Knowledge-Based Systems 93 (2016) 33–46

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Traffic big data prediction and visualization using Fast Incremental Model Trees-Drift Detection (FIMT-DD)

Ari Wibisono*, Wisnu Jatmiko, Hanief Arief Wisesa, Benny Hardjono, Petrus Mursanto

Faculty of Computer Science, Universitas Indonesia, Kampus UI Depok, 16424 Depok, Indonesia

ARTICLE INFO

Article history:
Received 7 May 2015
Revised 24 October 2015
Accepted 30 October 2015
Available online 10 November 2015

Keywords:
Intelligent traffic systems
Data stream
Traffic prediction
Traffic visualization

ABSTRACT

Information extraction using distributed sensors has been widely used to obtain information knowledge from various regions or areas. Vehicle traffic data extraction is one of the ways to gather information in order to get the traffic condition information. This research intends to predict and visualize the traffic conditions in a particular road region. Traffic data was obtained from Department of Transport UK. These data are collected using hundreds of sensors for 24 h. Thus, the size of data is very huge. In order to get the behavior of the traffic condition, we need to analyze the huge dataset which was obtained from the sensors. The uses of conventional data mining methods are not sufficient to use, due to the process of knowledge building that should store data temporary in the memory. The fact that data is continuously becoming larger over time, therefore we need to find a method that could automatically adapt to process data in the form of streams. We use method called FIMT-DD (Fast Incremental Model Trees-Drift Detection) to analyze and predict the very large traffic dataset. Based on the prediction system that we have developed, we also visualize the prediction of traffic flow condition within generated sensor point in the real map simulation.

1. Introduction

Nowadays, the increased number of vehicle and people results on the heavily congested traffic condition. The growth of the road infrastructure could not keep up with the huge number of vehicles. Insufficient space and funds to construct the new roads become the reasons of the imbalance between the number of vehicles and road infrastructures. Vehicle traffic data was generated and collected from hundreds of sensors that are placed on road segments. The sensors operate in a 24 h a day in real time scenario, therefore the volume of the data will increase in size over time. Traditional data mining methods will not be able to process the very huge dataset because it computes the whole training dataset in the memory. Thus, the memory does not have a sufficient capacity to compute the very huge dataset. In this paper, we use data stream method in order to build a knowledge which represents the traffic datasets that have been trained by FIMT-DD algorithm. Traditional data mining uses memory in order to process the data, but when the data is very big, the memory does not have sufficient capacity to process it. Thus in this paper, we use data stream method in order to build a knowledge which is built by using huge traffic dataset [19].

There are a few methods to gather the traffic data from the segment of the road, we can use loop sensor which scattered the electromagnetic wave to obtain the traffic condition. We can also use image processing to extract the data frame by frame to obtain valuable information. Ni [25] defines traffic flow characteristics in ITS (Intelligent Transportation System) applications. Flow is the measure of throughput, density is the ideal indicators of traffic incident, and space mean speed which is the primary input to calculate the travel times. The quality of data depends on implemented sensors. The sensors that are commonly used are loop detectors, mobile sensors, aerial photography, satellite imagery, and mobile sensors such as global positioning system. Loop detectors are usually placed at a fixed location along the roadway. It counts and measures the amount of vehicles and also the speed in time domain.

The other sensor that is commonly used is camera. Cheng and Hsu [13] presented the intelligent highway surveillance system which is designed for day and night times. The proposed framework used background subtraction technique to detect the object in day time and use headlight pairing method to recognize the vehicle in night time. The proposed method has the ability to automatically adapt and differentiate when to use tracking for day and night times. Jianming et al. [21] also proposed a traffic congestion identification by using image processing. Although, variables like density, velocity, and flow are commonly used to show the condition of particular road area, these parameters could not reflect the detailed traffic scenarios. It

* Corresponding author. Tel.: +62 21 786 3419; fax: +62 21 786 3415.
E-mail address: ari.w@cs.ui.ac.id, ari.wibisonozbw@gmail.com (A. Wibisono).

http://dx.doi.org/10.1016/j.knosys.2015.10.028
0950-7051/

shows a novel method to detect traffic congestion recognition based on vehicle path as an image.

Besides camera and loop sensor, RFID can also be used as a sensor to capture the traffic data. Chattaraj et al. [12] use RFID sensor to improve the intelligent traffic control system. The system is implemented Decision Making System (DMS) for controlling the traffic signal. The DMS contains a set of algorithms that automatically control the traffic light. Besides calculating the number of cars, the system assigned priority parameter such as, the type of vehicle, priority assigned to the vehicle, priority assigned to the travel path, and the time of day.

The traffic data which were gathered from various sensors will be extracted in order to obtain valuable information. Some researchers have implemented several frameworks to process sensors data [5]. The intelligent system that has been developed is Intelligent Vehicle Highway Systems (IVHS). IVHS provide traffic management systems, driver information systems and vehicle control systems. Vaa et al. [29] described the accurate and real information is highly needed by the drivers and travelers. Thus, the development of the IVHS which makes the driver to be assisted to make decision based on routes and destination. Other technologies such as ADAS (Advanced Driver Assistance System) and RT (Roadside Telematics) also help people in facing dense traffic conditions. This research also has an objective to measure the effectiveness of the ITS, which will influence the driver behavior.

Sundar, Hebbar, and Golla [27] developed the intelligent traffic control to help the detection of stolen vehicle, clearance of ambulance, and controlling the traffic congestion. They used RFID which has been installed on the vehicle. This RFID acts as a sensor to monitor the number of vehicles. The RFID sensors could also be implemented to detect the ambulance which will pass the traffic light. When the ambulance is approaching, the system will use green wave method in order to set the green light on in every junction which will be passed by the ambulance. This system could also help to detect a stolen vehicle, when it passed the RFID reader. The car data is read and the authority will check the car data id with the list data of car that has been reported stolen.

Patel and Ranganathan [26] developed intelligent decision making system for urban control system which uses Artificial Neural Network (ANN) as a learning system. The knowledge of the system has been acquired by training of the traffic data which change dynamically over time. This acquired knowledge will be used for decision making to control the adaptive traffic light. The comparison of methods to control traffic lights is described by Araghi et al. [2]. This comparison provides several artificial intelligence methods to optimally control the traffic light. The knowledge of the system was gathered from the traffic information. The traffic lights in intersection act as intelligent agents that have knowledge in order to decide the effective green time for every phase.

Chan, Dillon, and Chang [10] use particle swarm optimization (PSO) combined with support vector machine (SVM) to present the traffic safety forecasting. It uses traffic data in China from 1970 to 2006. Chan et al. [11] use PSO and ANN in order to develop traffic flow model. This model acts as predictors to provide the traffic flow forecasting information. These algorithms were implemented to forecast the traffic flow condition of freeways in Western Australia. Xue and Dong [31] presented about contraflow operation. It is commonly used to minimize the traffic conditions where the numbers of traffic which comes from opposite directions have big difference with each other. It gives us clear explanation about the implementation of contraflow operation using intelligent system and fuzzy pattern recognition to predict precisely the coming traffic condition.

Lu et al. [22] present a survey in the field of transfer learning using computational intelligence which utilizes the knowledge that has been acquired before in order to solve new but similar problems effectively. For example, it may help us to recognize an apple if we learn to recognize pear. Darmoul and Elkosantini [14] develop decision support system in order to control disturbance using biological immunity. They develop the prototype to help decision makers to react in the case of disturbance (detection, reaction, and learning strategies). Billhardt and Lujak [9] present the implementation of dynamic coordination system of ambulances for emergency medical assistance services. The main objective of this research is to give assistance to transport the patient to medical center. EMA service is essential to reduce the average time of the ambulance fleet. Augeri et al. [4] present their research about the development of decision support system to decide the safe and appropriate speed for speed zones. Speed management becomes very important in order to improve road safety. There are a few factors that have to be considered in this research such as, accident rate, roadway geometry, roadway development, and traffic.

The amount of incoming stream of data is very large. In fact, the size of the data sequence is potentially infinite. This condition is going to make it nearly impossible to keep the entire data inside the memory. Only a small portion that could be computed and calculated and the rest will be discarded [17].

Data stream has attracted a lot of attention in the research community. Wang et al. [30] propose real-time streaming data processing system which implemented by using apache storm. Apache storm is a computation framework which is used to process distributed data in real time. The data which comes from vehicle probe sensor is processed by using multiple linear regression in order to get the vehicle traffic behavior. Data which is gathered continuously will get bigger in size [23]. The limitation that can be avoided is the limitation of memory. It can be solved by using streaming algorithms. Some researchers have developed algorithms with improved memory usage and performance [6].

In this research, we observe traffic data from sparse sensors that were collected by the Highway Agency in UK. We also validate the data and measure the error rate of the traffic flow condition. The algorithm that is used to measure the error rate is FIMT-DD (Fast Incremental Model Trees with Drift Detection) algorithm. In order to see the traffic behavior globally from the data, we built a map visualization based on the 5 years traffic data. There are several stages which were conducted to build a map visualization. First, we generate new point geolocation coordinates between each sensor in order to get the new point which is close to another. For example, if the previous distance between two sensors is 2 km, then we generate 1 sensor for every 0.5 km. Thus the total sensors placed are 4 sensors. If the number of sensors increases, it will result in a more detail view of the traffic flow visualization map.

After the sensor generation is complete, we need to predict the number of vehicles at each new point sensors. We build the knowledge and predict the traffic flow in the new point sensors. The rest of the paper is organized as follows. Related work is presented in Section 1. The research workflow, knowledge development, and algorithms are presented in Section 2. The result, visualization implementation, and error measurement area are presented in Section 3. We conclude our paper in Section 4.

2. Methods

Processing of traffic data can be solved with the regression method. Regression analysis is one of the most frequently discussed topics in machine learning and statistics area. Fast generation of data makes the data become very large. Therefore, we cannot use traditional regression methods which compute and store all the data for the processing inside the memory [7].

We used FIMT-DD (Fast Incremental Model Trees with Drift Detection) to analyze the huge vehicle traffic data. In this paper, FIMT-DD was used to solve the very huge dataset problems, FIMT-DD is an algorithm that works in the area of incremental learning of regression

and model trees from time-changing data streams. Instead of processing all of the data inside the memory, this algorithm allows us to process and learn the data on every stream arrival [20].

Furthermore we used the knowledge from the huge dataset in order to create more detailed visualization about traffic condition in particular road area.

2.1. Research workflow

In this research, we have designed a framework as a reference in carrying out research stages. Based on Fig. 2 the first step of the stages is data preparation (1.1), in this stage, we processed raw data as material input for testing. After performing data preparations, we performed data cleansing to clean data which is incomplete data. We have tested the data using data stream algorithm and measured the error performance (1.2 and 1.3).

The next step is prepared the data for visualization (2.1), we have prepared the parameters which become the candidate for training. Afterward we tested the data visualization and observed the error performance (2.2 and 2.3). After the process of error measurement is conducted, we generate new point geolocation coordinate between each sensors in order to get the new point which is more close to another (2.4). This process is done, in order to get the more detailed visualization of traffic flow from more number of sensors.

The next step is the development of the knowledge. This step was conducted in order to process data features which have been cleaned and processed. The input features for knowledge development are traffic sensor id which is converted to the point location of the sensor, date and time that is acquired online in 24 hours, average journey time (s) between sensors, the average speed of vehicles in particular time span, and the last is the distance between sensors (km). These input features are used to create the prediction of the traffic flow, which is the amount of cars between sensors in specific time (2.5). As can be seen in Fig. 2, we have performed tests for the knowledge that has been built (stages 1.1, 1.2, and 1.3).

All of the traffic data within 5 years were trained incrementally. This activity is done to get the knowledge information from the whole dataset. Data stream learning algorithm for prediction was used to build the knowledge. Furthermore, the quality of the prediction was evaluated using prediction sequential. This process evaluates the error between the predicted value and actual value incrementally.

Traffic data prediction was conducted after the process of knowledge development has been completed. It can be seen the process of these prediction scenarios in Fig. 2 (stages 2.1–2.6). The target of the prediction is to process the predicted value of traffic flow. As we know, traffic flow is the average number of vehicles in particular range of time and distance. The data itself provided us 15 min timespan, the distance was provided by the length between two sensors. The last step of the workflow is to implement the visualization of traffic prediction in a map (2.6). The result of the prediction was used to give more detailed visualization within maps. We used heat map, in order to show the visualization of the traffic condition. The simulation of the data is presented in the form of heat maps which is switched over time based on common period of time. It starts from 00:01–23:59 every day.

2.2. Traffic data acquisition

In this research, we obtained the traffic data from a Highway Agency in UK which is presented in Fig. 1. This data series provides flow information, average traffic speed, and average trip time for a period of 15 min. This data is available from 2009 to 2013 which is obtained from the highway managed by a Highway Agency in UK.

Travel time and average speed is calculated using a combination of sources, including Automatic Number Plate Recognition (ANPR) cameras, in-vehicle Global Positioning System (GPS) and inductive loops built into the road surface. Travel time is derived from a real vehicle observation and calculated by using adjacent time periods.

The data obtained from these sensors is stored on the server. One month sensor data have 800 MB average size. We used 5 years of traffic data starting from 2009 to 2013. Therefore the amount of data that we have to process is as 800 MB $\times$ 12 months $\times$ 5 years $=$ 48,000 MB or 48 GB. In this traffic data, there are several attributes that were used for data processing, initial point coordinates, end point coordinates, time and date, average travel time, average speed, the length between the starting point and end point. The class is targeted for predictive analysis is traffic flow (number of vehicles).

Fig. 1a–c is the visualization of the loop sensor traffic data. The loop sensor that was taken as an example is the reference link with unique id AL2989A. The data of AL2989A loop sensor contained the traffic data which was collected for 5 years. This data was gathered from particular highway road segment in UK. Fig. 1a–c describes the 5 years traffic data between 2009 and 2013. The sensor represents the data in time format, month, day, and year. Y-axis from Fig. 1a–c represents the traffic flow, which is the amount of vehicles that is detected by sensor within a 15 min period. The whole dataset is shown by Fig. 1a, this figure represents 5 years data. Fig. 1b magnifies the data into smaller time span. It shows the traffic flow data from 2009, January to 2009, December. Fig. 1c presents a more detailed characterization of traffic flow in this sensor. Traffic condition is quite dense in the morning and heavily congested at mid-day to afternoon. It is started to reduce at the end of the day. Fig. 1a–c is the example of one sensor that we have analyzed, overall there are 2500 sensors that are contained in the data which are scattered separately on the motorway in the UK [28].

Unique id of the data refers to the place where the sensor is placed. We have changed this feature unique id into latitude and longitude coordinate in maps. From this feature we have extracted two attributes. Those attributes are latitude and longitude. For date data, e.g. date of Dec, 30, 2014. We performed data conversion date into one feature, namely the number of days in one week, 1: Monday–7: Sunday.

Time data was extracted in grades between 1 and 96, which depicts 00:00–24:00. Average Journey Time feature is represented in seconds. Average speed is represented in km/h. The length between the starting point and end point is represented in km. Traffic Flow is represented in a number of vehicles.

Based on Table 1 linkRef is a reference link which connects two sensors. This data is represented as unique id. The second feature is the description of those connected links. The next feature is the date of the data is acquired. This data is represented with yy-mm-dd data format, year, month, and day. The time period is the marker for period of time within one day data. It marks the data every 15 min, integer 1 is represents as 00:15 o'clock and 2 for 00:30. So the integers that are used in time period feature are within 1–96. Another feature that we used for knowledge development is the average journey time between two sensors which is represented in seconds. In addition, loop sensors in this data also measures the average of vehicle speed passing through the sensor in a 15 min time span with km/h format. Data quality represents the quality of data that has been acquired, 1 indicates highest quality data and 5 is the lowest. Link length is the feature which represents the actual distance between sensors that is measure in km. Lastly traffic flow is the average number of vehicles between sensors with 15 min time span.

We have conducted data processing in order to represent and predict the traffic data effectively. The linkRef is the unique id which is the connected link between two sensors. It is converted into geographic location of those two sensors which is represented with the latitude and longitude of each sensor. The converted linkRef assists us in representing each of sensors into the digital map. The geolocations of the sensors were combined with the traffic flow were

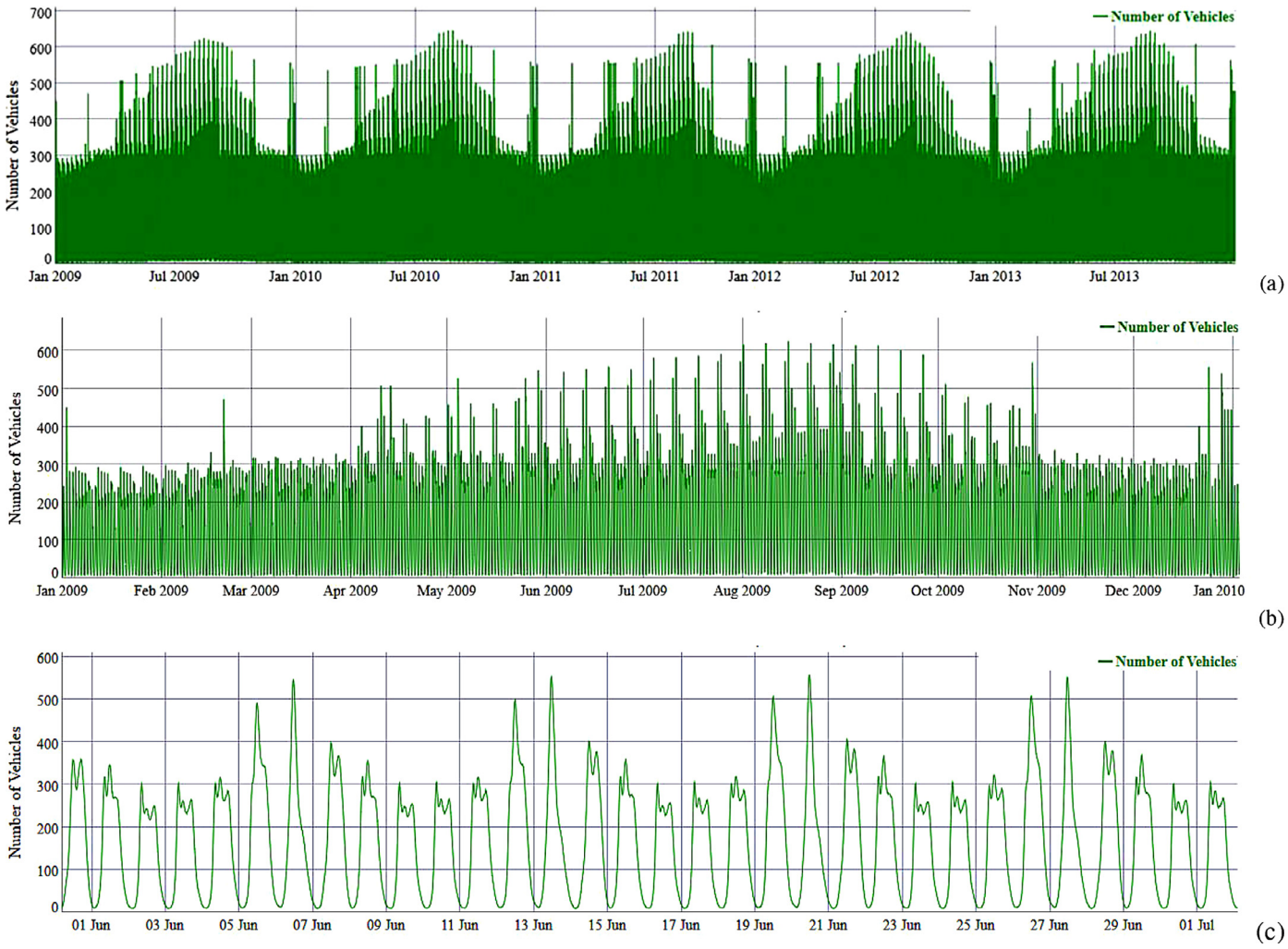

Fig. 1. Vehicle data of (AL2989A) sensor from 2009 to 2013 (a) Number of Vehicles between 2009 and 2013 (AL2989A), (b) Zoom to 1 year Jan 2009–Dec 2009 (AL2989A), (c) Zoom to 1 month 1 June 2009–31 June 2009 (AL2989A).

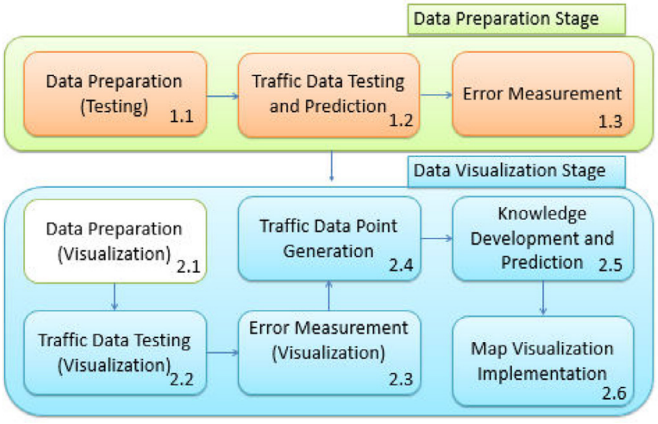

Fig. 2. Research workflow.

represented as weight that enables us to represent the data as a heat map simulation.

Table 1
Original traffic data attributes.

No	Original attribute	Description
1	LinkRef	A unique alphanumeric link id representing a junction to junction link on the Highway's Agency managed road network.
2	LinkDescription	Description of the link.
3	Date	Date of travel.
4	TimePeriod	One of 96 15-min intervals in the day that the data refers to (0–95 where 0 indicates 00:00 to 00:15).
5	AverageJT	The average journey time to travel across the 'LinkRef' in seconds, of vehicles entering the junction to junction link within a given 15-min time period.
6	AverageSpeed	The average speed (in km/h) of vehicles entering the junction to junction link within a given 15-min time period.
7	DataQuality	Indicator showing the quality of the journey time data for the link and time period. 1 indicates the highest quality data and 5 the lowest.
8	LinkLength	The length of the link (km).
9	Flow	An average of the observed flow for the link, time period and day type.

2.3. Knowledge development

2.3.1. FIMT-DD algorithm

There are problems in the learning data streams using tree models. The dataset is no longer fixed in size, it grows automatically based on the stream. It is not possible to store all data in memory and studying the data as a whole.

We need an incremental algorithm that can determine the decision to split the tree. The other problem, training datasets should

have also several different distributions. The model requires monitoring and updating when a change is detected. In this research, we used the FIMT-DD (fast incremental model of trees with drift detection) algorithm. This algorithm starts with a blank leaf after reading the examples in a sequence based on the arrival. FIMT-DD finds the best split for each attributes and will sort it based on measurement evaluation. If it meets the splitting criteria then this algorithm will split the attribute based on the best attribute. If the change has been detected, adaptation strategy on the tree will be performed [20].

2.3.2. Split criterion

Some of the literatures have mentioned a few strategies in implemented split criterion [1]. Hoeffding bound is used in attribute selection process in order to decide the best attribute between the samples [24,18]. In FIMT-DD, the splitting criterion which was used is SDR (Standard Deviation Reduction) which can be done incrementally. For example, the dataset S of size N to be analyzed, hA in attribute A will split the data into two subsets, name SL and SR with the size of NL and NR with the notation $S = SL\ U\ SR;\ N = NL + NR$. The formula for measuring split of SDR hA is

$$\mathrm{SDR}(hA) = sd(S) - \frac{NL}{N} sd(S_L) - \frac{NR}{N} sd(S_R) \tag{1}$$

$$sd\ (S) = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} (yi - y)^2\right)} = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} yi^2 - \frac{1}{N}\left(\sum_{i=1}^{N} yi\right)^2\right)} \tag{2}$$

In formula (2) it could be observed, this algorithm maintains the value of predicted attribute y and y squared with the number of instances that have passed leaf nodes. If hA is the best split of attribute A and hB is the best split of attribute B then, we can decide the real-value random variable r between 0 and 1.

$$r = \frac{\mathrm{SDR}(h_B)}{\mathrm{SDR}\ (h_A)} \tag{3}$$

After that we can check the ratio between the second best split after the few examples of streams. Each r value of each stream can be represented in real numbers $r1, r2, \ldots, rn$. Hoeffding bound probability [18] can be used to get the high confidence interval of the mean random variables. Probability bound can help us to determine $1 - \delta$ which is the average of a random sample of N variables with range R and the distance ε. Value of ε is calculated using the formula

$$\varepsilon = \sqrt{\frac{R^2 \ln\left(\frac{1}{\delta}\right)}{2N}} \tag{4}$$

The value of ε continues to decrease as the number of observations N. With many values have been observed, the sampled mean gets near to the true mean. Hoeffding bound gives an exponential decrease of the probability for the sum of random variables to deviate from its expected value. We describe the upper and lower bounds on the estimated sample like:

$$r^+ = r + \varepsilon \quad \text{and} \quad r^- = r - \varepsilon \quad \text{and} \quad r^- \le r_{true} \le r^+ \tag{5}$$

If the upper bound of the sample average is below 1 then the average true is below 1, the best analyzed attribute of sample data is also the best attributes of the entire distributions. Therefore, with confidence $1 - \delta$ the split hA is the best split. If a situation occurs where two or more splits have almost the same SDR value then the confidence intervals were determined by ε is reduced. As a result of this, we cannot find the difference, we need to choose the SDR for the best splitting criterion. We can use the stopping rule which is defined as threshold τ for ε. If the error occurred is smaller than the ε and splitting criterion not found then the best split is detected with a higher value of the SDR.

2.3.3. Numerical attributes

Selection of the distribution tree is dependent upon the number of points of divisions. The use of computing resources and large memory were influenced by the great value difference for the numerical tribute. Technique is usually used to solve this problem is pre-processing technique. This technique requires initial initialization before the data is processed. However, this technique cannot be done, because the data stream has fast arrival rate [18,15].

Therefore we need an algorithm which can adaptively make distributions to correspond with the arrival of the data. In FIMT-DD algorithm, it uses E-BST (Extended Binary Search Tree) which could perform statistical quantification and help us to do the online sorting sequence. E-BST using two arrays of 3 elements, among others: The first array stores the statistics to compare wether the value of the attribute values are less than or equal to the key. (1) The number of instance counter has reached these nodes. (2) The sum value of the target attributes. (3) The sum of squares value of the target attributes. Array (2) contains the statistics of the instance to reach the node with a value more than key. E-BST structures are incrementally updated at any new example in leaf.

2.3.4. Linear models in leaves

In FIMT-DD algorithm, the weights are updated for every instance in the stream. FIMT-DD uses an easy approach, it did not select the attribute. All of numerical attributes are calculated using the regression equation without activation function. The weight of relation within the parameters of the perceptron is a linear equation.

FIMT-DD algorithm using gradient descent method (using the mean square error) is similar to the perceptron rule for training the perceptron. The fundamental difference is that the weights are updated every arrival of new data instances instead of using the entire dataset. In order to perform the update, this algorithm uses the Delta Rule where the learning process can be convergent. In order to get the output value, the new incoming instances are processed using the current weight. Each weight is updated based on the value of the difference in the output (o), the real value (y), normalized attributes (xi) and learning rate (η)

$$\omega_i = \omega_i + \ \eta\ (o - y)x_i,\ i \ne 0 \tag{6}$$

Variables can be categorized and converted into binary (numerical) variables before the learning process. The values of the attributes should be normalized (standardized) before the learning process begins. Therefore, each of the attributes has the same effect in the training process. This normalization process can be done incrementally. It stores the sum of all the target values and the square of the target value square during the beginning of the initial learning process. FIMT-DD algorithm uses standard deviations, which is shown by the formula (7)

$$x_i' = \frac{x_i - x}{3\ \sigma} \tag{7}$$

Each perceptron learning phase is done in parallel with the addition of nodes. If the node separation is complete, linear mode will be given to a child node, this is done to avoid training the data from scratch.

2.4. Traffic prediction and visualization

The traffic condition prediction is processed after the knowledge development, which is the basic constraint to make predictions. Prediction is done by doing training for the entire data traffic within the last 5 years from 2500 sensors spread across the motorway in the UK. After we performed data cleansing and enhancement features, the size of data traffic reaches 22 GB. To be able to process the data, we use data stream method which is built by Ikonomovska et al. [20].

Table 2
LinkRef conversion.

LinkRef	LinkDescription	StartLat	StartLng	EndLat	EndLng
AL1000	(AL1000)	52.72649	1.74652	52.69244	−1.78818
AL1001	A38 between A5127 and A513 (AL1001)	52.69238	−1.7884	52.72652	−1.7467
AL1004	A38 between A513 and A5121 (AL1004)	52.72652	−1.7467	52.78825	−1.67542
AL1007	A38 between A5121 and A513 (AL1007)	52.78924	−1.67476	52.72649	−1.74652

To measure the error of traffic prediction, we used MAE (Mean Absolute Error). Usually, these measurement techniques are done by using cross validation technique, but due to the huge traffic data which have been acquired we cannot use this technique to measure the prediction. The other alternative way to measure the error is done by using prediction sequential method. It measures the error at each of iterations together with the knowledge development process, so the trend of the error result is updated continuously. We use MOA (Massive Online Analysis) to build the knowledge, test the data [8], and show the data using dygraphs library [16].

In the visualization stage, we used traditional map as an interactive medium to perform the simulation. Simulation is done by placing the sensor together with the position of latitude and longitude respectively. Each sensor is represented in the form of point base in geographic position in map. The color of the sensor is changed based on the number of vehicle counted by the sensors. We used 2500 sensors data which is directly implemented into the digital map, however there is a gap between the sensors (approximately 2–5 miles each). The gap between each of the sensors is illustrated by the map.

We proposed a method to minimize the number of gaps in the map, so that we can minimize the gap between sensors in visualization predictions. In this method, we have generated the geo-location between the sensors to obtain a virtual geographic location sensor at every 0.5 km. We used Google Maps API in order to generate the virtual location on the map. In every process of sensor's virtual location generation, we have got 5 to 10 new generated sensors.

In addition, in order to predict the traffic flow in each sensor, we have trained and built a knowledge using FIMT-DD algorithm. Once the training process is done, and the knowledge for prediction has been obtained. The process of creating of the prediction based on the constraint of the knowledge will be conducted. The result of this process is the traffic flow prediction, it gives us more detailed representation of the traffic condition that is animated in each sensor.

The amount of traffic flow which is simulated by each sensor will be described in heatmap form. Some of the points on the map will change its color according to the number of traffic flow in a particular time. Heatmap at each sensor is colored green if the traffic flow of the vehicle is in a good traffic condition, and red if traffic flow is in heavy traffic.

3. Results and discussions

The main objective of this research is to predict and visualize the traffic behavior across the motorway in the UK. The data set that we have observed is very big. We should extract the important information from that data and build the knowledge about the traffic condition. The first stage that we did in this research was to evaluate the FIMT-DD algorithm. The evaluation of the algorithm is done by using sequential prediction evaluation approach. Target parameter that is the subject of evaluation is the number of vehicles traffic flow at particular time. Based on the evaluation result in the first stage, the next stage we focused on the development of the knowledge. We used FIMT-DD algorithm to build the knowledge. It will be used to predict the traffic situation in each sensor. We also do a sensor generation of the distance between each of the sensors. We have also generated extra sensors between each existing sensor to minimize the distance between each sensor. This will create a more detailed visualization of the traffic flow animation, as can be seen in Fig. 4. In Fig. 4, we generate and minimize the distances of each sensor in order to visualize the traffic flow animation in more detail. The traffic flow prediction is done on each of the new sensors, producing the traffic flow on each of the new sensors. We also visualized the traffic flow of the vehicle with interactive time simulation on a map.

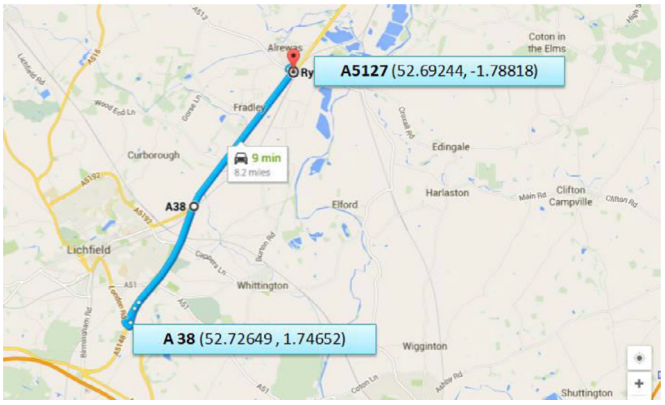

Fig. 3. LinkRef representation in maps.

Table 3
Simulation feature attributes.

No	Original attribute	Simulation attribute
1	LinkRef	– Source latitude – Source longitude – Destination latitude – Desitnation longitude
2	LinkDescription	(Removed)
3	Date	Converted into number of day in a week. (1–7)
4	TimePeriod	One of 96 15-min intervals in the day that the data refers to (0–95 where 0 indicates 00:00 to 00:15).
5	AverageJT	Average journey time (in seconds), calculate every 15 min
6	AverageSpeed	Average speed (km/h
7	DataQuality	(Removed)
8	LinkLength	The length of the link (km).
9	Flow	An average of the observed flow for the link, time period and day type.

3.1. Evaluation and testing

Several stages were conducted in order to get the knowledge information and predict the traffic flow. First, we extract and cleaned the data. In this stage the 5 years traffic data from more than 2500 sensors were integrated. In line with the data integration process, we also conducted the extraction process of the unique sensor id into new features. As we know, unique sensor id is represented as a link between sensors on the motorway of UK. We extracted both the start point and point of the id into geolocation data. After the extraction process has completed, we will have 4 new features, those are start point latitude, start point longitude, end point latitude, and end point longitude. Next, we conduct the data cleansing.

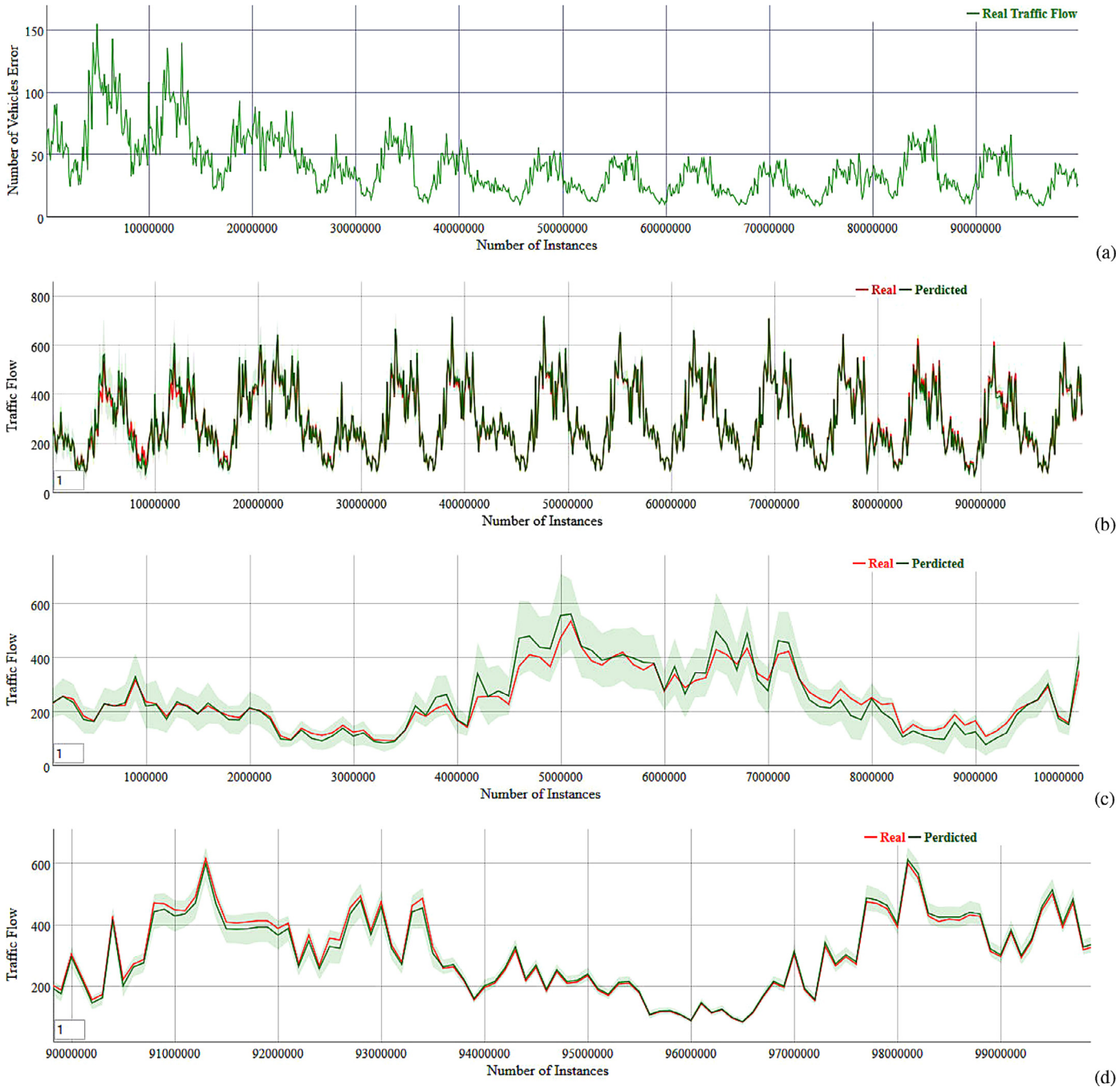

Fig. 4. Result of raw data simulation (a) Traffic flow error measurement for 1–100 million instances, (b) Real traffic flow (red), predicted traffic flow(green), and error traffic flow (shaded green) for 1–100 million instances, (c) Real traffic flow (red), predicted traffic flow (green), and error traffic flow (shaded green) for 1 million–10 million instances, (d) Real traffic flow (red), predicted traffic flow (green), and error traffic flow (shaded green) 90 million–100 million instances. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

LinkRef is the unique id of sensors which represents the connectivity between sensors. From Table 2, we can see the LinkRef Al 1000 represents the connectivity between A38 sensor and A5127 sensor. Each of the unique sensor ids was represented into the geolocation in map. A38 sensor is represented as 52.72649, 1.74652 and sensor A5127 is represented as 52.69244, −1.78818 for its latitude and longitude position respectively. In Fig. 3, we can see the representation of the connectivity between sensors. The representation of each geolocation data in each sensor becomes one of the features in the data processing and testing.

Next, we conducted the prediction of the sequential testing for all the traffic data. The data itself, after extraction process, has 8 features and the number of lines in the file is about 200,000,000. The testing is measured every 100,000 iterations in order to observe the trend of error in the testing process. Some of the features from the original data are removed for knowledge building and testing, such as LinkDescription and DataQuality. LinkDescription is the text description of the data and DataQuality is in the form of integer which represented the quality of data from the sensors. The complete features that are evaluated are described in Table 3.

The traffic data itself contain 22 GB of data which should be trained to get the knowledge extraction. The knowledge extraction is done by using this computer specification Intel(R) Core(TM) i7-4790K CPU @ 4.00 GHz, Memory 32 GB, HDD 2 TB. We measure the performance of MAE (Mean Absolute Error). The formula of MAE is shown in formula (8). f_i is the real data and y_i is the predicted value of data and n is the number of data calculated

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| \tag{8}$$

Simulations were performed to determine how much the prediction error is obtained from FIMT-DD algorithm. This result has become the justification to implement FIMT-DD in the visualization step. The simulation was performed using 100,000,000 instances. The entire instances were simulated using prediction sequential method. Error measurement is calculated in every 100,000 instances. We used prediction sequential method in order to determine the magnitude of the error in every 100,000 iterations. It is expected, by the end of the simulation, the traffic prediction knowledge that have learned about the traffic behavior could justify the amount of traffic flow accurately.

Overview of the error magnitude measurement which is generated by the algorithm is shown in Fig. 4a. In Fig. 4b we can see the error trend of traffic flow prediction result and the comparison between the real world traffic with the predicted traffic flow data. Fig. 4a shows the trend of error declined since the beginning of the simulation until the end of the simulation. The simulation starts from 100,000 instances, the first 100,000 instances generated an error of 74.36. It then proceed to the next instance of 200,000, it generated an error of 90.34. The error measurement calculation continued until 100,000,000 instances. The error trend result from the simulation of sequential prediction is likely to decrease. The results that have the highest value among the data are between instances 0 and 10,000,000. More precisely, at instances 5,000,000 the highest value is 155.71.

After that, the error is generated decreased gradually from 20,000,000 to 90,000,000. The lowest error result value is at instances 96,000,000 which has 9.2 errors. In general, after the error results of evaluations have been carried out, the results of the error tend to have little value after learning by using huge number of instances instead of learning using small instances.

Fig. 4b shows an overall picture of three components: the actual number of traffic flow (red line), the prediction of traffic flow (green line), and the number of errors on the flow of traffic (green shaded). By using Fig. 4b we can see the trend of the overall position prediction of traffic flow compared to the real traffic flow along with the average number of errors generated. Fig. 4c shows the trend of data ranging from 100,000 to 10,000,000 instances and have the same components as Fig. 4b. Fig. 4c shows more detailed comparison between the real traffic flow, predicted traffic flow and error.

In Fig. 4c we can see at instances 400,000, the predicted traffic is 172.66, whereas the average real traffic flow is 185.45 and the average number of errors is 54.36. Error differentiation results at instances 400,000 indicated by shaded green color that has a range between 120.44 (172.66 − 54.36) and 224.88 (172.66 + 54.36). According to these results, we can see that the prediction results obtained were good results. The average result of the generated prediction is close to the actual traffic flow.

The error value has a tendency to decrease along with the increasing number of instances that is learned by the system. Fig. 4d shows us predicted value, real value, and the error of traffic flow. It shows the data in a zoomed perspective with the data ranging from 90 million to 100 million. Based on the diagram, we can see that, the error value is close to the predicted line between instances 95,000,000–96,000,000. At the instances 96,000,000, the average number of traffic flow generated prediction is 89.39, the average number of real traffic flow is 89.36 and the average error is generated is 9.2. Fig. 4d shows the error number at instances 96,000,000 is very small and the difference between the prediction and the real traffic flow is very little. From the results of this simulation, the error values will generally decreases with the increasing number of instances that were trained. In order to visualize the traffic condition accurately, we should have knowledge on how much error is generated. The predictive simulation and error performance measurement that we have done is the basis for us to visualize the state of the traffic from at 00:00 to 23:59 from Monday-week day.

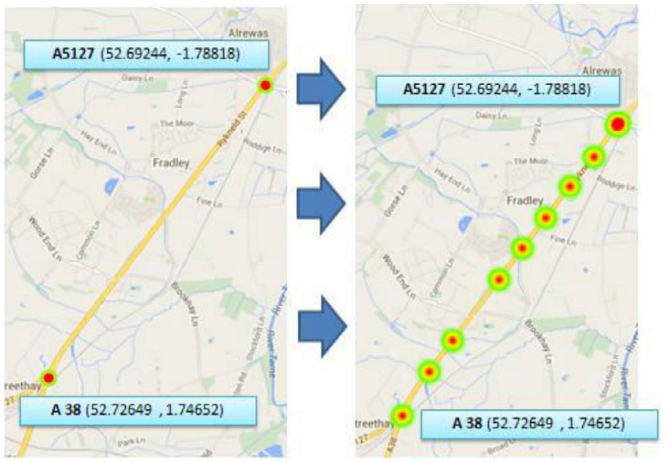

Fig. 5. Sensor generation every 0.5 km.

3.2. Implementation traffic map visualization

We have conducted the implementation of traffic map visualization in two stages. The first stage is data extraction. The second stage is data cleansing, and the third stage is extraction of unique id which has the same process in evaluation process. In order to create a more detailed visual animation, we minimize the distance of each sensor.

As can be seen in Fig. 3, the distance between each sensor is about 2–3 km, thus, in order to get the more detail visualization on the map, we have minimized the distances to 0.5 km. The detail algorithm of traffic point generation is presented in Appendix A Algorithm 1: Traffic Point Generation Web Service and Algorithm 2: Traffic Loop Point. We use the help of Google Maps API in order to generate the points with a distance of 0.5 km. The detail process of the traffic points generation can be seen in Fig. 5.

In stage 2.3, we conduct evaluation to measure the error of visualization data done in each of 100,000 iterations. After the evaluation has been done, we conducted the traffic point generation in stage 2.4. The knowledge development using FIMT-DD is done in stage 2.5. After the knowledge has been acquired, we conducted the prediction of traffic flow in each generated sensor which we have been generated in stage 2.4. In the last stage of this research, we visualized the traffic flow data in the real map animation (2.6).

Visualization of the traffic condition on the motorway in the UK used 5 years traffic data, from 2009 to 2013. The data has 2.500 location points which has been generated to 7.500 points in order to visualize more detailed traffic conditions at certain time. To obtain the data visualization, we did a test on traffic data that will become the primary knowledge to predict the traffic flow for data visualization. We make the process similar with the previous simulation. The process of sequential predictive simulation is still used, but for data visualization, we have reduced some of the data feature. The data which is only needed for visualization are in the form of latitude and longitude, day number (number of day in the week), and time number data as the data features. The data target is traffic flow. Before we build

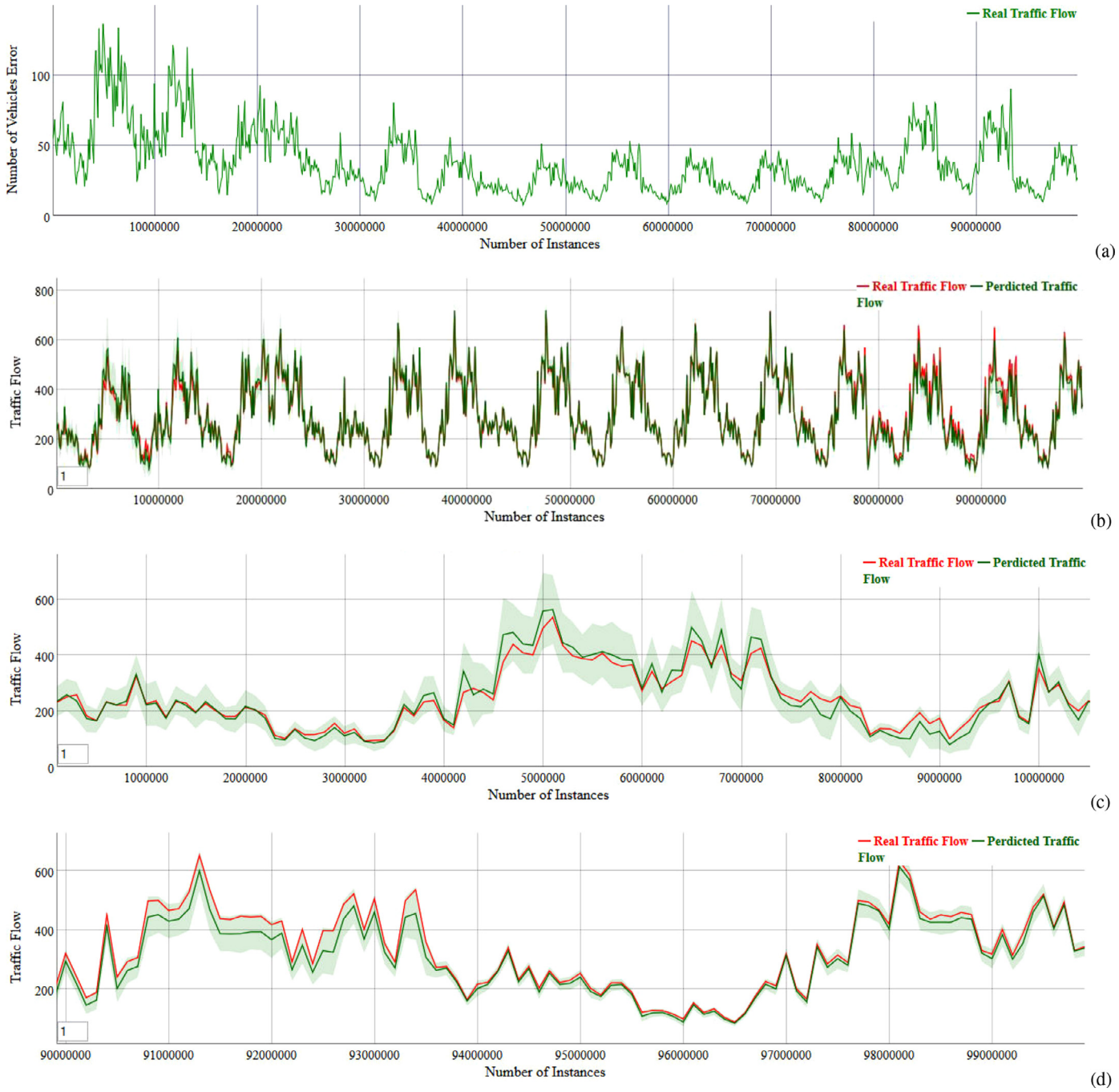

Fig. 6. Result of visual data simulation (a) Traffic flow error measurement for 1–100,000,000 instances, (b) Real traffic flow (red), predicted traffic flow (green), and error traffic flow (shaded green) for 1–100,000,000 million instances, (c) Real traffic flow (red), predicted traffic flow (green), and error traffic flow (shaded green) for 1,000,000–10,000,000 instances, (d) Real traffic flow (red), Predicted traffic flow(green), and traffic flow error (shaded green) 90,000,000–100,000,000 instances. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the data for visualization of traffic flow, first we need to perform a sequential prediction with features latitude and longitude positions, number of days in the week (1–7), time 24 h divided into 15-min time span (1–96).

Secondly, the results were generated from the prediction error which is shown by Fig. 6a. The average error in Fig. 6a tends to have the largest value in the ranges between 0 and 10,000,000 instances with an average of 136.76 at instance 5,000,000. The result of error has tendency to decrease in the range of 90,000,000–100,000,000 which can be seen in Fig. 6d. The comparison between the average number of errors, the average number of predictions, and the average real data traffic flows is described in Fig. 6b. The green color indicates the prediction of traffic flow, the red color indicates the real traffic flow, and the shade of green color indicates the error that occurred in its instances. Fig. 6c shows the instances ranging from 0 to 10,000,000. From that figure, we can see in the diagram at instances 5,000,000, the measurement of the average traffic flow prediction is 557.82, the average real traffic flow 495.97, and the average error is 136.77. As can be seen in Fig. 6c the error values (shaded green) that occurred in the range between 0 and 10,000,000 are much larger than the errors that occurred in the range between 90,000,000 and 100,000,000 million. It can be described from Fig. 6a, at instances

Table 4
Traffic data error measurement for visualization.

Data	Real	Predicted	MAE	RMSE	Error %
2,500,000	204.77	202.5	52.53	94.62	30.9
5,000,000	220.64	234.47	64.7	126.08	33.93
7,500,000	366.28	380.97	94.66	162.47	29.96
10,000,000	196.86	176.01	52.92	107.21	33.26
12,500,000	333.5	350.46	75.88	138.36	27.38
15,000,000	324.96	334.71	64.43	118.78	23.87
17,500,000	179.47	168.38	32.93	70.04	24.33
20,000,000	373.37	383.16	54.09	99.45	18.87
22,500,000	447.23	458.75	63.05	117.35	18.69
25,000,000	347	351.2	48.91	100.11	18
27,500,000	177.62	175.72	27.02	60.1	20.07
30,000,000	248.64	250.51	31.82	69.05	16.84
32,500,000	177.63	178.35	21.34	48.63	16.47
35,000,000	427.46	435.63	51.76	106.04	15.86
37,500,000	216.83	220.41	21.91	55.75	13.39
40,000,000	405.99	415.33	34.63	74.3	11.62
42,500,000	326.74	333.05	27	61.39	11.33
45,000,000	240.02	242.35	18.64	41.82	10.98
47,500,000	237.91	240.93	18.61	47.03	11.46
50,000,000	455.02	462.52	32.44	64.38	9.48
52,500,000	228.13	226.89	19.67	41.46	11.66
55,000,000	265.93	266.6	22.97	52.92	12.07
57,500,000	430.42	433.21	36.89	72.57	10.85
60,000,000	193.96	191.52	16.22	30.26	11.84
62,500,000	342.53	341.62	26.06	46.73	10.45
65,000,000	390.56	390.23	29.98	50.88	10.03
67,500,000	205.2	200.88	17.19	31.51	12.11
70,000,000	354.63	350.76	27	47.34	10.85
72,500,000	373.15	369.46	30.63	51.85	11.07
75,000,000	187.15	177.01	18.77	33.77	14.83
77,500,000	386.21	371.2	33.05	54.32	12.45
80,000,000	321.25	303.22	35.21	66.04	15.95
82,500,000	200.22	180.53	30.54	67.23	20.4
85,000,000	406.01	372.03	56.54	119.23	18.51
87,500,000	334.68	301.48	49.03	103.7	19.68
90,000,000	172.38	151.13	25.91	50.06	22.39
92,500,000	399.81	356.19	55.51	99.27	20.2
95,000,000	321.23	297	37.83	75.31	15.82
97,500,000	183.62	175.71	18.38	32.82	14.83

96,500,000 the average traffic flow prediction is 88.81, the average real traffic flow is 85.15, and the error is 9.96. In general, the resulting error will be reduced on a large instances position. According to the results which are obtained in the simulation of traffic flow for data visualization, we make prediction knowledge with 100,000,000 instances which are used to predict the data visualization. Every instances data visualization consists of position (latitude and longitude), time (number of days in the week), and the representation of time (1–96) 24 h divided by 15 min.

The number of instances of traffic data are about 100,000,000 instances. It can be seen on the *x*-axis in Fig. 6(a). In order to simply present the traffic data evaluation table, we present the average error every 2,500,000 instances in Table 4 in column 1 shows the sequence instance of the data, column 2 shows the average value of the actual traffic flow, column 3 represents the average Predictive generated by the system, column 4 shows the MAE (Mean Absolute Error), column 5 (Root Mean Square Error, and column 6 is the SMAPE (Symmetric Mean Absolute Percentage Error) [3]. The formula of SMAPE is shown in Eq. (9). f_i is the real data and y_i is the predicted value of data and n is the number of data calculated.

$$\text{SMAPE} = \frac{1}{n} \times 100\,\% \sum_{i=1}^{n} \frac{|f_i - y_i|}{|f_i + y_i| / 2} \tag{9}$$

We can see in Table 4 2,500,000 first instance has the biggest average MAE. It is 52.53 and the number of percentage error is 25.66%. Once the iterative process of learning process and evaluation arrive at instance 97,500,000, the value of average error is 18.38 and average percentage error number 10.01%. Based on the average error evaluation conducted, it shows that learning process using the iterative algorithm by using data stream can enrich the knowledge traffic prediction models information and minimize the error value.

Trend of percentage error rate evaluation can be seen in Fig. 7. In Fig. 7 we can see the number of error average value decreases slowly from instance 2,500,000 to instance of 97,500,000. Based on Fig. 7 the smallest error percentage is obtained when the learning process arrives at instance 50,000,000. It shows the value of 9.48% percentage of error. The evaluation of error shows that the biggest percentage error value resulted in this evaluation is 33.93% which is conducted in the first 5,000,000 instances.

Based on these results, this system has a relatively small error and can be used to make predictions. Once we know the system has a small error value, we predict the value of the traffic flow using the generated position. It is resulted from traffic data point generation process in stages 2.4. We develop knowledge of the 100,000,000 instances which were used as references for prediction. Prediction is done using these parameter latitude, longitude, number of days in the week (1–7), time 24 h divided into 15-min time span (1–96), and the target value of the prediction is the number of traffic flow. All of these parameters are generated to each of 7.205 generated sensor for each day and timespan. For example, the sensor with geo location point at 50.6805, −3.5154 generated to number of days in week (1–7) and 15 min timespan (1–96) on each day. Thus each of the sensor data has $7 \times 96 = 672$ rows of data. After the generation has been done we predict traffic flow with those parameters.

After the prediction sequential test and the development of knowledge have been carried out, we make predictions for the data visualization. It has 7205 points that consist of latitude and longitude, and the amount of traffic flow between 00:00 and 23:59. The implementation of the traffic flow visualization used Google Maps API which is shown by Fig. 8. We used heatmap library in order to represent the amount of traffic flow. Heatmap library will have a red

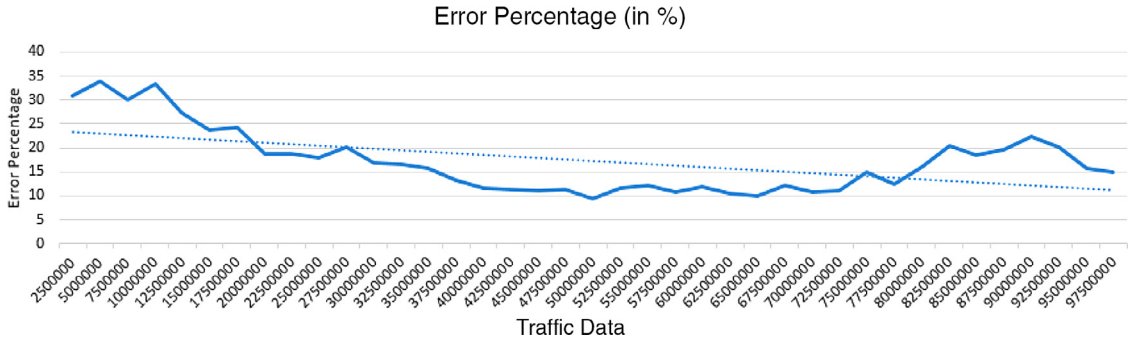

Fig. 7. Traffic data error measurement trend.

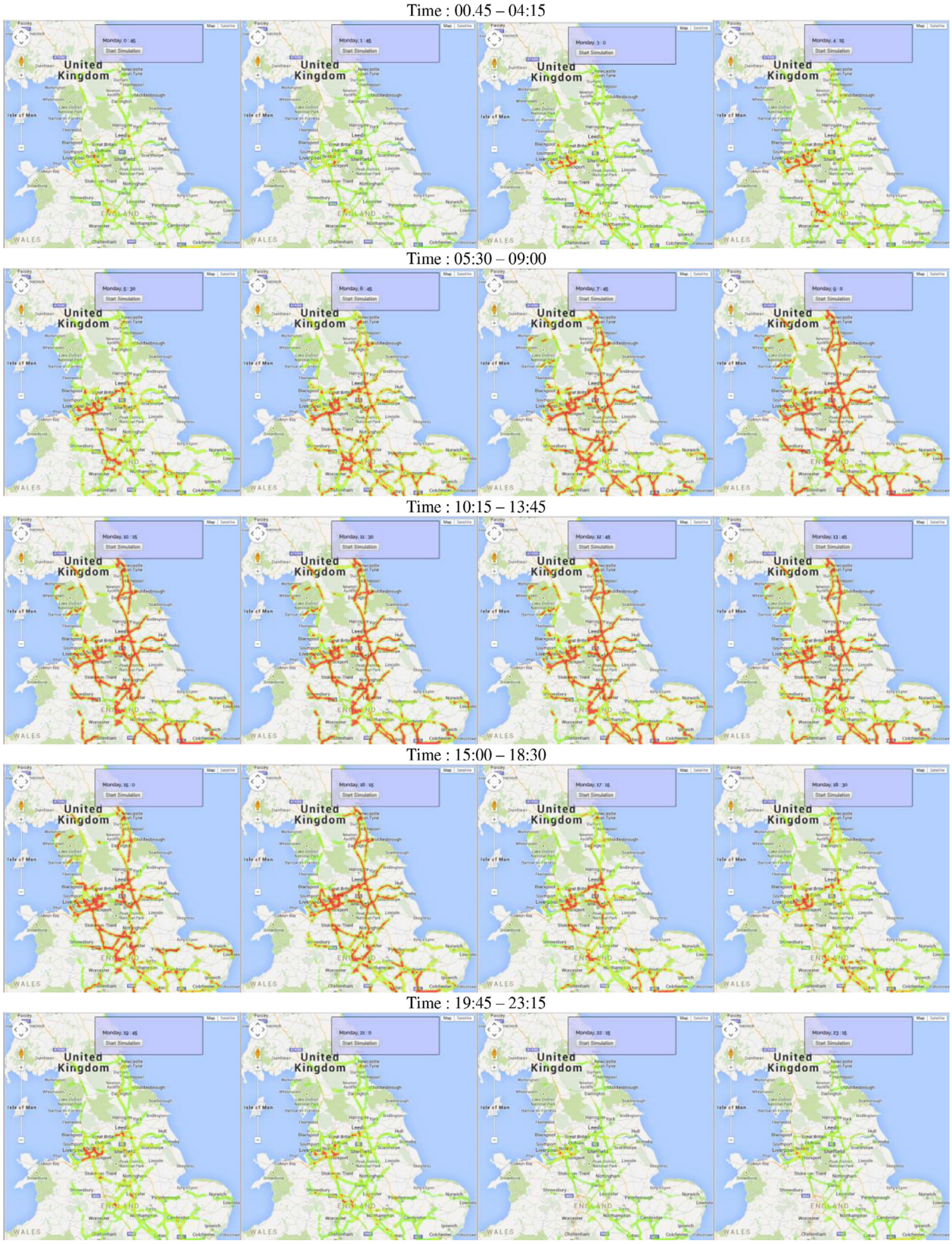

Fig. 8. Result of visualization using predicted data.

color if the traffic flow reaches the maximum point that is already defined (>800) and the heatmap will have a green color if the amount of traffic flow is relatively small (<100). Before the simulation is produced, we do the partitioning of the data visualization. The data are divided into any instances of time, 1-1 means 1 represents the first day of the week (Monday) and the other number represents the first 15 min in 1 day (00:15). File 1-1 consists of 7.502 data points in latitude and longitude format accompanied by the weight for traffic flow. The file instance that should be produced is 7 (the number of days in a week) × 96 (the number of time in one day 1–96) = 672 iterations. The file is slowly loaded by Google Maps API every 0.75 s to describe the visualization prediction of traffic flow that occurred in 7.502 points in the UK motorway. Visualization in the form of changes in traffic flow was animated from Monday–Sunday. The algorithm of traffic map visualization is presented in Appendix A, Algorithm 3: Traffic Flow Visualization.

In Fig. 8 we can see an animation that runs on Monday, starting at 00:15 to 23:15. From the pictures that are shown in Fig. 8, we can see the amount of traffic flow is low between the hours of 00:45 to 04:00. This is indicated by the color green that dominated the heatmap. At 5:30 a.m. to 9:00 the amount of traffic flow showed a significant increase, in general, almost all the heatmap points were red. This is consistent with the fact at this hour, it is the time when the resident have started their activities (morning rush hour). Furthermore, at 10:15 to 13:45 the traffic was also dense. At 15:00–18:30, the heatmap still indicated the traffic flow is congested, but at around 18:30, the heatmap representation in red began to decrease. This is consistent with the fact usually at these hours resident ended their activities, so that the amount of traffic flow in some points is reduced. At 19:45–23: 15, the number of traffic flows tends to be dominated by the green color of heatmap indicating that the traffic flow is not dense. Afterward, green heatmap representation increases dominantly, which means the number of vehicles on the motorway is decreased. This simulation has been done for each day of the week, so hopefully with this simulation we can see the characteristics of people who use the toll road at particular location and time.

4. Conclusions

In this research we carry out the processing of traffic data during the 5-year in motorway in UK. Those data are processed and analyzed using FIMT-DD algorithm. Some of the attributes that we have used in this study are the latitude and longitude coordinates of the sensor, the distance between the sensors, average speed, average travel time, and day (date). The target attribute is the number of vehicles or traffic flow. The system that could learn as well as perform traffic condition prediction in the form of map visualization has been built in this research. We have built the knowledge of traffic condition based on huge training data (100 million lines). The data has been tested iteratively in order to get the error performance. The method that we used in order to measure the performance in each of iterations is prediction sequential. The result of prediction sequential shows the error performance decreased steadily along with the increasing number of trained data. We continue to build the knowledge of traffic data and developed the visualization based on the knowledge which has been built. The visualization of traffic condition is simulated in the form of time animation. The traffic condition is described as traffic flow number in a particular area of road in the form of heatmap. It simulated the traffic condition iteratively from 00:00 to 23:59. By using the predicted traffic condition animation which has been developed, we can see the traffic behavior that uses the toll road on particular road and time.

Acknowledgments

We would like to express our gratitude for the grant received from Universitas Indonesia and Directorate of Higher Education MP3EI (2012–2014) with Hibah Riset Awal, Grant No: 3356/H2.R12/HKP.05.00/2014 in supporting this research.

Appendix A

Algorithm 1. Traffic Point Generation Web Service

```
Input : StartLat, StartLong, EndLat, EndLong
Step = 500 // 500 m
Intialize(StartPosition, EndPosition) //Initialize start position and end position
createMarker() //create marker between start position and end position
NumberofStep = StepCalculation (StartPosition, EndPosition)
currentPosition = startPosition
for (i=0; i<NumberofStep, i++){
        if (currentPosition > endPosition and currentPosition < endPosition){
                currentPosition = getPointAtDistance (currentPosition + step)
                // add the current position 500 m ahead
                writeTextFile(currentPosition);
                // write the latitude and longitude in file
        }
}
```

Algorithm 2. Traffic Loop Point

```
Input : Set of PointObject in startPosition and endPosition
urlPoint = pointGeneration // url pointed to Algorithm 1: Traffic Point Generation web service
foreach (PointObject as pos){
        urlPoint = url + pos.startlat + pos.startlong + posendLat + posendLong
        //create url to call traffic point generation web service based on latitude and longitude parameter
        window.open (urlPoint) // call the web service with initialize url
        sleep (2000)// sleep for 2 seconds
}
```

Algorithm 3. Traffic Flow Visualization

```
Input : Set of files with parameters of traffic flow, latitude, longitude
//Each file represent the timestep in every 15 minute
heatmap1 //initialize heatmap
arrayOfTraffic // initialize array of traffic with paramters of traffic flow, latitude, and longitude
day //intialize date
fileArray // initialize Array of File
// function to visualize animation in every 15 minutes timestep in a week
function timeLoop (){
        generateArray();
        for (i < number of day * number of timestep in a day){
                processData (fileArray[i])
                setTimeOut (9);
        }
}
//function to load the content of the files into the memory
function generateArray(){
        url = location of the files //url to the location of the files
        for (var dk = 1;dk<=numberOfDayinaWeek;dk++){
                        for (var mk=1;mk<=96;mk++)//24 hours divided by 15 minutes
                        {
                                fileArray [loopK] = readFile(url+dk+"-"+mk) // assign every file into array in memory
                                loopK++;
                        }
}
// function to process data in every time step
function processData(fileArray){
        //iterate every row of the content into googleMaps Location Array
        foreach(row as fileArray){
                arrayOfTraffic.push ({
                location: new
                google.maps.LatLng(
                        row["lat"], //assign latitude
                        row["lon"]), //assign longitude
                        weight: row["flow"]   //assign the traffic flow
                });
        }
        loadHeatmap(arrayOfTraffic); //call the heatMap Function
}
//function to create the heatmap in googleMaps
function loadHeatmap (arrayOfTraffic){
        pointArray = new google.maps.MVCArray(arrayOfTraffic)
        //create array of points according to the location of the latitude and longitude also the flow number represent the color of the heatmap
        day = day.add(minutes : 15)
        //add the initialize date to forward 15 minutes
        heatmap1.setData(pointArray)
        //execute the heatmap with pointArray
}
```

References

[1] C. Agrawal, Data Streams: Models and Algorithms, Springer, New York, 2006.
[2] S. Araghi, A. Khosravi, D. Creighton, A review on computational intelligence methods for controlling traffic signal timing, Expert Syst. Appl. 42 (3) (2015) 538–1550.
[3] J. Armstrong, Long-Range Forecasting: From Crystal Ball to Computer, Wiley, 1985.
[4] M.G. Augeri, P. Cozzo, S. Greco, Dominance-based rough set approach: an application case study for setting speed limits for vehicles in speed controlled zones, Knowledge-Based Syst. 89 (2015) 288–300.
[5] L. Baskar, B. Schutter, J. Hellendorn, Z. Papp, Traffic control and intelligent vehicle highway systems: a survey, IET Intel. Transp. Syst. 5 (1) (2009) 38–52.
[6] A. Bifet, Adaptive Stream Mining Pattern Learning and Mining from Evolving Data Streams, IOS Press BV, 2010.
[7] A. Bifet, C. Carlos, P. Chirita, & I. Weber (2005). An analysis of factors used in a search engine's ranking, in: *First International Workshop on Adversarial Information Retrieval on the Web*.
[8] A. Bifet, G. Holmes, R. Kirby, B. Pfahringer, MOA: massive online analysis, J. Mach. Learn. Res. 11 (2010) 1601–1604.
[9] H. Billhardt, M. Lujak, Dynamic coordination of ambulances for emergency medical assistance services, Knowledge-Based Syst. 70 (3) (2014) 268–280.
[10] K.Y. Chan, T.S. Dillon, E. Chang, An intelligent particle swarm optimization for short-term traffic flow forecasting using on-road sensor systems, IEEE Trans. Ind. Electron. 60 (2013) 4714–4725.
[11] K. Chan, T. Dillon, E. Chang, J. Singh, Prediction of short-term traffic variables using intelligent swarm-based neural networks, IEEE Trans. Control Syst. Technol. 21 (2013) 263–274.
[12] A. Chattaraj, S. Bansal, A. Chandra, An intelligent traffic control system using RFID, IEEE Potentials 28 (3) (2009) 40–43.
[13] H.-Y. Cheng, S.-H. Hsu, Intelligent highway traffic surveillance with self-diagnosis abilities, IEEE Trans. Intel. Transp. Syst. 12 (2011) 1462–1472.
[14] S. Darmoul, S. Elkosantini, Artificial immunity to control disturbances in public transportation systems: concepts, mechanisms and a prototype implementation of a knowledge based decision support system, Knowledge-Based Syst. 68 (2014) 58–76.
[15] P. Domingos, G. Hulten, Mining high speed data streams, in: Proceedings of 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, 2000, pp. 71–80.
[16] Dygraphs (2010). *Dygraphs*. Retrieved April 24, 2015, from Dygraphs: http://dygraphs.com/.
[17] J. Gama, R. Rocha, P. Medas, Accurate decision trees for mining high-speed data streams, in: Proceedings of 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, 2003, pp. 523–528.
[18] W. Hoeffding, Probability for sums of bounded random variables, J. Am. Stat. Assoc. 58 (1963) 13–30.
[19] G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams, in: Proceedings of 7th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, 2001, pp. 97–106.
[20] E. Ikonomovska, J. Gama, S. Džeroski, Learning model trees from evolving data streams, Data Min. Knowl. Discovery J. 23 (1) (2010) 1–41.
[21] H. Jianming, M. Qiang, W. Qi, Z. Jiajie, Z. Yi, Traffic congestion identification based on image processing, IET Intel. Transport Syst. 6 (2) (2011) 153–160.
[22] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, G. Zhang, Transfer learning using computational intelligence: a survey, Knowledge-Based Syst. 80 (2015) 14–23.
[23] C. Lucchese, High Performance Closed Frequent Itemsets Mining inspired by Emerging Computer Architectures, Universit'a Ca' Foscari di Venezia, 2008.
[24] R. Musick, J. Catlett, S. Russel, Decision theoretic sub-sampling for induction on large databases, in: Proceedings of 10th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, 1993, pp. 212–219.

[25] D. Ni, Determining traffic-flow characteristics by definition for application in ITS, IEEE Trans. Intell. Transp. Syst. 8 (2007) 181–187.
[26] M. Patel, N. Ranganathan, IDUTC: an intelligent decision-making system for urban traffic-control applications, IEEE Trans. Veh. Technol. 50 (3) (2001) 816–829.
[27] R. Sundar, S. Hebbar, V. Golla, Implementing intelligent traffic control system for congestion control, ambulance clearance, IEEE Sens. J. 15 (2015) 1109–1113.
[28] UK Highways Agency. (n.d.). *Highways Agency network journey time and traffic flow data*. Retrieved June 12, 2014, from Data.gov.uk: http://data.gov.uk/dataset/dft-eng-srn-routes-journey-times.
[29] T. Vaa, M. Penttinen, I. Spyropoulou, Intelligent transport systems and effects on road traffic accidents: state of the art, IET Intel. Transp. Syst. 1 (2007) 81–88.
[30] F. Wang, L. Hu, D. Zhou, R. Sun, Estimating online vacancies in real-time road traffic monitoring with traffic sensor data stream, Ad Hoc Networks 35 (2015) 1–11.
[31] D. Xue, Z. Dong, An intelligent contraflow control method for real-time optimal traffic scheduling using artificial neural network, fuzzy pattern recognition, and optimization, IEEE Trans. Control Syst. Technol. 8 (2000) 183–191.