Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-5 (2020) 512–517

Traffic Congestion Prediction using Decision Tree, Logistic Regression and Neural Networks

Tariku Sinshaw Tamir [1, 2], Gang Xiong [1, 3], Senior Member, IEEE, Zhishuai Li [1, 2], Hao Tao [4], Zhen Shen [5], Bin Hu [1, 5] (Corresponding Author), Heruye Mulugeta Menkir [6]

[1] State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
[3] Guangdong Engineering Research Center of 3D Printing and Intelligent Manufacturing, Cloud Computing Center, Chinese Academy of Sciences, Dongguan 523808, China
[4] China Ship Development and Design Center, Wuhan 430064, China
[5] Beijing Engineering Research Center of Intelligent Systems and Technology, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[6] Debremarkos University, Institute of Technology, Debremarkos 269, Ethiopia
tamir@ia.ac.cn, gang.xiong@ia.ac.cn, lizhishuai2017@ia.ac.cn, aiar.th@stu.xjtu.edu.cn, zhen.shen@ia.ac.cn, binhu@ia.ac.cn

Abstract: Traffic congestion is a serious problem around the world and to a great extent influences urban communities in various manners including increased stress levels, delayed deliveries, fuel wastage, and monetary losses. Therefore, an accurate congestion prediction algorithm to limit these misfortunes is fundamental. This paper presents a comparative study of traffic congestion prediction systems including decision tree, logistic regression, and neural networks. Five days of traffic information (1,231,200 samples) are utilized to drive the prediction model. The TensorFlow and the Clementine machine learning platforms are used for data preprocessing, training, and testing of the model. The confusion matrix clears that decision tree has better prediction performance and leads the other two methods with accuracy (97%), macro-average precision (95%), macro-average recall (96%), and macro-average F1_score (96%) in the python programming environment. Moreover, performance of the three prediction models is verified in clementine environment and decision tree outperforms all other models with an accuracy of 97.65%.

Keywords: Confusion matrix, Data preprocessing, Python, TensorFlow, Clementine environment, ITS.

1. INTRODUCTION

In recent years, there has been a progressive increment in traffic around the world causing pollutions, congestions, and accidents. Traffic congestion is the most widely recognized major issue and has been becoming a challenge for most cities and possibly causes low transportation efficiency, significant greenhouse gas emissions, and a lower quality of life for urban residents. For routing services and to ensure open security, timely Prediction information is necessary to decrease ongoing congestion problems in real-time response and add in structural change planning. Traffic congestion costs billions of dollars to the general public consistently because of the continuous increment in urbanization and steady-state mobility demands growth (Wang et al., 2017).

In intelligent transportation system (ITS) applications, predicting accurate traffic information such as travel time, speed, and congestion state is a significant undertaking. However, the dynamic fluctuations in traffic conditions make the prediction harder. In fact, the road conditions including highways and freeways in urban environment influence the driving speeds and the congestion state of the corresponding road (Liu et al., 2017).

Generally speaking, traffic congestion prediction in urban areas is an exceptionally troublesome undertaking. So far, the most commonly applied traffic prediction approaches used simulations and theoretical modeling. Nowadays, the availability of real-time massive datasets on traffic conditions gives an advantage to design different statistic and data driven approaches. a large number of real-world observations are needed to account for various situations in real-world. The widely available wireless sensor technologies mounted in every road network are used to derive vehicle trajectory data for designing data-driven machine learning approaches (Antoniou et al., 2007).

The rest of this paper is organized as follows. Section 2 discusses the literatures work. Section 3 presents the methods used. Section 4 shows modeling and evaluation of the prediction model, followed by the conclusion remarks in Section 5.

2. RELATED WORK

In recent years, a lot of researches has been conducted in the field of traffic congestion prediction in different road networks.

D'Andrea and his co-worker Marcelloni developed an expert system to detect traffic congestions on every road network by using current and past recent traffic speed (D'Andrea and Marcelloni, 2017). Similarly, a scalable method to predict congested traffic flow in a grid framework is proposed (Gidofalvi, 2015). Anwar and co-workers discussed another framework using a spectral-clustering based approach to monitor connected congested road sets (Anwar et al., 2016). In extended way, a customized Density-based spatial clustering of applications with noise (DBSCAN) algorithm is developed to detect and analyze repeatedly congested clusters of grids (An et al., 2016).

Considering traffic flow density and the road types, Liang and co-workers proposed a new prediction model to estimate the next time-step traffic volume on a single road segment to identify its congestion using parameters like current inflow, outflow, and traffic volume on it and adjacent upstream road segments (Liang and Wakahara, 2014). More importantly, this model is improved by Xiangjie and co-workers using a machine learning algorithm called support vector machine to predict the next time-step traffic speed and volume and subsequently estimate the congestion state of the road segments (Kong et al., 2016). Furthermore, Xiaolei and co-workers developed a deep learning-based prediction model by using a Restricted Boltzmann Machine and a Recurrent Neural Network to predict traffic congestions for all road segments in the next time-step (Ma et al., 2015).

An efficient one parameter prediction model is developed to estimate traffic conditions. The autoregressive model is combined with other prediction algorithms to guarantee the optimization of traffic flow prediction performance (Kong et al., 2013, Davoodi et al., 2016). A model combining Artificial neural networks with root mean squared error as a metric is developed by taking singular point probabilities (Qian et al., 2017).

A large portion of works in the literature focused on a pre-decided and fixed number of traffic flow parameters for a particular road network such as average speed, density, queue length, flow rate, and so on for traffic congestion prediction. However, the influence or contribution of these factors in traffic congestion varies from one location to another. Which means, in some locations, the low average speed of all the vehicles reflects the presence of congestion although that couldn't happen in the reality. Therefore, This paper proposes an adaptive prediction model that excellently incorporates different types of traffic conditions for a large number of road networks. Firstly, Tensorflow machine learning framework is used to model congestion prediction system. Secondly, clementine environment verifies the prediction model performance. Moreover, the paper presents a comparative study between the three machine learning algorithms including decision tree, neural network, and logistic regression taking accuracy, precision, recall, and F1_score as common metrics.

3. METHODOLOGY

3.1. Data Resources

The GCM (Gary-Chicago-Milwaukie) Corridor consists of sixteen urbanized counties and two thousand five hundred (2,500) miles of roadways connecting the three cities. Eight hundred fifty-five (855) sensors are placed on the roads, and each sensor collects two hundred eighty-eight (288) streams every day (one sample every 5 minutes). Each sensor collects the real-time traffic information at its location and sends it to the central server over wireless connections periodically. The general sensor integrated prediction system shown in Fig.1 works in a closed-loop manner to effectively deliver congestion estimation to the road users. Each stream sample consists of ten input attributes and one target attribute as shown in Table 1.

Four days traffic information (984,960 samples) are used for training sets and one-day traffic information (246,240 samples) are used for testing sets.

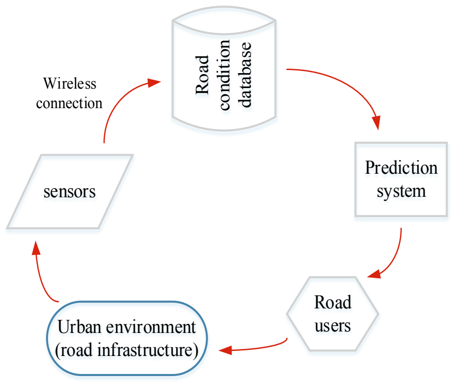

Figure 1. The general sensor integrated prediction system

Table 1: prediction system attributes

Input attributes	Target attribute
Date	
Time	
Direction of the road	
Type of road	
Link ID of the road sensor	Congestion level
Length of the road	
Travel time	
Road Volume	
Speed of car	
Occupancy of the road	

3.2. Methods used

The prediction systems are designed by combining operations including getting data, processing data, modeling prediction algorithms, and finally evaluating the prediction performance. The general system design procedure is shown in Fig.2.

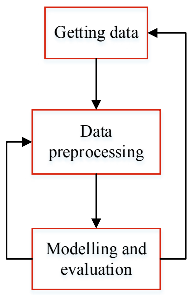

Figure 2. The general workflow

3.2.1. Getting Data

TensorFlow and Clementine machine learning platforms are used for data processing and modeling the prediction algorithms. The raw sensor data has to be processed in a systematic way to design effective and reliable prediction model. The overall data manipulation operations to be used prior to data preprocessing including reading data, checking data information, and calculating the missed data values in the whole dataset structure need to be performed.

3.2.2. Data preprocessing

In designing machine learning algorithm, more time and effort is spent on data preprocessing stage. Data preprocessing starts with categorizing the raw sensor data to different types of data structures including Categorical data (Direction, Type of road, LinkID of road sensors, Congestion Level) and Numerical data (Date, Time, Length of the road, Travel Time, road Volume, Speed of the car, Occupancy of the road). Both of the data types should be preprocessed and get ready for constructing the prediction model. The following three tasks are associated with data preprocessing.

A. Unnecessary data:

Table 2: deleting the wrong attribute labels

Name of attribute	Wrong data
Congestion level	Unknown congestion level
Direction of the road	Unknown direction type
Direction of the road	STH bound

The unnecessary wrong attribute labels shown in Table 2 are forced to be deleted. Besides, the least important attributes should be identified. The attribute named 'Link ID of the road sensor' has no role in the prediction process so that not to be considered.

From the prediction model, it is observed that an attribute 'Speed of car' has a high correlation coefficient with congestion level (i.e. 96% correlation coefficient). Therefore, it should be kicked out from the attributes. Otherwise, the prediction model could be badly affected by a single attribute.

B. Filling the missing values:

There are different methods to handle the missing data in data preprocessing; (1) Ignore the tuple, (2) fill in the missing value manually, (3) Fill in it automatically with Mean, Mode, and Median, (4) select the most probable value. In this paper, the missing values are treated by automatically filling with mode.

C. Assign numerical values for categorical data

The categorical attributes in the dataset need to be converted to numerical values. For the three types of attributes shown in Table 3, numerical values are assigned individually.

Table 3: Assigning numerical values

Type of attribute	Categorical value	Numerical value
Direction	North bound, South bound, West bound, and East bound,	0, 1, 2, and 3 respectively
Types of road	Freeway, Ramp, Arterial, Freeway express, Local road, and Freeway reversible	0, 1, 2, 3, 4, 5, and 6 respectively
Congestion level	Non congestion, Light congestion, Medium congestion, and High congestion	0, 1, 2, and 3 respectively

After the completion of data preprocessing, the dataset is left with eight input attributes and one target attribute. Consequently, training and testing datasets need to be identified prior to prediction model design. The training datasets are used to construct prediction model. The size of training dataset is an influential factor to assure better prediction of the model. To evaluate prediction performance, testing datasets are feed to the designed machine learning algorithm and subsequently, the performance is measured by the common metrics.

TensorFlow and Clementine machine learning platforms are used to design three types of congestion prediction system including decision tree, logistic regression, and neural network. Accuracy, precision, recall, and F1_score are typical metrics to evaluate those models separately.

The confusion matrix is used to summarize the performance of the classification algorithm. Table 4 shows a four-level classification confusion matrix including Non-congestion, light-congestion, medium-congestion, and high-congestion for the road network.

Table 4: Four level confusion matrix

		Predicted value			
	Congestion level	**N**on	**L**ight	**M**edium	**H**igh
Actual value	**N**on	NN	NL	NM	NH
	Light	LN	LL	LM	LH
	Medium	MN	ML	MM	MH
	High	HN	HL	HM	HH

Where,

NN: Non-congestion is predicted truly

NL: Non-congestion is predicted as Light-congestion

NM: Non-congestion is predicted as Medium-congestion

NH: Non-congestion is predicted as High-congestion

LN: Light-congestion is predicted as Non-congestion

LL: Light-congestion is predicted truly

LM: Light-congestion is predicted as Medium-congestion

LH: Light-congestion is predicted as High-congestion

MN: Medium-congestion is predicted as Non-congestion

ML: Medium-congestion is predicted as Light-congestion

MM: Medium-congestion is predicted truly

MH: Medium-congestion is predicted as High-congestion

HN: High-congestion is predicted as Non-congestion

HL: High-congestion is predicted as Light-congestion

HM: High-congestion is predicted as Medium-congestion

HH: High-congestion is predicted truly

4. Modeling and Evaluation

4.1. Modeling in python programming environment

4.1.1. Logistic regression

Logistic regression prediction algorithm is designed by taking the road network datasets. Consequently, the model generates confusion matrix as shown in Table 5 which could be used to compute performance evaluation metrics.

Table 5: Python-based logistic regression confusion matrix

		Predicted value			
	Congestion level	Non	Light	Medium	High
Actual value	Non	168496	4461	25	0
	Light	6968	30798	529	0
	Medium	42	3517	6637	5
	High	0	7	35	956

Feeding testing datasets to the logistic regression prediction model generates the prediction probabilities of the four congestion levels. Moreover, The 10 bin histogram-based graphical representation for each of probabilities including 'Non-congestion', 'Light-congestion', Medium-congestion', and 'Heavy-congestion' are shown in Fig.3, Fig.4, Fig.5, and Fig.6 respectively.

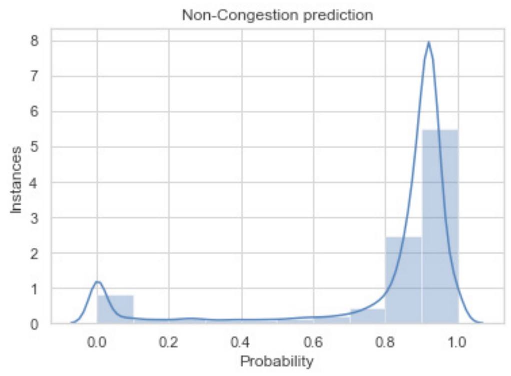

Figure 3. Probability of being non-congestion in logistic regression case

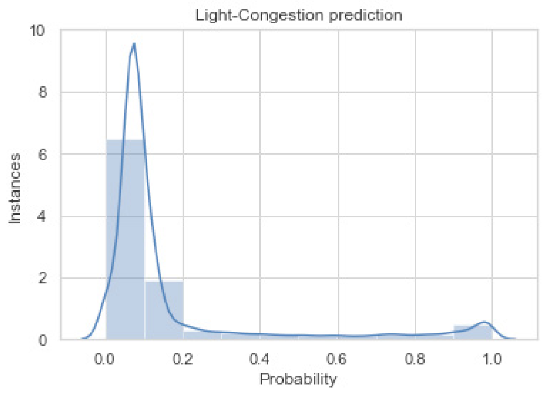

Figure 4. Probability of being light-congestion in logistic regression case

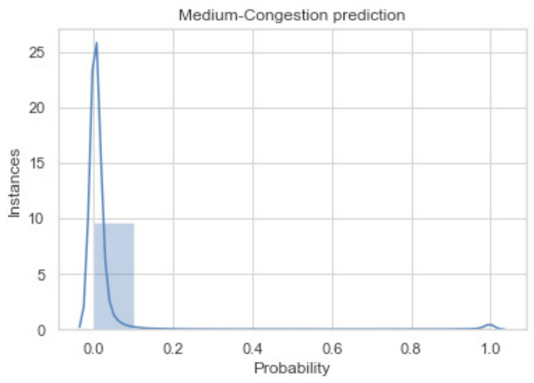

Figure 5. Probability of being medium-congestion in logistic regression case

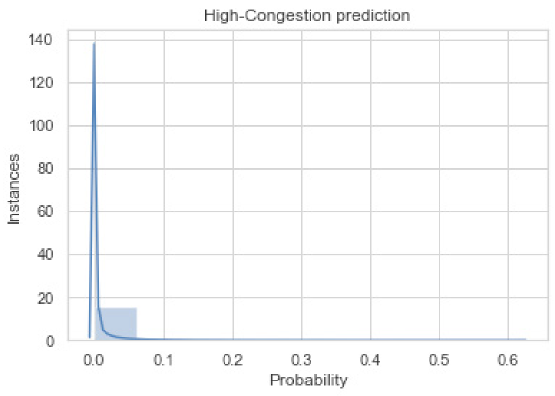

Figure 6. Probability of being heavy-congestion in logistic regression case

The probability distribution graph presents the four level congestion predictions in decreasing order: Non-congestion, Light-congestion, Medium-congestion, and High-congestion. There is a relatively high probability of Non-congestion and less probability of Heavy-congestion for the particular testing dataset.

4.1.2. Decision Tree

From decision tree prediction algorithm confusion matrix can be generated as shown in Table 6. The corresponding values of the matrix calculate evaluation metrics.

Table 6: Python-based decision tree confusion matrix

		Predicted value			
	Congestion level	Non	Light	Medium	High
Actual value	Non	168235	4744	3	0
	Light	1701	36393	198	3
	Medium	0	260	9892	49
	High	0	0	51	947

Four level congestion probability distributions are generated by making testing datasets as input to a decision tree prediction model. The 10 bin histogram-based graphical representation of congestion levels including 'Non-congestion', 'Light-congestion', Medium-congestion', and 'Heavy-congestion' are shown in Fig.7, Fig.8, Fig.9, and Fig.10 respectively.

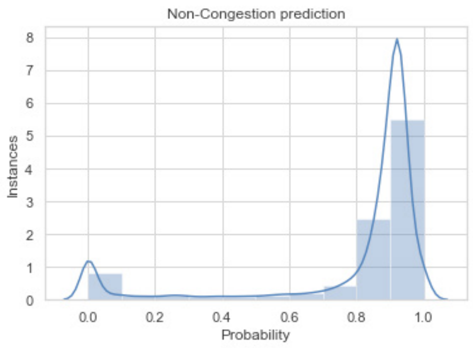

Figure 7. Probability of being non-congestion in decision tree case

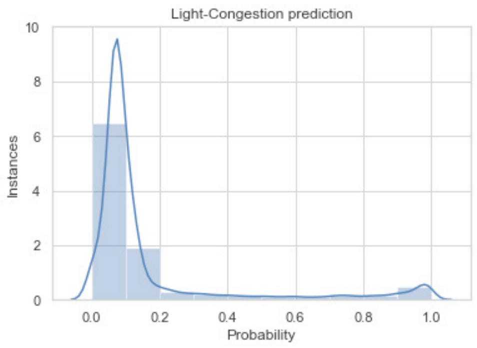

Figure 8. Probability of being light-congestion in decision tree case

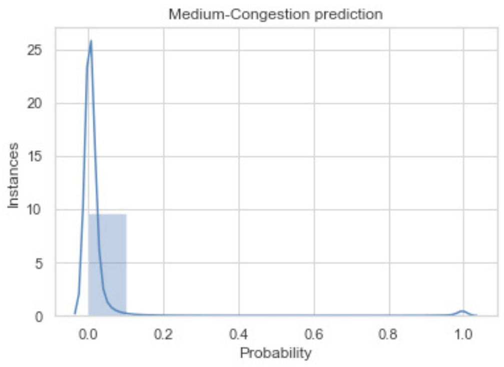

Figure 9. Probability of being medium-congestion in decision tree case

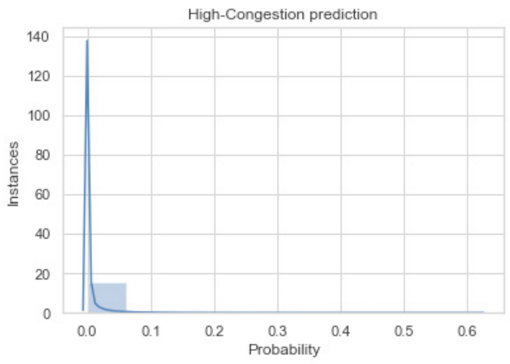

Figure 10. Probability of heavy-congestion in decision tree case

Similar to logistic regression prediction approach, the probability prediction graph presents the four congestion levels in decreasing order: Non-congestion, Light-congestion, Medium-congestion, and High-congestion. Relatively, high probability of non-congestion and less probability of heavy congestion is observed for the particular testing dataset.

The performance of the two congestion prediction models are evaluated by the common metrics including accuracy, precision, recall, and F1_score and summarized in Table 7.

Table 7: performance metrics in python environment

Metrics	Logistic regression	Decision tree
Accuracy	0.93	0.97
Precision	0.92	0.95
Recall	0.85	0.96
F1_score	0.88	0.96

4.2. Modeling in clementine environment

4.2.1. Logistic regression

Logistic regression type of prediction model is designed in clementine environment as shown in Fig. 11. An accuracy of 95.69% is achieved from the Confusion matrix shown in Table 8.

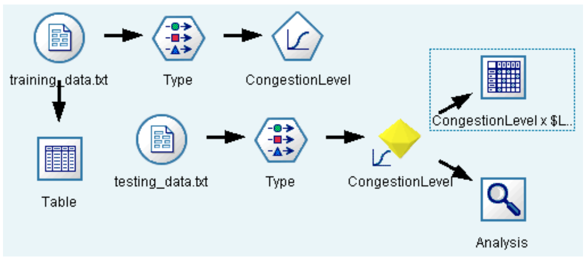

Figure 11. Logistic regression model in clementine environment

Table 8: Clementine-based logistic regression confusion matrix

		Predicted value			
	Congestion level	**Non**	**Light**	**Medium**	**High**
Actual value	**Non**	168449	4533	0	0
	Light	4519	33550	226	0
	Medium	8	268	9911	14
	High	0	0	25	973

4.2.2. Decision Tree

Decision tree prediction model is composed of the following parameter settings; (1) the tree depth is 30, (2) pruning severity is assumed to be 75, and (3) minimum records per child branch is assumed to be 4. Moreover, a total of six hundred twelve (612) decision rules are generated. The clementine model is shown in Fig.12. An accuracy of 97.65% is computed from the confusion matrix shown in Table 9.

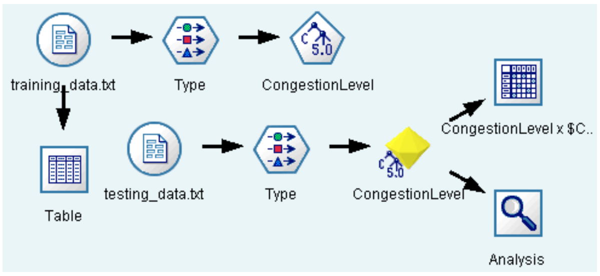

Figure 12. Decision tree model in clementine environment

Table 9: Clementine-based decision Tree confusion matrix

		Predicted value			
	Congestion level	Non	Light	Medium	High
Actual value	Non	169894	3087	1	0
	Light	1634	36485	176	0
	Medium	0	230	9936	35
	High	0	0	65	933

4.2.3. Neural Network

The neural network model is constructed with eight input attributes and one target attribute including the following parameters. (1) the network has input layer of 8 neurons (i.e. eight input attributes), (2) the network has one hidden layer with 10 neurons (i.e. the number of hidden layers and neurons are set by experimental trial), (3) the network has output layer of 4 neurons (i.e. four congestion levels), (3) *trainlm* is used as training function, (4) sigmoid types of activation function is used, and (5) accuracy as performance function as applied. The network model in clementine environment is shown in Fig.13. An accuracy of 93.75% is calculated from the confusion matrix shown in Table 10.

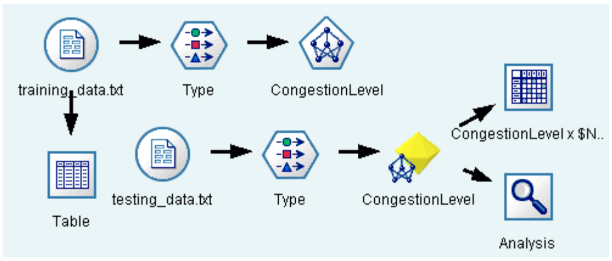

Figure 13. Neural network model in clementine environment

Table 10: Clementine-based neural network confusion matrix

		Predicted value			
	Congestion level	Non	Light	Medium	High
Actual value	Non	168100	4841	28	0
	Light	6943	31019	312	0
	Medium	11	743	9447	13
	High	0	0	21	998

The performance of the three prediction models are evaluated by taking accuracy as a common metrics and summarized in Table 11.

Table 11: performance metrics in clementine environment

Metrics	Logistic regression	Decision tree	Neural networks
Accuracy	0.95	0.97	0.93

5. CONCLUSIONS

This paper presents three types of machine learning algorithms including decision tree, logistic regression and artificial neural network to address the problem of predicting traffic congestion propagation patterns given multivariable traffic flow parameters. Python programming environment is used to design the three congestion prediction models and verified in clementine environment. The models have an excellent contribution to peoples and city planners by providing real-time congestion information. The performance of the entire prediction algorithms is evaluated by common metrics including accuracy, precision, recall, and F1-score. Those metrics are calculated from the corresponding confusion matrix and it is observed that decision tree achieved better prediction capability over the other two prediction models. As a recommendation, the prediction model can be further improved by linking the road condition database with satellite system. Moreover, deep learning based prediction algorithm can be incorporated to improve prediction capacity using massive real-time datasets.

ACKNOWLEDGMENT

This work was supported in part by Chinese Guangdong's S&T project (2019B1515120030), National Natural Science Foundation of China under Grants U1909204, 61701471, 61773381, U1811463, 61773382 & 61872365; Dongguan's Innovation Talents Project (Gang Xiong).

REFERENCES

AN, S., YANG, H., WANG, J., CUI, N. & CUI, J. 2016. Mining urban recurrent congestion evolution patterns from GPS-equipped vehicle mobility data. *Information Sciences,* 373**,** 515-526.

ANTONIOU, C., KOUTSOPOULOS, H. & YANNIS, G. 2007. Traffic State Prediction Using Markov Chain Models. *European Control Conference 2007, Kos, Greece, July 2-5, 2007.*

ANWAR, T., VU, H. L., LIU, C. & HOOGENDOORN, S. P. 2016. Temporal Tracking of Congested Partitions in Dynamic Urban Road Networks. *Transportation Research Record,* 2595**,** 88-97.

D'ANDREA, E. & MARCELLONI, F. 2017. Detection of traffic congestion and incidents from GPS trace analysis. *Expert Systems with Applications,* 73**,** 43-56.

DAVOODI, N., SOHEILI, A. R. & HASHEMI, S. M. 2016. A macro-model for traffic flow with consideration of driver's reaction time and distance. *Nonlinear Dynamics,* 83**,** 1621-1628.

GIDOFALVI, G. 2015. Scalable Selective Traffic Congestion Notification. *The 4th ACM SIGSPATIAL International Workshop on Mobile Geographical Information Systems, Bellevue, WA, USA, November 3, 2015.*: ACM Press.

KONG, Q., ZHAO, Q., WEI, C. & LIU, Y. 2013. Efficient Traffic State Estimation for Large-Scale Urban Road Networks. *IEEE Transactions on Intelligent Transportation Systems,* 14**,** 398-407.

KONG, X., XU, Z., SHEN, G., WANG, J., YANG, Q. & ZHANG, B. 2016. Urban traffic congestion estimation and prediction based on floating car trajectory data. *Future Generation Computer Systems,* 61**,** 97-107.

LIANG, Z. & WAKAHARA, Y. 2014. Real-time urban traffic amount prediction models for dynamic route guidance systems. *EURASIP Journal on Wireless Communications and Networking,* 2014**,** 85.

LIU, B., CHENG, J., CAI, K., SHI, P. & TANG, X. Singular Point Probability Improve LSTM Network Performance for Long-term Traffic Flow Prediction. 2017 Singapore. Springer Singapore, 328-340.

MA, X., YU, H., WANG, Y. & WANG, Y. 2015. Large-Scale Transportation Network Congestion Evolution Prediction Using Deep Learning Theory. *PLOS ONE,* 10**,** e0119044.

QIAN, Z., LI, J., LI, X., ZHANG, M. & WANG, H. 2017. Modeling heterogeneous traffic flow: A pragmatic approach. *Transportation Research Part B: Methodological,* 99**,** 183-204.

WANG, J., HU, F. & LI, L. Deep Bi-directional Long Short-Term Memory Model for Short-Term Traffic Flow Prediction. 2017 Cham. Springer International Publishing, 306-316.