the independent variables is relatively low in several sensors.

Generally, all the *OLS* based models produced higher accuracy prediction results when compared to the top selected solutions during the research competition based on a *k-fold* cross validation procedure. Since there exist large volume of data to use as training and validation, RMSE values could reasonably be used as good comparison methods. Though RMSE and R^2 results of the models is high in some sensors and low in other sensors, it is also evident that variations among sensors can be associated with the effectiveness of the features construction, the training model itself, the impact of the road network topology introduced through the spatial weighting as well as the use PCA to avoid collinearity among input features. But, in terms of the models proper specification, all the histogram representations of residuals analysis as on Figure 9 fulfills the assumption of the normal distributions of error terms with $N(\sigma^2, 0)$ to some extent. Moreover, the residual probability plot of these models demonstrated very large percentage of the residuals lie on the normality line. Thus,it is appropriate to conclude that the *OLS* based models proposed are well specified and adds up to the credibility of the *RMSE* and R^2 results obtained. However, some amount of the residuals are misaligned to the normality line of the residual probability plots. This accounts mainly to potential existence of *(a) outliers in the data* and *(b) certain non-linear relationship* in the data.

The misalignment in the probability plot of residuals may indicate the need for a non-linear modeling approach. Therefore, to examine the learning algorithm itself, a robust and mathematically sound statistical learning algorithm, SVMR is designed and implemented on similar operational setup. SVMR, a variant of the well known support vector machine highly effective in classification problems is recently emerging method for regression problems, is used to address the two concerns raised. All the input structures and the construction of the input features is identical to the *OLS* methods with a slight larger principal components selected in the *JSVMR-PCA* model than in the *JLR-PCA*. All the SVMR based models produced higher RMSE results than the selected models during the competition but the *SVMR-IS*. Though it is possible to implement SVMR in different operational setup such as using different solver algorithms, for scope reasons, the default setting of **fitrsvm**, on MATLAB version 2016a implementation *Sequential Minimal Optimization (SMO)* solver produced acceptable results in the context of the contest results as indicated in Table 1 and the results of the *OLS* based models as in Table 2. Moreover, the proposed models clearly indicated that the different road segments contribute differently onto the traffic flow prediction at each of the prediction location; thus, incorporating the impact of road network topology in the modeling process is important for and an intrinsic property of traffic flow in a network of road segments.

5 Conclusion and Future Work

The following subsection provide concluding remarks mainly on the features construction, models design and training as well as validation and evaluating results. Moreover, what future additions and modification the author recommends in order to enhance the results are briefly indicated.

5.1 Conclusions

The main objective of this research is to show that traffic flow prediction tasks are not only affected by temporal trends of flow history, but also by road network topology; thus, prediction methods in the *spatio-temporal* domain would produce more accurate and realistic results. Hence, this study is based on the traffic flow prediction problem from the 2010 ICDM international data mining research challenge organized by IEEE and TomTom. The traffic prediction research task took data sets from selected ten road segments in Warsaw City Poland, generated from a 100 hours 10 independent simulations produced from the Traffic Simulation Framework (TSF) which was developed based on the NagelSchreckenberg model.

Accordingly, from each sensor's historical flow data, five different partially overlapping relative temporal offsets were defined on a *30 minutes* long non-overlapping history window based on a *5, 10 and 30 minutes* temporal scales. The traffic flow values of each of the temporal offsets were represented by three statistical properties namely *statistical mean, rate of change and standard deviation* of flow measurements. Thus, from a pair of consecutive history windows, fifteen predictors from each preceding window and one response variable from each succeeding window were constructed for every sensor's stream of historical records. Following the features construction, two sets of prediction modeling approaches based on *Linear Regression* and *Support Vector Machine for Regression* were proposed. The models were implemented based on the MathWorks® MATLAB software standard implementation. These models were designed based on different formulation of inputs. The formulation is based on inputs constructed from a single sensor, from multiple sensors, from multiple sensors combined and spatially weighted as well as inputs constructed from a combination of neighboring sensors and a dimensionality reduction technique *Principal Component Analysis* based on the *Singular Value Decomposition* algorithm applied.

Input features constructed from different sensors combined were used to train and predict traffic flow at a specific sensor location mainly to examine if there exist interdependence among road segments (i.e. sensors). Spatially weighted regression model then examines the effect of the road network topology by spatially weighting features based on geographic proximity of each of the sensors to the location of prediction. To realize the spatial dependence in traffic flow prediction, a distance decay function using the shortest path network–distance among sensors was employed. When input features are constructed from these multiple sensors and through the statistical properties on the temporal offset defined, risks of multicolliearity are evident. Thus, models that are trained based on such inputs may produce *unreliable, biased or unstable coefficient* estimates. By applying PCA, it was possible to enhance the computational cost of the models by using smaller set of variables which accounts for only 3.33% of the total variables constructed from all neighboring sensors in the case of linear regression with PCA and 4.7% in the case of the support vector machine for regression with PCA. Based on this operational set up the models are evaluated based on the root mean square error values of a $k - fold$ cross validation method, where $k = 10$. Moreover, the OLS based models are evaluated based on the coefficient of determination that is R^2 as well as using basic *residual diagnosis* against model appropriateness.

All the OLS based models produced higher accuracy prediction results when compared to the top selected solutions during the research competition based on a $k - fold$ cross validation procedure. The use of RMSE score as a models' accuracy assessment can be accepted as the training data is very large. While the models proved that when the impact of road network topology is included in the modeling, the accuracy level varies with the SVMR based models and this indicates that not only temporal patterns of historical traffic flow but also road topology can impact traffic flow measurements Further the independent variables explanation power to the dependent variables is significant in some sensors and less significant in others. But generally, the coefficient of determination can indicate that the OLS based models have certain limitations in in such a problem. Moreover, the residual probability plot of these models demonstrated very large percentage of the residuals lie on the normality line but still show that there are misalignment mainly because of *(a) potential outliers in the data* and *(b) potentially non-linear relationships* may exist in the data. Therefore, SVMR is designed and implemented on similar operational setup to address the gaps in the OLS based morels. Hence, all the SVMR based models produced higher RMSE results than the selected models during the competition but the *SVMR-IS*. The SVMR based models clearly indicated that the different road segments contribute differently, and incorporating the impact of road network topology in the modeling process is important for and an intrinsic property of traffic flow in a network of road segments.

However, the following concerns can be raised on the overall approach. These are (1) by applying PCA on the input features, it is difficult to distinguish which set of variables contributed to what extent in the prediction accuracy. Since it's difficult to examine the large number of features individually, it requires more effort to study how each possible feature construction procedure followed in this paper behaves with each sensor data and each proposed model. (2) The cross validation approach implemented on both **fitlm** and **fitrsvm** methods in MATLAB system are based on *random selection of* n/k number of examples. Thus, any potential temporal trend of each of the history window from which individual features were constructed might be lost because of the randomness. Hence, selecting n/k consecutive examples may retain any temporal trends in the features from consecutive history windows. A preliminary assessment on the *LR-IS* produced very close RMSE result when validation set was defined based on n/k number of consecutive examples for each k. However, this can be recommended for further investigation for all proposed models. (3) Even though it might not be needed in this particular situation, applying PCA on large set of feature sets and considering only few top principal components makes it difficult to examine the effect of individual features on the prediction process. This is difficult to interpret individual variables $p - values$ during the training as individual variables no more hold values from a specific feature construction technique.(4) By getting access to the validation data set of each winning model, and to the half part of the test data that were used for contest winners selection, the models proposed in this study can be tested and complete direct comparison to the winning models would be possible. Nevertheless, the models proposed here are well specified as per the fundamental principles of linear regression. It was also possible to demonstrate the impact of road network effect on traffic flow measurements in a network of urban road segment.

5.2 Future work

This research can potentially be enhanced by employing more statistical properties such as measures of extremes values (minimum and maximum), measures of inter-quartile ranges etc. to represent group of values in the partially overlapping temporal offsets defined on the non–overlapping sequence of history windows of the history window during the features construction. Moreover, to investigate any non–linear relationship among the variables constructed based on different temporal scale and different statistical measures on the history windows non–linear Principal component analysis as dimensionality reduction technique may provide deeper insights into the characteristics of the data set. Furthermore, investigating the solver optimization algorithms of SVMR such as the $ISDA$, $L1QP$ Algorithms be used into SVMR so that SVMR may deal with the data in a better way than the OLS method.

References

[1] Ahn JY, Ko E, Kim E, editors. Predicting Spatiotemporal Traffic Flow Based on Support Vector Regression and Bayesian Classifier. Big Data and Cloud Computing (BDCloud), 2015 IEEE Fifth International Conference on; 2015: IEEE.

[2] Anderson OD. Time series analysis and forecasting: the Box-Jenkins approach: Butterworths London; 1976.

[3] Belloto J, Sokolovski T. Residual analysis in regression. American Journal of Pharmaceutical Education. 1985;49(3):295-303.

[4] Bermolen P, Rossi D. Support vector regression for link load prediction. Computer Networks. 2009;53(2):191-201.

[5] Brunsdon C, Fotheringham AS, Charlton ME. Geographically weighted regression: a method for exploring spatial nonstationarity. Geographical analysis. 1996;28(4):281-98.

[6] Brunsdon C, Fotheringham S, Charlton M. Geographically weighted regression. Journal of the Royal Statistical Society: Series D (The Statistician). 1998;47(3):431-43.

[7] Castro-Neto M, Jeong Y-S, Jeong M-K, Han LD. Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. Expert systems with applications. 2009;36(3):6164-73.

[8] Cheng T, Haworth J, Wang J. Spatio-temporal autocorrelation of road network data. Journal of Geographical Systems. 2012;14(4):389-413.

[9] Cong Y, Wang J, Li X. Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm. Procedia Engineering. 2016;137:59-68.

[10] Cook RD, Weisberg S. Residuals and influence in regression: New York: Chapman and Hall; 1982.

[11] Council, T. E. P. a. The Framework for the Development of Intelligent Transport Systems in the Field of Road Transport and for Interfaces with Othr Modes of Transport. In D. 2010/40/EU (Ed.), (207/1 ed.), 2010.

[12] Croarkin C, Tobias P, Zey C. Engineering statistics handbook: NIST iTL; 2002.

[13] Davis GA, Nihan NL. Nonparametric regression and short-term freeway traffic forecasting. Journal of Transportation Engineering. 1991;117(2):178-88.

[14] . Desmet L, Gijbels I. Local linear fitting and improved estimation near peaks. Canadian Journal of Statistics. 2009;37(3):453-75.

[15] Dong C, Xiong Z, Shao C, Zhang H. A spatialtemporal-based state space approach for freeway network traffic flow modelling and prediction. Transportmetrica A: Transport Science. 2015;11(7):547-60.

[16] Flores JHF, Engel PM, Pinto RC, editors. Autocorrelation and partial autocorrelation functions to improve neural networks models on univariate time series forecasting. Neural Networks (IJCNN), The 2012 International Joint Conference on; 2012: IEEE.

[17] Fotheringham AS, Brunsdon C, Charlton M. Geographically weighted regression: the analysis of spatially varying relationships: John Wiley & Sons; 2003.

[18] Habtemichael FG, Cetin M. Short-term traffic flow rate forecasting based on identifying similar traffic patterns. Transportation Research Part C: Emerging Technologies. 2016;66:61-78.

[19] Hamed MM, Al-Masaeid HR, Said ZMB. Short-term prediction of traffic volume in urban arterials. Journal of Transportation Engineering. 1995;121(3):249-54.

[20] Hotelling H. Analysis of a complex of statistical variables into principal components. Journal of educational psychology. 1933;24(6):417.

[21] Hyndman RJ, Athanasopoulos G. Forecasting: principles and practice: OTexts; 2014.

[22] IEEE ICDM Contest. Overview of Top Solutions. http://blog.tunedit.org/2010/10/26/ieee-icdm-contest-top-solutions-1/#more-339 (Accessed: 2016-09-20)

[23] Kamarianakis Y, Prastacos P. Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches. Transportation Research Record: Journal of the Transportation Research Board. 2003(1857):74-84.

[24] Karlsson M, Yakowitz S. Rainfall-runoff forecasting methods, old and new. Stochastic Hydrology and Hydraulics. 1987;1(4):303-18.

[25] Klein LA, Mills MK, Gibson DR. Traffic Detector Handbook: -Volume II. 2006.

[26] Ko E, Ahn J, Kim EY. 3D Markov process for traffic flow prediction in real-time. Sensors. 2016;16(2):147.

[27] Kohavi R, editor A study of cross-validation and bootstrap for accuracy estimation and model selection. Stanford, CA. Ijcai; 1995:14(2):1137–1145

[28] Koller D, Sahami M. Toward optimal feature selection. Stanford InfoLab, 1996.

[29] Leduc G. Road traffic data: Collection methods and applications. Working Papers on Energy, Transport and Climate Change. 2008;1(55).

[30] Lippi M, Bertini M, Frasconi P. Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning. IEEE Transactions on Intelligent Transportation Systems. 2013;14(2):871-82.

[31] Lovrić M, Milanović M, Stamenković M. Algoritmic methods for segmentation of time series: An overview. Journal of Contemporary Economic and Business Issues. 2014;1(1):31-53.

[32] Lv Y, Duan Y, Kang W, Li Z, Wang F-Y. Traffic flow prediction with big data: a deep learning approach. IEEE Transactions on Intelligent Transportation Systems. 2015;16(2):865-73.

[33] Lu H-p, Sun Z-y, Qu W-c. Big data-driven based real-time traffic flow state identification and prediction. Discrete Dynamics in Nature and Society. 2015;2015.

[34] Liu Y, Feng X, Wang Q, Zhang H, Wang X. Prediction of urban road congestion using a Bayesian network approach. Procedia-Social and Behavioral Sciences. 2014;138:671-8.

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

[36] McGill R, Tukey JW, Larsen WA. Variations of box plots. The American Statistician. 1978;32(1):12-6.

[37] Müller K-R, Smola AJ, Rätsch G, Schölkopf B, Kohlmorgen J, Vapnik V, editors. Predicting time series with support vector machines. International Conference on Artificial Neural Networks; 1997: Springer.

[38] Min W, Wynter L. Real-time road traffic prediction with spatio-temporal correlations. Transportation Research Part C: Emerging Technologies. 2011;19(4):606-16.

[39] Moore B. Principal component analysis in linear systems: Controllability, observability, and model reduction. IEEE transactions on automatic control. 1981;26(1):17-32.

[40] Nau R. Statistical forecasting: notes on regression and time series analysis. Durham: Fuqua School of Business, Duke University. 2015.

[41] Okutani I, Stephanedes YJ. Dynamic prediction of traffic volume through Kalman filtering theory. Transportation Research Part B: Methodological. 1984;18(1):1-11.

[42] Ozbay K, Kachroo P. Incident management in intelligent transportation systems. 1999.

[43] Peng Y, Chen K, Wang G, Bai W, Ma Z, Gu L, editors. Hadoopwatch: A first step towards comprehensive traffic forecasting in cloud computing. INFOCOM, 2014 Proceedings IEEE; 2014: IEEE.

[44] Rong Y, Zhang X, Feng X, Ho T-k, Wei W, Xu D. Comparative analysis for traffic flow forecasting models with real-life data in Beijing. Advances in mechanical engineering. 2015;7(12).

[45] Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. science. 2000;290(5500):2323-6.

[46] Saefuddin A, Saepudin D, Kusumaningrum D. Geographically Weighted Poisson Regression (GWPR) for Analyzing The Malnutrition Data in Java-Indonesia. 2013.

[47] Sánchez A VD. Advanced support vector machines and kernel methods. Neurocomputing. 2003;55(1-2):5-20.

[48] Schimbinschi F, Nguyen XV, Bailey J, Leckie C, Vu H, Kotagiri R, editors. Traffic forecasting in complex urban networks: Leveraging big data and machine learning. Big Data (Big Data), 2015 IEEE International Conference on; 2015: IEEE.

[49] Smith BL, Williams BM, Oswald RK. Comparison of parametric and nonparametric models for traffic flow forecasting. Transportation Research Part C: Emerging Technologies. 2002;10(4):303-21.

[50] Smith BL, Demetsky MJ. Traffic flow forecasting: comparison of modeling approaches. Journal of transportation engineering. 1997;123(4):261-6.

[51] Smola AJ, Schlkopf B. A tutorial on support vector regression. Statistics and computing. 2004;14(3):199-222.

[52] Tewolde GS, editor Sensor and network technology for intelligent transportation systems. Electro/Information Technology (EIT), 2012 IEEE International Conference on; 2012: IEEE.

[53] Tobler W. On the first law of geography: A reply. Annals of the Association of American Geographers. 2004;94(2):304-10.

[54] Transport Research Center. Managing Urban Traffic Congestion. European Conference of Ministers of Transport: OECD/ECMT2007. ISBN 978-92-821-0128-5, 2007

[55] Vapnik V, editor Principles of risk minimization for learning theory. NIPS; 1991.

[56] Vlahogianni EI, Karlaftis MG, Golias JC. Short-term traffic forecasting: Where we are and where were going. Transportation Research Part C: Emerging Technologies. 2014;43:3-19.

[57] Vlahogianni EI, Karlaftis MG, Golias JC. SpatioTemporal ShortTerm Urban Traffic Volume Forecasting Using Genetically Optimized Modular Networks. ComputerAided Civil and Infrastructure Engineering. 2007;22(5):317-25.

[58] Vlahogianni EI, Golias JC, Karlaftis MG. Shortterm traffic forecasting: Overview of objectives and methods. Transport reviews. 2004;24(5):533-57.

[59] Wang J, Deng W, Guo Y. New Bayesian combination method for short-term traffic flow forecasting. Transportation Research Part C: Emerging Technologies. 2014;43:79-94.

[60] Wang GC. How to handle multicollinearity in regression modeling. The Journal of Business Forecasting. 1996;15(1):23.

[61] Wardrop JG. ROAD PAPER. SOME THEORETICAL ASPECTS OF ROAD TRAFFIC RESEARCH. Proceedings of the institution of civil engineers. 1952;1(3):325-62.

[62] Williams B. Multivariate vehicular traffic flow prediction: evaluation of ARIMAX modeling. Transportation Research Record: Journal of the Transportation Research Board. 2001(1776):194-200.

[63] Wold S, Esbensen K, Geladi P. Principal component analysis. Chemometrics and intelligent laboratory systems. 1987;2(1-3):37-52.

[64] Wu Y-J, Chen F, Lu C-T, Yang S. Urban traffic flow prediction using a spatiotemporal random effects model. Journal of Intelligent Transportation Systems. 2016;20(3):282-93.

[65] Xia D, Wang B, Li H, Li Y, Zhang Z. A distributed spatialtemporal weighted model on MapReduce for short-term traffic flow forecasting. Neurocomputing. 2016;179:246-63.

[66] Xu Y, Kong Q-J, Liu Y, editors. Short-term traffic volume prediction using classification and regression trees. Intelligent Vehicles Symposium (IV), 2013 IEEE; 2013: IEEE.

[67] Zhang N, Wang F-Y, Zhu F, Zhao D, Tang S. DynaCAS: Computational experiments and decision support for ITS. IEEE Intelligent Systems. 2008;23(6).

Appendix: Matlab Code

```

%% Mesele Atsbeha Gebresilassie
%% Royal Institute of Technology - KTH
% School of Architecture and The Build Environment
% Stockholm, Sweden
%=======================================================
%% Master Thesis on:
%% Spatio-temporal Traffic Flow Prediction and Modeling
%=======================================================
function main()

disp('===================================================');
fprintf('\n TRAFFIC FLOW PREDICTION MODELS: \n \n 1. LR-IS \n 2. JLR \n 3. JLR-PCA \n 4. SWR \n 5. SVMR_IS \n 6. JSVMR \n 7. JSVMR-PCA \n 8. SWSVMR');
mdl_choice = input(' \n SELECT MODEL TO RUN: ','s');
switch (mdl_choice)
    case '1'
        fprintf('\n To run LR-IS, press ENTER:')
        [RMSE] = LR_IS();
    case '2'
        fprintf('\n To run JLR, press ENTER:')
        [err] = JLR_Pr();
    case '3'
        fprintf('\n To run JLR-PCA, press ENTER:')
        [err] = JLR_PCA_Pr();
    case '4'
        fprintf('\n To run SWR, press ENTER:')
        [err] = Spat_wgt_Reg();
    case '5'
        fprintf('\n To run SVMR_IS, press ENTER:')
        [svmr_err] = SVMR_IS();
    case '6'
        fprintf('\n To run SVMR_JLR, press ENTER:')
        [JSVMR_err] = JSVMR();
    case '7'
        fprintf('\n To run SVMR_JLR-PCA, press ENTER:')
        [err] = JSVMR_PCA();
    case '8'
        fprintf('\n To run SWSVMR, press ENTER:')
        [SWSVMR_err] = ST_SVMR();
    otherwise
        fprintf('\n WRONG MODEL SELECTED: ')
        disp(mdl_choice);
end
end
```

```

% Function that reads the "Training, Testing and Baseline_Solution Datasets
function [Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
format short
%   Read the "Baseline_solution dataset"
fid_base = fopen('BaseTxtFile.txt');
flow_baseline = textscan(fid_base, '%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f');
flow_baseline = [flow_baseline{1} flow_baseline{2} flow_baseline{3} ...
    flow_baseline{4} flow_baseline{5} flow_baseline{6} flow_baseline{7} ...
    flow_baseline{8} flow_baseline{9} flow_baseline{10} flow_baseline{11} ...
    flow_baseline{12} flow_baseline{13} flow_baseline{14} flow_baseline{15} ...
    flow_baseline{16} flow_baseline{17} flow_baseline{18} flow_baseline{19} flow_baseline{20}];
fclose(fid_base);

% Read the "Test_dataset"
fid_test =fopen('test.txt');
flow_test = textscan(fid_test, '%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f');
flow_test = [flow_test{1} flow_test{2} flow_test{3} flow_test{4} flow_test{5} ...
    flow_test{6} flow_test{7} flow_test{8} flow_test{9} flow_test{10} ...
    flow_test{11}  flow_test{12} flow_test{13} flow_test{14} flow_test{15} ...
    flow_test{16} flow_test{17} flow_test{18} flow_test{19} flow_test{20}];
fclose(fid_test);

```

```
% Read the "Training_dataset"
fid_training = fopen('training.txt');
Trainingdata = textscan(fid_training,'%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f');
Trainingdata = [Trainingdata{1} Trainingdata{2} Trainingdata{3} ...
    Trainingdata{4} Trainingdata{5} Trainingdata{6} Trainingdata{7} ...
    Trainingdata{8} Trainingdata{9} Trainingdata{10} Trainingdata{11} ...
    Trainingdata{12} Trainingdata{13} Trainingdata{14} Trainingdata{15} ...
    Trainingdata{16} Trainingdata{17} Trainingdata{18} Trainingdata{19} Trainingdata{20}];
fclose(fid_training);

% Reads the "Network Spatial Weights" Generated from ArcGIS "Generate
% Network Spatial Weights" based on wij = exp(-d^2/h^2)
fid_weight = fopen('Network_weights.txt');
SensorWeights = textscan(fid_weight,'%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f');
SensorWeights = [SensorWeights{1} SensorWeights{2} SensorWeights{3} SensorWeights{4} ...
    SensorWeights{5} SensorWeights{6} SensorWeights{7} SensorWeights{8} SensorWeights{9} ...
    SensorWeights{10} SensorWeights{11} SensorWeights{12} SensorWeights{13} SensorWeights{14} ...
    SensorWeights{15} SensorWeights{16} SensorWeights{17} SensorWeights{18} SensorWeights{19} SensorWeights{20}];
fclose(fid_weight);
end
```

```
function [RMSE] = LR_IS()
LR_Pred_result = zeros(1000,20);
rmse = zeros(20,1);
R_sqr = zeros(20,1);
LR_IS_Residulas = zeros(1000,20);
%% MODEL TRAINING, EVALUATION AND PREDICTION
for i = 1:20
    % Decompose the time-series
    [hourlyTraffic] = decomposeTraffic(i);
    % Get the features both 'Predictors' and 'Response'
    [x,y]  = extractFeatures(hourlyTraffic);
    % execute the regression model - linear fit
    LR_Mdl = fitlm(x,y,'linear','RobustOpts','on');
    % Testing
    [x_test]  = extractTestFeatures(i);
    LR_Pred_result(:,i) = predict(LR_Mdl,x_test);
    % Model evaluation in RMSE
    rmse(i,1) = LR_Mdl.RMSE;
    % Model R-Square results
    R_sqr(i,1) = LR_Mdl.Rsquared.Ordinary*100;
    % Model Residual diagnosis
    LR_IS_Residulas(:,i) = LR_Mdl.Residuals.Raw;
end
%% MODEL ANALYSIS
% Baseline solution analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');

% LR-IS Model: Results analysis
figure()
histogram(LR_IS_Residulas);
xlabel('Residuals');
ylabel('Frequency');
figure()
plotResiduals(LR_Mdl,'probability')
title('');

LR_IS = [rmse R_sqr];
disp('... RMSE ... R-Square');
disp(LR_IS);
disp('The overall average RMSE');
```

```
RMSE = mean(rmse);
disp('Maximum flow predicted');
max_flow = max(LR_Pred_result)
disp('Minimum flow predicted');
min_flow = min(LR_Pred_result)
disp('Average flow predicted');
avg_flow = mean(LR_Pred_result)
figure()
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('LR-IS: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off

%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('LR-IS_test_prediction.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile,'%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t  %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',LR_Pred_result);
fclose(preFile);
end
```

```
function [err] = JLR_Pr()
coeff = zeros(301,20); % Parameters for each sensor
rmse = zeros(20,1); % RMSE for each sensor
JLR_input = ones(1000,1); % Regression input
JLR_Predion = zeros(1000,20); % Predicted results for each sensor
R_sqr = zeros(20,1); % R-Square for each sensor
tst_input = ones(1000, 1); % Joint test data input
resp_y = zeros(1000,20); % Response variable
JLR_residuals = zeros(1000,20); % Model residuals
%% FEATURES CPMSTRUCTION - JOINT
for i = 1:20
    [hourlyTraffic] = decomposeTraffic(i);
    [x,y]  = extractFeatures(hourlyTraffic);
```

```matlab
    JLR_input = [JLR_input x];
end
%% RESPONSE VARIABLE EXTRACTION
for j = 1:20
    [y] = extractY(j);
    resp_y(:,j) = y;
end
%% TEST VARIABLES EXTRACTION – JOINT
for t = 1:20
    [x_test]  = extractTestFeatures_joint(t);
    tst_input = [tst_input x_test];
end
%% REGRESSION: LEARNING, VALIDATION and TESTING
for rg = 1:20
    % Regression model 'fitlm' ~ learning
    JLR_Mdl = fitlm(JLR_input(:,2:301),resp_y(:,rg),'linear','RobustOpts','on')

    % Model validation 'rmse' ~ Validation
    rmse(rg,1) = JLR_Mdl.RMSE;
    % Parameters estimated ~ variables' coefficients
    coeff(:,rg) = JLR_Mdl.Coefficients.Estimate;
    % Models evaluation 'R–Square' ~ power of expression
    R_sqr(rg,1) = JLR_Mdl.Rsquared.Ordinary*100;
    % Model prediction errors ~ Residuals
    JLR_residuals(:,rg) = JLR_Mdl.Residuals.Raw;

    % Model testing 'predict' ~ predicting
    JLR_Predion(:,rg) = predict(JLR_Mdl,tst_input(:,2:301));
end
err = mean(rmse);
%% MODEL ANALYSIS
% Baseline solution analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');

% RMSE and R-Square in each sensor
disp('...RMSE ... R-Square');
JLR = [rmse R_sqr];
disp(JLR)
disp('The overall average RMSE');
disp(err);
figure()
histogram(JLR_residuals);
xlabel('Residuals');
ylabel('Frequency');
figure()
plotResiduals(JLR_Mdl,'probability')
title('');

disp('Maximum flow predicted');
max_flow = max(JLR_Predion)
disp('Minimum flow predicted');
min_flow = min(JLR_Predion)
disp('Average flow predicted');
avg_flow = mean(JLR_Predion)
figure()
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('JLR: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off
%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('JLR_Results_16_03.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile, '%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',JLR_Predion);
```

```
fclose(preFile);
end
```

```
function [err] = JLR_PCA_Pr()
coeff = zeros(11,20); % Parameters for each sensor
rmse = zeros(20,1); % RMSE for each sensor
JLR_input = ones(1000,1); % Regression input
JLR_PCA_Predion = zeros(1000,20); % Predicted results for each sensor
R_sqr = zeros(20,1); % R-Square for each sensor
tst_input = ones(1000, 1); % Joint test data input
resp_y = zeros(1000,20); % Response variable extracted
JLR_residuals = zeros(1000,20); % Residuals

%% PREDICTORS CONSTRUCTION - JOINT
for i = 1:20
    [hourlyTraffic] = decomposeTraffic(i);
    [x,y] = extractFeatures(hourlyTraffic);
    JLR_input = [JLR_input x];
end
% APPLYING 'pca' ON INPUT FEATURES
[coeff_in,score_in,latent_in,tsquared_in] = pca(JLR_input(:,2:301));
pca_input = JLR_input(:,2:301)*coeff_in;
% selects the top 15 principal components
input_set = pca_input(:,1:10);

%% EXTRACT THE RESPONSE VARIABLE
for j = 1:20
    [y] = extractY(j);
    resp_y(:,j) = y;
end
%% TEST FEATURES CONSTRUCTION
for t = 1:20
    [x_test]  = extractTestFeatures_joint(t);
    tst_input = [tst_input x_test];
end
% APPLYING 'pca' ON TEST FEATURES
[coeff_tst,score_tst,latent_tst,tsquared_tst] = pca(tst_input(:,2:301));
pca_tst = tst_input(:,2:301)*coeff_tst;
% Selects the top 15 principal components
pca_test_set = pca_tst(:,1:10);
```

```

%% MODEL EXECUTION, EVALUATIONS AND PREDICTION
for rg = 1:20
    % Model building: ~ Learning
    JLR_Mdl_pca = fitlm(input_set,resp_y(:,rg),'linear','RobustOpts','on');

    % Model evalutation: ~ validation
    % RMSE for individual sensors
    rmse(rg,1) = JLR_Mdl_pca.RMSE;
    % R-Squared for indvidual sensors
    R_sqr(rg,1) = JLR_Mdl_pca.Rsquared.Ordinary*100;
    % Residuals of estimation
    JLR_residuals(:,rg) = JLR_Mdl_pca.Residuals.Raw;
    % Coefficients estimated
    coeff(:,rg) = JLR_Mdl_pca.Coefficients.Estimate;

    % Model Testing ~ Prediction
    JLR_PCA_Predion(:,rg) = predict(JLR_Mdl_pca, pca_test_set);
end
%% MODEL SUMMARY
err = rmse;

% Baseline solution analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');
hold off
disp('The average RMSE over all sensors');
average_rmse = mean(err);
disp(average_rmse);
figure()
```

```
% Illustrate the residuals using histogram
histogram(JLR_residuals);
xlabel('Residuals');
ylabel('Frequency')

% R-Square report
disp('The R-Square value of each sensor');
disp(R_sqr);

figure()
plotResiduals(JLR_Mdl_pca,'probability')
title('');
JLR_PCA = [rmse R_sqr];
disp('... RMSE ... R-Square');
disp(JLR_PCA);
disp('The overall average RMSE');
RMSE = mean(rmse);

disp('Maximum flow predicted');
max_flow = max(JLR_PCA_Predion)
disp('Minimum flow predicted');
min_flow = min(JLR_PCA_Predion)
disp('Average flow predicted');
avg_flow = mean(JLR_PCA_Predion)
figure()
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('JLR-PCA: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off
%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('JLR_PCA_Results.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile, '%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',JLR_PCA_Predion);
fclose(preFile);
```

```
end
```

```
function [err] = Spat_wgt_Reg()
format long
rmse = zeros(20,1);
r_Sqare = zeros(20,1);
respons_var = zeros(1000,20);
Spat_Wgt_residuals = zeros(1000,20);
coeff = zeros(301,20);
gwr_R_Pred = zeros(1000,20);
testing = ones(1000,1);

%% RESPONSE VARIABLE EXTRACTION
for resp = 1:20
    [y] = extractY(resp);
    respons_var(:,resp) = y;
end
%% TEST DATA EXTRACTION AND WEIGHTING
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
% applying the weights
for j = 1:20
    for wgt = 1:20
        if wgt == j
            tmp_wgt = 1;
        else
            tmp_wgt = SensorWeights(wgt,j);
        end
        [x_test]  = extractTestFeatures(j);
        tmp = (tmp_wgt*x_test) +x_test;
    end
    testing = [testing tmp];
end
%% MODEL TRAINING, EVALUATION AND PREDICTION
for i = 1:20
    % Get the input from all sensors
    [JLR_input, y_response] = gwr_JRinput(i);
    % run fitlm: learning
    Mdl_gwr_R = fitlm(JLR_input, respons_var(:,i), 'linear','RobustOpts','on')
    % model evaluation: validating
    % Models RMSE
    rmse(i,1) = Mdl_gwr_R.RMSE;
    % R-Square
    r_Sqare(i,1) = Mdl_gwr_R.Rsquared.Ordinary*100;
    % Model coefficient estimates
    coeff(:,i) = Mdl_gwr_R.Coefficients.Estimate;
    % Residuals
    Spat_Wgt_residuals(:,i) = Mdl_gwr_R.Residuals.Raw;
    % model testing: predicting
    gwr_R_Pred(:,i) = predict(Mdl_gwr_R,testing(:,2:301));
end
%% MODEL SUMMARY
% Residual analysis
figure()
histogram(Spat_Wgt_residuals);
xlabel('Residuals');
ylabel('Frequency');
% Root Mean Square Error
RMSE = mean(rmse)
disp('The individual RMSE');
disp(rmse);
figure()
%plot of residual probability
plotResiduals(Mdl_gwr_R, 'probability');
title('')
% R-Square
R_sqr = r_Sqare;
disp('The individual R-Square results');
disp(R_sqr);

baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');

disp('Maximum flow predicted');
max_flow = max(gwr_R_Pred)
disp('Minimum flow predicted');
```

```matlab
min_flow = min(gwr_R_Pred)
disp('Average flow predicted');
avg_flow = mean(gwr_R_Pred)
figure()
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('Spt.Wgt.R: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off
%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('File20032017_weighted.txt', 'w');
% Prints 'MLR_Prediction_Reults' into file using tab separated format
fprintf(preFile, '%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',gwr_R_Pred);
fclose(preFile);
end
```

```matlab
function [svmr_err] = SVMR_IS()
format long
% Prediction results
SVMR_Pred_result = zeros(1000,20);
% Training error
rmse = zeros(20,1);

%% MODEL BUILDING
for i = 1:20
    % Decompose the time-series
    [hourlyTraffic] = decomposeTraffic(i);
    % Get the features both 'Predictors' and 'Response'
    [x,y]  = extractFeatures(hourlyTraffic);
    % execute the regression model - Gaussian Kernel
    SVMR_IS_Mdl = fitrsvm(x,y,'KernelFunction','gaussian',...
        'KernelScale','auto','Standardize',true);
    % Number of iteration at which the model converges
    SVMR_IS_Mdl.ConvergenceInfo.Converged;
    % K-FOLD CROSS VALIDATION; k = 10
    CVMdl = crossval(SVMR_IS_Mdl);
    loss = sqrt(kfoldLoss(CVMdl));
    rmse(i,1)= loss;
    % Get the test data for 'Individual Sensors'
    [x_test]  = extractTestFeatures(i);
    % Make the prediction
    SVMR_Pred_result(:,i) = predict(SVMR_IS_Mdl,x_test);
end
%% MODEL SUMMARY

[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');
hold off
disp('The individual sensors RMSE');
disp(rmse);
disp('Average RMSE');
svmr_err = mean(rmse);
disp('Maximum flow predicted');
max_flow = max(SVMR_Pred_result)
disp('Minimum flow predicted');
min_flow = min(SVMR_Pred_result)
disp('Average flow predicted');
avg_flow = mean(SVMR_Pred_result)
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('SVMR-IS: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','
Baseline Average flow');
hold off
%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('SVMR_IS_Result_03_17.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile,'%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t  %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',SVMR_Pred_result);
fclose(preFile);
end
```

```
function [JSVMR_err] = JSVMR()
JLR_input = ones(1000,1); % Regression input
JLR_Predion = zeros(1000,20); % Predicted results for each sensor
tst_input = ones(1000, 1); % Joint test data input
resp_y = zeros(1000,20); % Response variable
rmse= zeros(20,1);

%% Extract PREDICTORS, RESPONSE AND TEST
for i = 1:20
    [hourlyTraffic] = decomposeTraffic(i);
    [x,y] = extractFeatures(hourlyTraffic);
    JLR_input = [JLR_input x];
end
%% Get all RESPONSE values for each sensor
for j = 1:20
    [y] = extractY(j);
    resp_y(:,j) = y;
end
 %% get all the test data -JOINT
 for t = 1:20
    [x_test]  = extractTestFeatures_joint(t);
    tst_input = [tst_input  x_test];
 end

%% REGRESSION: Learning, Validation & Testing
for rg = 1:20
    % Regression'fitRSVM' ~ NON-LINEAR Learning - uses GUASSIAN kernel
    JSVMR_Mdl = fitrsvm(JLR_input(:,2:301),resp_y(:,rg),'KernelFunction',...
        'gaussian', 'KernelScale','auto','Standardize',true);
    % Model testing 'predict' ~ PREDICTION
    JLR_Predion(:,rg) = predict(JSVMR_Mdl,tst_input(:,2:301));
    % iterating towards convergence
    JSVMR_Mdl.ConvergenceInfo.Converged;
    % k = 10: k-fold validation
    CVMdl = crossval(JSVMR_Mdl);
    loss = sqrt(kfoldLoss(CVMdl));
    rmse(rg,1)= loss;
end

%% MODEL SUMMARY
% Baseline solution Analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');
hold off

% JSVM-R model results analysis
disp('The individual sensor RMSE');
disp(rmse);
disp('The average RMSE');
JSVMR_err = mean(rmse);
disp('Maximum flow predicted');
max_flow = max(JLR_Predion)
disp('Minimum flow predicted');
min_flow = min(JLR_Predion)
disp('Average flow predicted');
avg_flow = mean(JLR_Predion)
plot(max_flow);
hold on
```

```
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('JSVM-R: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off

%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('JSVMR_Results_17_03.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile, '%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',JLR_Predion);
fclose(preFile);
end
```

```
function [err] = JSVMR_PCA()
JLR_input = ones(1000,1); % Regression input
JLR_PCA_Predion = zeros(1000,20); % Predicted results for each sensor
tst_input = ones(1000, 1); % Joint test data input
resp_y = zeros(1000,20); % Response variable
rmse= zeros(20,1);

%% PREDICTORS CONSTRUCTION
for i = 1:20
    [hourlyTraffic] = decomposeTraffic(i);
    [x,y] = extractFeatures(hourlyTraffic);
    JLR_input = [JLR_input x];
end
%% APPLY 'pca' ON INPUT FEATURES
[coeff_in,score_in,latent_in,tsquared_in] = pca(JLR_input(:,2:301));
pca_input = JLR_input(:,2:301)*coeff_in;
% Select the top principal components that yield reasonably min error
input_set = pca_input(:,1:14);

%% EXTRACT THE RESPONSE VARIABLE FOR EACH SENSOR
for j = 1:20
```

```
    [y] = extractY(j);
    resp_y(:,j) = y;
end

%% EXTRACT TEST FEATURES
for t = 1:20
    [x_test]  = extractTestFeatures_joint(t);
    tst_input = [tst_input x_test];
end
%% APPLY 'pca' ON TEST FEATURES
[coeff_tst,score_tst,latent_tst,tsquared_tst] = pca(tst_input(:,2:301));
pca_tst = tst_input(:,2:301)*coeff_tst;
% Select the top principal components same us in the training
pca_test_set = pca_tst(:,1:14);

%% REGRESSION Model
for rg = 1:20
    % Model building: ~ Learning
    JSVMR_Mdl_pca = fitrsvm(input_set,resp_y(:,rg),'KernelFunction','gaussian',...
          'KernelScale','auto','Standardize',true);
    % Model Testing ~ Prediction
    JLR_PCA_Predion(:,rg) = predict(JSVMR_Mdl_pca,pca_test_set);
    % iterations at which model converged
    JSVMR_Mdl_pca.ConvergenceInfo.Converged;
    % k = 10, k-fold cross validation
    CVMdl = crossval(JSVMR_Mdl_pca);
    loss = sqrt(kfoldLoss(CVMdl));
    rmse(rg,1)= loss;
end
%% MODEL ANALYSIS

% Baseline solution Analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
```

```
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');
hold off

disp('The RMSE of individual sensors');
disp(rmse);
disp('Average RMSE');
err = mean(rmse);
disp('Maximum flow predicted');
max_flow = max(JLR_PCA_Predion)
disp('Minimum flow predicted');
min_flow = min(JLR_PCA_Predion)
disp('Average flow predicted');
avg_flow = mean(JLR_PCA_Predion)
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('JSVMR-PCA: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off
%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('JSVMR_PCA_Results.txt', 'w');
% Prints 'MLR_Prediction_Results' into file using tab separated format
fprintf(preFile, '%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',JLR_PCA_Predion);
fclose(preFile);
end
```

```
function [ST_SVMR_err] = ST_SVMR()
respons_var = zeros(1000,20);
testing = ones(1000,1);
rmse= zeros(20,1);
ST_SVMR_Pred=zeros(1000,20);
```

```

%% RESPONSE VARIABLES
for resp = 1:20
    [y] = extractY(resp);
    respons_var(:,resp) = y;
end
yy= respons_var(1:20,1);
%% LOAD AND EXTRACT WEIGHTS
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
% Extract test data
for j = 1:20
    for wgt = 1:20
        if wgt == j
            tmp_wgt = 1;
        else
            tmp_wgt = 10000*SensorWeights(wgt,j);
        end
        [x_test]  = extractTestFeatures(j);
        tmp = (tmp_wgt*x_test)+x_test;
    end
    testing = [testing tmp];
end

%% TRAINING AND TESTING
for i = 1:20
    % Get the input from all sensors
    [JLR_input, y_response] = gwr_JRinput(i);
    % Run the 'fitrsvm'
    ST_SVMR_Mdl = fitrsvm(JLR_input, respons_var(:,i), 'KernelFunction','gaussian',...
        'KernelScale','auto','Standardize',true);
    % Test 'predict'
    ST_SVMR_Pred(:,i)= predict(ST_SVMR_Mdl,testing(:,2:301));
    % itetation towards convergence
    ST_SVMR_Mdl.ConvergenceInfo.Converged;
    % k = 10 k-fold cross validation
    CVMdl = crossval(ST_SVMR_Mdl);
    loss = sqrt(kfoldLoss(CVMdl));
    rmse(i,1)= loss; % RMSE
end

```

```
%% MODEL SUMMARY
disp('The individual RMSE of Sensors');
disp(rmse);
disp('Average RMSE');
ST_SVMR_err = mean(rmse);
disp(ST_SVMR_err);

% Baseline solution Analysis
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
baseline_Max = max(flow_baseline);
baseline_Min = min(flow_baseline);
baseline_Average = mean(flow_baseline);
figure();
plot(baseline_Max);
hold on
plot(baseline_Min);
plot(baseline_Average);
xlabel('Sensors');
ylabel('Baseline: Traffic flow');
legend('Maximum flow','minimum flow','Average flow');
hold off

% Prediction results analysis ST_SVMR
disp('Maximum flow predicted');
max_flow = max(ST_SVMR_Pred);
disp('Minimum flow predicted');
min_flow = min(ST_SVMR_Pred);
disp('Average flow predicted');
avg_flow = mean(ST_SVMR_Pred);
plot(max_flow);
hold on
plot(min_flow)
plot(avg_flow)
plot(baseline_Average,'-r','LineWidth',2.5);
xlabel('Sensors');
ylabel('ST-SVMR: Traffic flow');
legend('Maximum flow','minimum flow','Average flow','Baseline Average flow');
hold off

%% PRINTING THE PREDICTION RESULTS INTO FILE
preFile = fopen('ST_SVMR_Pred_17_03.txt', 'w');
```

```
% Prints 'MLR_Prediction_Reults' into file using tab separated format
fprintf(preFile,'%f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t  %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f\t %f \n',ST_SVMR_Pred);
fclose(preFile);
end
```

```
% Extract Response Values
function[y] = extractY(sensor)

y_tmp = zeros(1000,1);
% Get y value/ the response variable for each sensor passed
t = 30;
%Get the decomposed sensor
[hourlyTraffic] = decomposeTraffic(sensor);
% extract all the training dependent values
for i = 1:1000
    y_tmp(i,1) = sum(hourlyTraffic(i,t+11:t+20));
end
y = y_tmp;
end
```

```
function[x_test_matrix]  = extractTestFeatures(s)

[hourlyTestTraffic] = decomposeTest(s);

% temp input matrix
inputMatrix = zeros(1000,15);
t = 30; % window size
% the passed test set
testData = hourlyTestTraffic;

for i = 1:1000
    % Average traffic (i.e. mean) in TEST DATA
    meana1 = mean(testData(i,t-4:t));
    meana2 = mean(testData(i,t-9:t));
    meana3 = mean(testData(i,t-19:t-10));
    meana4 = mean(testData(i,t-29:t-20));
    meana5 = mean(testData(i,t-29:t));

    % Rate of change of traffic (roc) in TEST DATA
    roca1 = (testData(i,t-4)- testData(i,t))/(5);
    roca2 = (testData(i,t-9)- testData(i,t))/(10);
    roca3 = (testData(i,t-19)- testData(i,t-10))/(10);
    roca4 = (testData(i,t-29)- testData(i,t-20))/(10);
    roca5 = (testData(i,t-29) - testData(i,t))/(30);

    % Standard deviaion of traffic (i.e. std) in TEST DATA
    stda1 = std(testData(i,t-4:t));
    stda2 = std(testData(i,t-9:t));
    stda3 = std(testData(i,t-19:t-10));
    stda4 = std(testData(i,t-29:t-20));
    stda5 = std(testData(i,t-29:t));

    % Fill into the input TEST MATRIX
    inputMatrix(i,:) = [meana1, meana2, meana3, meana4, meana5, ...
        roca1, roca2, roca3, roca4, roca5,...
        stda1, stda2, stda3, stda4, stda5];

end % end 'for i = 1:1000 loop'
x_test_matrix = inputMatrix; % FUNCTION RETURN VALUE
end % END OF FUNCTION
```

```
function[x_test_matrix] = extractTestFeatures_joint(s);
[hourlyTestTraffic] = decomposeTest(s);

% temp input matrix
inputMatrix = zeros(1000,15);
t = 30; % window size
% the passed test set
testData = hourlyTestTraffic;

for i = 1:1000
    % Average traffic (i.e. mean) in TEST DATA
    meana1 = mean(testData(i,t-4:t));
    meana2 = mean(testData(i,t-9:t));
    meana3 = mean(testData(i,t-19:t-10));
    meana4 = mean(testData(i,t-29:t-20));
    meana5 = mean(testData(i,t-29:t));

    % Rate of change of traffic (roc) in TEST DATA
    roca1 = (testData(i,t-4)- testData(i,t))/(5);
    roca2 = (testData(i,t-9)- testData(i,t))/(10);
    roca3 = (testData(i,t-19)- testData(i,t-10))/(10);
    roca4 = (testData(i,t-29)- testData(i,t-20))/(10);
    roca5 = (testData(i,t-29) - testData(i,t))/(30);

    % Standard deviaion of traffic (i.e. std) in TEST DATA
    stda1 = std(testData(i,t-4:t));
    stda2 = std(testData(i,t-9:t));
    stda3 = std(testData(i,t-19:t-10));
    stda4 = std(testData(i,t-29:t-20));
    stda5 = std(testData(i,t-29:t));


    % Fill into the input TEST MATRIX
    inputMatrix(i,:) = [meana1, meana2, meana3, meana4, meana5, ...
        roca1, roca2, roca3, roca4, roca5,...
        stda1, stda2, stda3, stda4, stda5];

end % end 'for i = 1:1000 loop'
x_test_matrix = inputMatrix; % FUNCTION RETURN VALUE
end % END OF FUNCTION
```

```
function[x_train_matrix, y_train_vector] = extractFeatures(s);
inputMatrix = zeros(1000,15);
y = zeros(1000,1);
t = 30; % window size

% the passed training set in the k-fold
newTraining = s;

% NOTE: for intervals t = 30 (i.e. first half hour)
% a1 = [t-5,t] --> last 5 minute
% a2 = [t-10,t] --> last 10 minute
% a3 = [t-20,t-10] --> Middle minute
% a4 = [t-30,t-20] --> First 10 minute
% a5 = [t-30,t] --> The whole window of 30 minute
for i=1:1000
    % Average traffic (i.e. mean)
    meana1 = mean(newTraining(i,t-4:t));
    meana2 = mean(newTraining(i,t-9:t));
    meana3 = mean(newTraining(i,t-19:t-10));
    meana4 = mean(newTraining(i,t-29:t-20));
    meana5 = mean(newTraining(i,t-29:t));

    % Rate of change of traffic (roc)
    roca1 = (newTraining(i,t-4)- newTraining(i,t))/(5);
    roca2 = (newTraining(i,t-9)- newTraining(i,t))/(10);
    roca3 = (newTraining(i,t-19)- newTraining(i,t-10))/(10);
    roca4 = (newTraining(i,t-29)- newTraining(i,t-20))/(10);
    roca5 = (newTraining(i,t-29) - newTraining(i,t))/(30);

    % Standard deviaion of traffic (i.e. sdv)
    stda1 = std(newTraining(i,t-4:t));
    stda2 = std(newTraining(i,t-9:t));
    stda3 = std(newTraining(i,t-19:t-10));
    stda4 = std(newTraining(i,t-29:t-20));
    stda5 = std(newTraining(i,t-29:t));

    % Fill into the input vector (iVector)
    inputMatrix(i,:) = [meana1, meana2, meana3, meana4, meana5,...
        roca1, roca2, roca3, roca4, roca5,stda1, stda2, stda3,...
        stda4, stda5];

    y(i,1) = sum(newTraining(i,t+11:t+20));

end % end 'for i = 1:1000 loop'
x_train_matrix = inputMatrix;
y_train_vector = y;
end % end of 'extractFeature()' function
```

```
function[hourlyTraffic] = decomposeTraffic(sensor);

% By calling the 'readFiles()' function, training data is loaded
[Trainingdata, Testdata, Baselinedata] = readFiles();

% To navigate down along the dataset of a station, set the start and window
% into 1 and 60 respectively
start = 1;
win = 60;
```

```
hrs = zeros(60,1000);
%Navigate down a station's data and split into an hour long colomns
for col = 1:length(Trainingdata(:,sensor)) % which is 60k
    hrs(:,col) = Trainingdata(start:start+59,sensor);
    if(col < 1000)
        start = start + win;
    else
        hourlyTraffic = hrs'; % so, hourlyTraffic - matrix of 60 x 1000 is transposed & returned
        break;
    end % end of 'col<2000 if'
end % end of 'col - for loop'
end % end of function 'decomposeintohours()'
```

```
function[hourlyTestTraffic] = decomposeTest(sensor);
% By calling the 'readFiles()' function, test data data is loaded
[Trainingdata, Testdata, Baselinedata] = readFiles();

% To navigate down along the dataset of a station, set the start and window
% into 1 and 30 respectively
start = 1;
win = 30;
hrs = zeros(30,1000);
% Navigate down a station's
% data and split into a half an hour long colomns
for i = 1:length(Testdata(:,sensor))
    hrs(:,i) = Testdata(start:start+29,sensor);
    if(i<1000)
        start = start + win;
    else
        % So, hourly test data - matrix of 60-by-1000 is transposed & returned
        hourlyTestTraffic = hrs';
        break;
    end % end of 'col<1000 if'
end % end of 'col - for loop'
end % end of function 'decomposeTraffictest()
```

```
function [JLR_input, y_response] = gwr_JRinput(current_Sensor)
```

```

% This function extracts relevant model input features from each of the
% sensors, then the inputs will be used in the model
tmp_input = ones(1000,1);

% The Network Spatial Weight for each sensro location is generated based on
% the form ”wij = exp(1− dˆ2/h2)” as generated from ArcGIS. The distance
% used is based on network distance using ” shortest route ”!!
[Trainingdata, flow_test, flow_baseline, SensorWeights] = readFiles();
for sen = 1:20
    % decomposes the time series data into multiple hour long record
    [hourlyTraffic] = decomposeTraffic(sen);
    % returns the features from the ’sensor’
    [x,y]  = extractFeatures(hourlyTraffic);
    % Accomulating features into an input of matrix by weighing each set of
    % variables extracted from a sensor
    if current_Sensor == sen
        tmp_wgt = 1;
    else
        tmp_wgt = 10000*SensorWeights(current_Sensor,sen);
    end
    tmp = (tmp_wgt*x);%+x; % applying the weight to each variable
    tmp_input = [tmp_input tmp];
end
[y_response]= extractY(current_Sensor);
JLR_input = tmp_input(:,2:301);
end
```

TRITA TRITA-GIT EX 17-010
ISSN 1653-5227
ISRN KTH/GIT/EX--17/010-SE

www.kth.se