Kapil Bastola

March 22, 2017

DS 670

Midterm Proposal

## Contribution

There are three important numerical values present in the dataset. One is the time taken to travel between two points, next is the average time it takes the vehicles to travel from one point to another and the third is the number of vehicles that travel between the two points for each reading.

First thing I would like to do with the dataset is to look at the descriptive statistics of the different fields in the dataset. In particular, for the numerical fields like Average time taken, Average speed, or Number of vehicles, I will look at the histograms of each of these field to see the distribution of values. Datasets consists of 9 different columns with a mix of numerical, categorical and date time variables. With SQL and R in zeppelin, I will look at the summary of each variable. For categorical variables I will look at the number of factors or levels in the variables. For the numerical variables I will look at the distribution and variation of each variable along with measures of central tendencies like mean and median. For the numerical variables I will also look at any correlation that might exist between two variables. This will help me in sorting out the variables. I will look at histograms for distribution and scatterplots to evaluate correlations between numerical variables.

In order to perform supervised learning algorithms, we need to train the dataset using labels. In this dataset there are no labels present. Average speed column in the dataset can be used to label the dataset as speed represents how fast the cars are going and this tells us how congested the road is. If we want to predict traffic congestion, velocity of cars in the road can be considered the label we want to predict. Traffic congestion can be broken down in 3 different levels, heavy, mild, and low. For our algorithm, we will assign average speed of less than 30 kilometers per hour (kph) to be heavy traffic congestion, average speed of between 31 and 60 kph to be mild traffic congestion and average speed of 60 kph or higher to be low traffic congestion. Using these labels we can now train our models and predict for any new data.

Since the data we have is time based, we can conduct time-series analysis. In order to do the time series analysis, first we will plot the entire time series to see if there is any trend in the dataset and to check for any seasonality in the dataset. When I look at the time stamp field in the datasets, there are data points for every five seconds. We can use the velocity data to predict future velocity or number of vehicles at a given time using time series models like Auto-Regressive Integrated Moving Average (ARIMA) model. I will use the auto correlation function (ACF) and partial auto correlation function (PACF) to determine which order of auto regression and moving average is required for the model. By predicting velocity at a certain time we can make suggestions as to what time are busy and when to be careful when using the roads.

After building the models we can cross validate the models created using both training and test sets to find the model that performs the best

## State of art

In the paper “Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning,” the authors Lippi et. al. look at what existing traffic flow forecasting methods there are and present a comparison between these methods. The authors also provide two new models using support vector regression that they claim to benefit from traffic flow seasonality and represent compromise between accuracy and efficiency. The limitation of this study is it only looks at the short-term flow and does not have scope out the long-term traffic flow conundrum. [1]

For the road traffic dataset, the velocity and the number of vehicles can be plotted in two-dimensional space. We can then try to group these into different clusters. After the algorithm produces different clusters of the data set it is importance to validate if the results we got is any useful or not. This will also determine whether the value of k we used is relevant or not. According to Steinley and Brusco, the steps of cluster validation are to first determine if the dataset consists of more than one cluster and if so, determine the number of clusters and then interpret, test and validate the results [2]. According to Steinley and Brusco we need to conduct cluster validation in order to determine whether a cluster is good or not. For any cluster analysis we need to test a portion of a dataset to diagnose whether a clustering model represents the entire data set or not. The diagnosis process can involve both subjective as well as objective evaluation. The testing, that uses external validation, are subjective because they require the expertise about the dataset to make judgment while determining a set of theoretical justified covariates and the interpretation of each cluster in the solution of the model will also be subjective because it depends on the understanding of theoretical significance of the variables included in the clustering procedure. To complete the validation process after testing and interpretation is replication. This is the method that tells us whether the clustering result is valid or not and also whether the solution of the clustering can be generalized or not. One of the most widely used replication technique, simply called replication analysis, will check to see if there are any replicable solutions or not. [2]

Clustering using representatives or CURE is a clustering technique that does not necessarily favor spherical shapes and sizes of clusters and tries to incorporate outliers as much as possible to the clusters. This helps in clustering when the clusters are not uniform size or shape. [3] According to the authors Guha, et. al., CURE method is able to incorporate the outliers by representing each cluster by generating certain number of fixed points by selecting well scattered points and then shrinking these points toward a center of the group by a fraction. This allows there to be more than one representative for a cluster unlike the k-means, where each cluster only has one representative point.

Traffic forecasting can be a good way for a city to manage the situation of traffic in the roads. Short term forecasting can be achieved using many parametric and non-parametric techniques but ARIMA has been found to be the most accurate method to forecast traffic flow according to the referenced paper by Kumar and Vanajashi. [4] The paper mentions that predicting of traffic flow in short term based on past traffic data is an important element of an intelligent transportation system (ITS). Short term traffic flow forecasting involves predicting traffic volume in the next time interval in the range of five minutes to one hour and this has been researched extensively in the past. The number of vehicles in the road or the traffic flow is a random process. The statistical techniques used to predict traffic flow can be both parametric and non-parametric. The non-parametric techniques like non-parametric regression and neural networks include both descriptive and inferential statistics while the parametric techniques like linear and non-linear regression, historical average algorithms, smoothing techniques, and autoregressive linear processes assume that the dataset has a distribution with stable set of parameters. The paper suggests that the time series analysis based techniques like the autoregressive integrated moving average (ARIMA) is one of the most precise methods for the prediction of traffic flow when compared to other available techniques mentioned above. [4]

The paper called Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches by Smith and Demetsky affirm the notion by Kumar and Vanajashi that predicting traffic flow plays an important part in any intelligent highway or transportation system. [5] Smith and Demetsky confer that nearest neighbor models are well suited for this type of traffic flow prediction and will provide models that are very accurate and portable. They also argue that field personnel who will be using the output from the model in day-to-day basis can easily understand the non-parametric models using nearest neighbors. [6]

In IEEE paper called Traffic flow data forecasting based on interval type-2 fuzzy sets theory, the authors Li et. al., suggest a method of forecasting traffic flow data based on fuzzy type-2 sets theory. Since type -2 fuzzy sets have advantages in modeling unpredictability because of their membership functions being fuzzy, the confidence interval data retain the randomness of the traffic flow while reducing the noise from the detection data. The proposed scheme gets not only the traffic flow forecasting result but can also show the possible range of traffic flow variation with high precision using upper and lower limit forecasting result. The paper shows the effectiveness of the proposed model by using the application on a sample event. The limitation of this paper is that the scope of this paper is limited to only one method and there is no comparison with other methods to see if there are other methods that are better. [8]

In IEEE paper called “Traffic Flow Forecasting for Urban Work Zones,” authors Hou et. al., suggest that not much research has been done on traffic flows in work zones and propose four models to forecast traffic flows for city work zones. The four models are random forest, regression tree, multilayer feed forward neural network, and nonparametric regression. In this paper, authors do both long tern and short term forecasting. Data used were from work zone events in two places, one in a freeway and another in a signalized artery in St. Louis, MO. The results from the four models were compared and it was found that the result from random forest had the most accurate long term and short-term work zone traffic flow forecast. The limitation of this paper is that it only looks at one city to generalize about all urban areas. [9]

In IEEE paper called “Traffic Flow Prediction for Road Transportation Networks With Limited Traffic Data,” authors Abadi et. al., proposes method to predict traffic flow where there are not many sensors and the data is limited. In this paper, the authors use a simulator to generate traffic flows using available traffic information, estimated demand, and historical traffic data available. The model predicts traffic flow up to 30 min ahead using real-time and estimated traffic data. The prediction algorithm is based on an autoregressive model that adapts itself to unpredictable events. This method was used to simulate traffic flow network in San Francisco, CA, and the results demonstrated that the simulations were accurate with error ranging from average of 2% for 5 minutes prediction window and 12% prediction window. This paper’s limitation is that it only looks at the short-term prediction and does not look into long-term simulations. [10]

# In IEEE paper called “Repeatability and Similarity of Freeway Traffic Flow and Long-Term Prediction Under Big Data,” authors Hou and Li develop a long term forecasting method for traffic flow based on repeatability and similarity of traffic flow data. The authors first define the repeatability and similarity of traffic flow data by dividing the traffic series into basis series and deviation series. These definitions are verified using real time data from 102 detecting sites in Shenzhen, China. After the forecasting is done, the effectiveness of the algorithm is verified using real data as well. The limitation of this paper is that the abstract does not exactly explain what the forecasting methodology is and what the data collection method is. [11]

In the paper called “Traffic Flow Prediction With Big Data: A Deep Learning Approach,” authors Lv et. al. propose a new deep learning traffic prediction method that uses spatial and temporal correlations inherently. A stacked auto-encoder model is used to learn generic traffic flow features, and it is trained in a greedy layer-wise fashion. Authors claim that existing models are shallow and are unsatisfactory in real world applications. The paper experimentally demonstrates that the proposed method for traffic flow prediction has superior performance. [12]

# In the paper called “Improving Traffic Flow Prediction With Weather Information in Connected Cars: A Deep Learning Approach,” authors Koesdwiady et. al. elaborate on two objective of the paper; first being investigation between weather and traffic flow and second being prediction of traffic flow with new architecture of data incorporating weather data. There are experimental data in the study from San Francisco, CA that confirm the effectiveness of the proposed approach in relation to current state of the art. I think the limitation of this study is that it only focuses on one other variable, the weather, while there could be other factors apart from weather that could affect the traffic flow and if we consider other factors as well in the study, we might be able to improve prediction models even more. [13]

# In the paper “A Map Reduce-Based Nearest Neighbor Approach for Big-Data-Driven Traffic Flow Prediction,” authors Xia et. al. show a new neural neighbor approach for predicting traffic flow on a Hadoop platform based on correlation analysis. A real-time prediction system is developed including offline distributed training and online parallel prediction. A parallel k-nearest neighbor classifier, which incorporates correlation information, is also built. A new calculation method that combines current data observed in online parallel prediction and classification results from offline distributed training is proposed. Empirical results suggest that this approach outperforms state of art approaches like autoregressive integrated moving average, Naïve Bayes, multilayer perceptron neural networks, and NN regression, in terms of accuracy. Limitation of this method is the utilization of Map Reduce as it is not widely available yet. [14]

In the paper "Bag of Contextual-Visual Words for Road Scene Object Detection From Mobile Laser Scanning Data” the authors Yu et. al. propose a new algorithm to detect road objects like poles, traffic signs, cars, etc. from a 3D mobile laser scanning point cloud data for transportation-related applications. The quantitative evaluation of the study disclosed that recall, precision, quality and F-scores were very high for this study. Compared to existing methods all the aforementioned performance indicators were better using the method used in this study. [15]

# 

In the paper “Accurate and Interpretable Bayesian MARS for Traffic Flow Prediction,” authors Xu et. al. propose an interpretable and adaptable spatiotemporal Bayesian multivariate adaptive-regression splines (ST-BMARS) model to predict short-term freeway traffic flow accurately. Experimental results indicate that the proposed interpretable ST-BMARS model is robust and can generate superior prediction accuracy in contrast with the temporal MARS model, the parametric model autoregressive integrated moving averaging (ARIMA), the state-of-the-art seasonal ARIMA model, and the kernel method support vector regression. The limitation of this study is it only looks at the short-term flow and does not have scope out the long-term traffic flow conundrum. [16]

# In the paper “An Intelligent Particle Swarm Optimization for Short-Term Traffic Flow Forecasting Using on-Road Sensor Systems,” authors Chan et. al. explain that misleading forecasting results are likely to be produced when short-term traffic flow predictors are designed using time-invariant assumptions. To tackle these time-invariant assumptions, an intelligent particle swarm optimization (IPSO) algorithm is proposed to develop short-term traffic flow predictors by integrating the mechanisms of PSO, neural network and fuzzy inference system, to adapt to the time-varying traffic flow characteristics and the time-varying configurations of the on-road sensor systems. Again, only looking at short-term data is the limitation of this study. [17]

# In the paper “On-Road Sensor Configuration Design for Traffic Flow Prediction Using Fuzzy Neural Networks and Taguchi Method” authors Chan and Dillon explain that not all data taken from traffic sensors are useful and we should only select the data that are highly correlated with future traffic flow to predict the traffic flow based on sensor data. In this paper, the Taguchi method, which is a robust and systematic optimization approach for designing reliable and high-quality models, is proposed for determinations of appropriate on-road sensors, in order to capture useful traffic flow conditions for forecasting. The effectiveness of the Taguchi method is demonstrated by developing a traffic flow predictor based on the architecture of fuzzy neural networks. This paper does not look at other methods and comparison between different methods is not performed. [18]

In the paper, “High-Order Gaussian Process Dynamical Models for Traffic Flow Prediction,” the authors Zhao and Sun use the fourth order Gaussian process dynamical model (GPDM) to model traffic flow. This model incorporates weighted k-NN to predict latent variables for efficient prediction. Future flow is estimated by average of results and comparison of performance show that his method provides a significantly better performance compared to other popular methods in traffic flow prediction. [19]

In the paper, “Linguistic Dynamic Analysis of Traffic Flow Based on Social Media—A Case Study,” authors Mo et. al. analyze the congestion time, congested place, and congestion on the traffic police's micro-bo. In addition, the authors build dynamic fuzzy rules on time-varying universe to provide the corresponding traffic management rules. For the case study, this paper uses Shenzhen's traffic police micro-bo to study the information of traffic congestion, including jam session, congestion location and reasons, and disposal methods, and present these results by language form. The limitation from this study is that the results are not very quantitative and are very subjective. [20]

## Data

The dataset consists of data from traffic in a city called Aarhus in Denmark. This dataset is collection of traffic data between two points for certain duration of time in CSV format for different durations. A CSV metadata file is also available that provides additional information regarding the different two points. Upon further analysis it was found that the dataset has 9 columns.

The **status** column has “OK” in all of the first 15 lines that I pulled. I wanted to see if there are any other values for this field. I looked at all the distinct values in the status column and saw that “OK” is the only value in the whole column.

**avgMeasuredTime** column has numerical values. The two sensors on two points of the road measure how long it took a vehicle in seconds to reach the second point from the first point. This field gives us the mean of total time taken in seconds by different vehicles to reach from the first point to the second point for each reading.

**avgSpeed** column also consists of numerical values. This column provides the average speed of vehicles between the two points in kilometer per hour (kmh).

**extID** column consists of 3-4 digit numerical values. Initially I was not certain what this column represented so I looked at all the distinct values in the column. I saw that they are sequential numbers and there are total of 449 distinct values in the table. Since we had 449 total files in our dataset, extID is a unique identifier for each file.

**medianMeasuredTime** column also consists of numerical values and has similar values as the avgMeasuredTime in the first 15 data points we looked at. This column gives us the median of total time taken in seconds by different vehicles to travel between the first and the second point for each reading.

**TIMESTAMP** column consists of date and time values and gives us the date and time of each reading.

**vehicleCount** column consists of numerical values. For each reading there are multiple vehicles passing between the two points. This column gives us the number of vehicles that travel between the two points during the readings.

**\_id** column consists of numerical values as well. In the first 15 rows, there are 6 digit numerical values that are all different. I wanted to count the distinct number of values in the \_id column and saw that the number of distinct values in this column is equal to the count of rows in the table. This suggests that \_id is the unique identifier for each row of data.

**REPORT\_ID** column consists of numerical values as well. In the first 15 rows, all the values were same in this column. Hence, I looked at the count of distinct values and found that there are 449 total unique values in the column. This suggests that this is an identifier for each file as well. In the metadata provided in the website I see that there is a column with same name and the values as the report\_id in the dataset. Hence, report\_id can be used to join the data set with the metadata file to obtain more information on each of the reading that took place.

The metadata file is a single .csv file with more information on the data streams. It has 449 rows of data implying that each row corresponds to each file in the dataset. The metadata file has information on where exactly the two points were. It contains information like street, city, latitude, longitude, postal code, and country for the two points. Apart from this it also contains ext\_id and REPORTID. These columns were present in the data set as well. Upon further look, ext\_id in the metadata file did not match the ext\_id present in the datasets but the REPORTID were same in the both file. So we can use the REPORTID in the metadata file with the REPORTID in the dataset to join the two tables if the need arises.

## Method



**Data Retrieval, Combining** **and** **Labeling**

After retrieval and combining of the dataset, in order to perform supervised learning algorithms, we need to train the dataset using labels. In this dataset there are no labels present. Average speed column in the dataset can be used to label the dataset as speed represents how fast the cars are going and this tells us how congested the road is. If we want to predict traffic congestion, velocity of cars in the road can be considered the label we want to predict. Traffic congestion can be broken down in 3 different levels, heavy, mild, and low. For our algorithm, we will assign average speed of less than 30 kilometers per hour (kph) to be heavy traffic congestion, average speed of between 31 and 60 kph to be mild traffic congestion and average speed of 60 kph or higher to be low traffic congestion. Using these labels we can now train our models and predict for any new data.

**Splitting Dataset (Training and Test Sets)**

For building any supervised learning models we can break up datasets into a training set and a testing set. Only training set is used to train the classification model. We train and tune our model using the training set and test how well the model can generalize using the test set with data that the model has never seen. One common way of splitting the dataset is by using 10% of dataset for testing and 90% of dataset for training. This may lead to some bias in the classification result and the model we built may not necessarily be generalizable. In order to prohibit that we can use another well-accepted method called N-Fold cross validation, in which you randomize the dataset and create N number of almost equal size partitions. Then we can choose the Nth partition for testing and remaining partitions for training the classifier. Within the training set we can further employ another K-fold cross validation to create a validation set and find the best parameters. And repeat this process N time to get an average of the metric. Since we want to get rid of classifier bias we repeat this above process certain number of times by randomizing data and splitting into N fold and take average of the metric. This will result in a non-biased classification model that we can generalize.

**Machine Learning Algorithms/ Predictive Models**

Machine learning algorithms can be generalized into two different types, supervised learning and unsupervised learning. In supervised learning, we train a model for each input with a corresponding target and later predict target for any new input. If the targets are in distinct classes we call it a classification model and if the target is continuous we call it a regression model. Where as in unsupervised learning there are no targets. We evaluate the relationship between different inputs and their structure in unsupervised learning. One of the most important unsupervised learning methods is clustering; where we group input data based on the inherent structure of those inputs and build a model to place a new input data in one of the groups created.

For the purpose of the study we will look into following machine learning algorithms:

**Decision Tree**

Decision tree is an algorithm used for building classification models whose output looks like a tree structure. Decision tree consists of root node, test node and decision nodes (leaf node). A decision tree is a flowchart-like structure in which each internal node represents a test on an attribute, each branch represents the outcome of the test and each leaf node represents a class label, which is the decision taken after computing all attributes. The paths from root to leaf represent classification rules. In decision analysis a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values of competing alternatives are calculated. A decision tree classifies data by following the route from root node to the decision node according the set attributes or criteria. Once the decision tree is build we start at the root node to apply the testing scenario and follow the branch that fits the scenario, ending up at one of the leaf nodes which are one of the classes of the classification model. We can use Rpart package in R through zeppelin to perform decision trees algorithms.

**Support Vector Machines**

Support Vector Machines (SVM) is another widely used algorithm for supervised learning as can be used for both classification and regression. The basic idea of SVM is to map the data into a high-dimensional feature space via a nonlinear mapping. A linear learning machine in a kernel induced feature space learns a non-linear function while the capacity of the system is controlled by a parameter that does not depend on the dimensionality of the space. With a given training data set that are marked with labels, SVM training algorithm builds a model that assigns new data point into a specific category. We can use e1071 package in R through zeppelin to build SVM model.

**Random Forest**

Random Forest is a machine learning algorithm that uses decision trees. Random forest does a collective classification using decisions from different decision tress. We don’t make decision based on just one decision tree, but by an almost unanimous prediction made by a set number of decision trees. A random forest consists of thousands of individual trees that are trained on different part of a training set. Random forest provides very good accuracy compared to other classification algorithms and it can be very efficient for large data sets, which makes this algorithm very scalable. Another benefit of random forest algorithm is that it can handle many variables at one time thus allowing the model to be built for high dimensional data. We can use randomForest package in R through zeppelin to build a Random Forest model.

**Time Series Analysis: Autoregressive Integrated Moving Average (ARIMA)**

Since the dataset we are looking at is time based we can conduct time series analysis to predict future points in the series. We can also look into seasonality as we may guess that the traffic data changes during the course of the day and the week. We can use the velocity data to predict future velocity or number of vehicles at a given time using time series models like Auto-Regressive Integrated Moving Average (ARIMA) model. The auto correlation function (ACF) and partial auto correlation function (PACF) can be used to determine which order of auto regression and moving average is required for the model. By predicting velocity at a certain time we can make suggestions as to what time are busy and when to be careful when using the roads. If the ACF shows any significance at any of the lags we use that order of lag for the moving average and if PACF shows any significant spikes at any of the lags we use that order of lags for the auto regression. After we set the order for the model we can generate the model using R through Zeppelin. We can then predict next series of data points based on the model. In order to predict the accuracy of the model we can use mean average predicted error (MAPE).

**Evaluation of Model**

When we are done with building of the different algorithms we need to decide which algorithm gives us the best result. In other words, we need to evaluate our models. We use the model on the test dataset to get the prediction from the model and compare them against the expected labels that we had in hand. There are many performance metrics that help us evaluate the performance of our models.

For binary classification, we can look at the number of true positives (tp), which are the correct affirmative predictions, true negatives (tn), which are the correct negative predictions, false positives (fp), which are the incorrect affirmative predictions, and false negatives (fn), which are the incorrect negative predictions. Using these values we can calculate different measures of model performance like accuracy (tp + tn)/ (p + n), precision (tp / (tp + fp)), recall (tp / (tp + fn)), specificity (tn / (fp + tn)), fall-out (fp / (fp + tn)), F1 score (2 \* tp / (2\*tp + fp + fn)), etc.

We can also look at a receiver operating characteristic curve or ROC curve to evaluate model performance. ROC curve can be plotted for different models to visualize which model is performing the best. The x-axis of the ROC curve is false positive rate while the y-axis is true positive rate. The ROC curve is hence the recall as a function of fall out. We look at area under the curve (AUC) to calculate the probability that a model will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

Using these measures we can evaluatewhich models performs best on the test dataset to come up with a final model to be used as the prediction model.

**Prediction**

# This is the final model we pick based on the evaluation of different models that we build. For any new dataset this model will predict the class based on the inputs provided.

# Reference:

1. Lippi, M., Bertini, M., & Frasconi, P. (2013). Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning. *IEEE Transactions on Intelligent Transportation Systems*, 14(2), 871-882. doi:10.1109/tits.2013.2247040
2. Steinley, D., & Brusco, M. J. (2011). Choosing the number of clusters in Κ-means clustering. *Psychological Methods*, *16*(3), 285-297. doi:10.1037/a0023346
3. Guha, S., Rastogi, R., & Shim, K. (2001). Cure: an efficient clustering algorithm for large databases. *Information Systems*, 26(1), 35-58. doi:10.1016/s0306-4379(01)00008-4
4. Kumar, S. V., & Vanajakshi, L. (2015). Short-term traffic flow prediction using seasonal ARIMA model with limited input data.*European Transport Research Review, 7*(3), 1-9. doi:http://dx.doi.org/10.1007/s12544-015-0170-8
5. Smith, B., & Demetsky, M. (n.d.). Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches. *Proceedings of IEEE International Conference on Systems, Man and Cybernetics.* doi:10.1109/icsmc.1994.400094
6. Zhao, Z., Chen, W., Yue, H., & Liu, Z. (2016). A novel short-term traffic forecast model based on travel distance estimation and ARIMA. *2016 Chinese Control and Decision Conference (CCDC)*. doi:10.1109/ccdc.2016.7532126
7. Business, F. S. (n.d.). Identifying the orders of AR and MA terms in an ARIMA model. Retrieved February 07, 2017, from http://people.duke.edu/~rnau/411arim3.htm
8. Li, R., Jiang, C., Zhu, F., & Chen, X. (2016). Traffic flow data forecasting based on interval type-2 fuzzy sets theory. (2016). *IEEE/CAA Journal of Automatica Sinica*, 3(2), 141-148. doi:10.1109/jas.2016.7451101
9. Hou, Y., Edara, P., & Sun, C. (2015). Traffic Flow Forecasting for Urban Work Zones. *IEEE Transactions on Intelligent Transportation Systems*, 16(4), 1761-1770. doi:10.1109/tits.2014.2371993
10. Abadi, A., Rajabioun, T., & Ioannou, P. A. (2014). Traffic Flow Prediction for Road Transportation Networks With Limited Traffic Data. *IEEE Transactions on Intelligent Transportation Systems,* 1-10. doi:10.1109/tits.2014.2337238
11. Hou, Z., & Li, X. (2016). Repeatability and Similarity of Freeway Traffic Flow and Long-Term Prediction Under Big Data. IEEE Transactions on Intelligent Transportation Systems, 17(6), 1786-1796. doi:10.1109/tits.2015.2511156
12. Lv, Y., Duan, Y., Kang, W., Li, Z., & Wang, F. (2014). Traffic Flow Prediction With Big Data: A Deep Learning Approach. *IEEE Transactions on Intelligent Transportation Systems,*1-9. doi:10.1109/tits.2014.2345663
13. Koesdwiady, A., Soua, R., & Karray, F. (2016). Improving Traffic Flow Prediction With Weather Information in Connected Cars: A Deep Learning Approach. *IEEE Transactions on Vehicular Technology,*65(12), 9508-9517. doi:10.1109/tvt.2016.2585575
14. Xia, D., Li, H., Wang, B., Li, Y., & Zhang, Z. (2016). A Map Reduce-Based Nearest Neighbor Approach for Big-Data-Driven Traffic Flow Prediction. *IEEE Access*, 4, 2920-2934. doi:10.1109/access.2016.2570021
15. Yu, Y., Li, J., Guan, H., Wang, C., & Wen, C. (2016). Bag of Contextual-Visual Words for Road Scene Object Detection From Mobile Laser Scanning Data. *IEEE Transactions on Intelligent Transportation Systems,*17(12), 3391-3406. doi:10.1109/tits.2016.2550798
16. Xu, Y., Kong, Q., Klette, R., & Liu, Y. (2014). Accurate and Interpretable Bayesian MARS for Traffic Flow Prediction. *IEEE Transactions on Intelligent Transportation Systems,* 15(6), 2457-2469. doi:10.1109/tits.2014.231579
17. Chan, K. Y., Dillon, T. S., & Chang, E. (2013). An Intelligent Particle Swarm Optimization for Short-Term Traffic Flow Forecasting Using on-Road Sensor Systems. *IEEE Transactions on Industrial Electronics,* 60(10), 4714-4725. doi:10.1109/tie.2012.2213556
18. Chan, K. Y., & Dillon, T. S. (2013). On-Road Sensor Configuration Design for Traffic Flow Prediction Using Fuzzy Neural Networks and Taguchi Method. *IEEE Transactions on Instrumentation and Measurement*, 62(1), 50-59. doi:10.1109/tim.2012.2212506
19. Zhao, J., & Sun, S. (2016). High-Order Gaussian Process Dynamical Models for Traffic Flow Prediction. *IEEE Transactions on Intelligent Transportation Systems*,17(7), 2014-2019. doi:10.1109/tits.2016.2515105
20. Mo, H., Hao, X., Zheng, H., Liu, Z., & Wen, D. (2016). Linguistic Dynamic Analysis of Traffic Flow Based on Social Media—A Case Study. *IEEE Transactions on Intelligent Transportation Systems*, 17(9), 2668-2676. doi:10.1109/tits.2016.2530698