

Predicting Short Term Traffic Congestion on Urban Motorway Networks

Taiwo Olubunmi Adetiloye

A Thesis

In

The Concordia Institute

For

Information Systems Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of

Doctor of Philosophy (Information & Systems Engineering)

Concordia University

Montréal, Québec, Canada

June 2018

© Taiwo Olubunmi Adetiloye, 2018



Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Taiwo Olubunmi Adetiloye**

Entitled: **Predicting short term traffic congestion on urban motorway networks**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information & Systems Engineering)

complies with the regulations of the University and meets the accepted standard with respect to originality and quality.

Signed by the following examining committee:

_____ Chair
Dr. Ion Stiharu

_____ External Examiner
Dr. Luis Antonio

_____ External to Program
Dr. Akshay Kumar Rathore

_____ Examiner
Dr. Jia Yuan Yu

_____ Thesis Supervisor
Dr. Anjali Awasthi

Approved by _____
Dr. Abdessamad Ben Hamza, Director, Concordia Institute for Information Systems Engineering

July 24, 2018

Dr. Amir Asif
Dean, Faculty of Engineering and Computer Science



Abstract

Predicting Short Term Traffic Congestion on Urban Motorway Networks

Taiwo Olubunmi Adetiloye, Ph.D.
Concordia University, 2018

Traffic congestion is a widely occurring phenomenon caused by increased use of vehicles on roads resulting in slower speeds, longer delays, and increased vehicular queueing in traffic. Every year, over a thousand hours are spent in traffic congestion leading to great cost and time losses. In this thesis, we propose a multimodal data fusion framework for predicting traffic congestion on urban motorway networks. It comprises of three main approaches. The first approach predicts traffic congestion on urban motorway networks using data mining techniques. Two categories of models are considered namely neural networks, and random forest classifiers. The neural network models include the back propagation neural network, and deep belief network. The second approach predicts traffic congestion using social media data. Twitter traffic delay tweets are analyzed using sentiment analysis and cluster classification for traffic flow prediction. Lastly, we propose a data fusion framework as the third approach. It comprises of two main techniques. The homogeneous data fusion technique fuses data of same types (quantitative or numeric) estimated using machine learning algorithms. The heterogeneous data fusion technique fuses the quantitative data obtained from the homogeneous data fusion model and the qualitative or categorical data (i.e. traffic tweet information) from twitter data source using Mamdani fuzzy rule inferencing systems.

The proposed work has strong practical applicability and can be used by traffic planners and decision makers in traffic congestion monitoring, prediction and route generation under disruption.

Acknowledgments

I would like to express my sincere appreciation to the many seen and unseen forces that helped guide my doctoral work to a successful completion.

Firstly, to my Creator, the Almighty God, whose mysteries continue to defile all understanding; and who is ever willing to give divine wisdom to them that seek with humility and work diligently to find answers to specific problems.

Secondly, so much gratitude to my supervisor, Prof. Anjali Awasthi, for accepting to supervise my master's and doctoral thesis spanning a good time period of interesting research and development. I remain indebted for your steadfastness in teaching and research, patience and extraordinary erudition in advisory. My immense thanks also to Prof. Satyaveer Chauhan and Prof. Mustapha Ouhimmou as well as the NSERC Value Chain Optimization research group for their academic and financial support at the onset of this doctoral program.

Thirdly, my profound gratitude to Genetec in Montreal, Canada, for providing us support with their software traffic engine.

To the teaching and non-teaching staff from the very start of my education until the present time, I say "Thank you so much".

Dedication

This thesis is dedicated to my dear family:

Parents: Philip Omoniyi, and Catherine Monisola

Siblings: Charles Oluwaseun,

Kehinde Oluyemisi,

Taiwo Tope Richard,

Philip Tomi Kehinde.

Contents

List of Figures	xi
List of Tables	xiv
List of Acronyms	xv
List of Symbols	xvi
Chapter 1: Introduction	1
1.1. Foreword	1
1.2. Data sources for traffic flow	5
1.3. Thesis objectives and contributions	9
1.4. Limitations	11
1.5. Thesis outline	11
Chapter 2: Data mining models for traffic congestion prediction.....	12
2.1. Introduction.....	12
2.2. Problem definition	12
2.3. Literature review	13
2.3.1. Artificial neural networks.....	15
2.3.2. Neuro Fuzzy	17
2.3.3. Deep learning and deep belief network.....	20
2.3.4. Random forest	23

2.4. Solution Approach	24
2.4.1. Experimental setup	26
2.4.2. Selected data mining models	30
2.5. Application results	35
2.5.1. Trend visualization	40
2.5.2. Model verification	43
2.6. Results validation.....	46
2.7. Conclusions.....	48
Chapter 3: Twitter data analysis for traffic congestion prediction	49
3.1. Introduction.....	49
3.2. Problem definition	50
3.3. Literature review	51
3.3.1. Traffic twitter sentiment analysis	51
3.3.2. Traffic twitter cluster classification	52
3.4. Solution approach	55
3.5. Numerical application.....	62
3.5.1. Discussion of results.....	62
3.6. Conclusions.....	65

Chapter 4: Data fusion for traffic congestion prediction	66
4.1. Introduction.....	66
4.2. Problem definition	67
4.3. Literature review	68
4.3.1. Data sources for modeling of traffic congestion	69
4.3.2. Levels of data fusion.....	71
4.3.3. Data fusion architectures	73
4.3.4. Data fusion algorithms for traffic congestion estimation	77
4.4. Solution approach	78
4.4.1. Homogeneous traffic data fusion.....	78
4.4.2. Heterogeneous traffic data fusion.....	79
4.5. Numerical application.....	80
4.6. Results validation.....	86
4.7. Conclusions.....	91
Chapter 5: Conclusions and future works	92
5.1. Conclusions.....	92
5.2. Future works	93
References.....	94
Appendix A.1: Sample of the vehicle count data	111
Appendix A.2: Source code (Spark) –NN for traffic congestion prediction	114

Appendix A.3: Source code (Spark) –RF for traffic congestion prediction	117
Appendix B.1: Agent-based modeling of traffic congestion	120
i. Process modeling library	120
ii. Traffic agents structure.....	121
iii. Single-lane system.....	123
Appendix B.2: Diagram of traffic state manager.....	131

List of Figures

Figure 1.1: Qualitative example of fundamental diagram: (a) Flow-density relationship (fundamental diagram) (b)The speed-density (c) speed space- gap and (d) link-travel-time-flow (source: Kerner [14, 15, 16]).....	4
Figure 1.2: Smart traffic sensors for monitoring traffic congestion. Source: BlipTrack [17]	6
Figure 1.3: Traffic flow simulation using AnyLogic (Adetiloye, and Awasthi [19]) via https://goo.gl/XySrJ4	7
Figure 1.4: Example of probe vehicle systems (Adapted from Sato [21])	8
Figure 1.5: Navstar: GPS Satellite Network. Source: Howell [22]	8
Figure 1.6: Streaming processing pipeline architecture. Adapted from Hortonworks [23] and EndoCode [24].....	9
Figure 2.1: Common types of MFs: (a) Triangular MF ($x; 10, 50, 70$); (b) trapezoidal ($x; 10, 20, 40, 80$); (c) Gaussian ($x; 50, 20$); (d) Generalized bell-shaped ($x; a, b, c$), Sigmoidal $s(x; 0, 1, c)$. (Adapted from Nof [46]).....	18
Figure 2.2: Deep belief network (Adapted from Hinton <i>et al.</i> [67]).....	22
Figure 2.3: Representation of the data mining model with the IVU and TFE.....	25
Figure 2.4: Motorway network of Montreal region (source: Quebec 511 [79]).....	27
Figure 2.5: IRTIR composed of units for the traffic data analysis	28
Figure 2.6: viptraffic model (adapted from [82]).....	29
Figure 2.7: Detection and counts of moving vehicles on the traffic lane.	29
Figure 2.8: BP-NN Traffic Congestion Pipeline Model	30

Figure 2.9: DBN Traffic Congestion Pipeline Model.....	32
Figure 2.10: Traffic Congestion RF Pipeline Model	34
Figure 2.11: Example of predicting traffic congestion using Tensorflow.(Adapted from Adetiloye [90] https://github.com/taiwotman/TensorflowPredictCongestionTypes).....	36
Figure 2.12: Sample predictions of high way traffic using traffic image data taken from some selected regions of New South Wales, Australia.	37
Figure 2.13: Excerpt of vehicle count data	39
Figure 2.14: DBN traffic congestion model	41
Figure 2.15: RF traffic congestion model.....	41
Figure 2.16: NN traffic congestion model	41
Figure 2.17: Overlaying the predictive models with different date/time.....	42
Figure 2.18: Overall predictive models	43
Figure 2.19: Best validation performance: $v(t + 15)$ (NN-predict).....	45
Figure 2.20: Best validation performance: $v(t + 15)$ (DBN-predict).....	45
Figure 3.1: Twitter traffic analytic system.....	55
Figures 3.3. Functional relationship between the Phrase search and Forward-Positional Intersect	60
Figure 3.3: Excerpt of the Montreal traffic tweets.....	63
Figure 3.4: Pie charts showing proportion of the total sentiment	64
Figure 3.5: Traffic delay trending events.....	65
Figure 4.1: Map of the Montreal motorway network (source: GBTTSE [128])	66
Figure 4.2: (near) real-time traffic data for Montreal motorway network (source: GBTTSE [128])	67

Figure 4.3: Data fusion framework (adapted from the JDL, Data Fusion Lexicon [158]).....	72
Figure 4.4: Centralized architecture (source: Castanedo [157]).....	74
Figure 4.5: Decentralized architecture (source: Castanedo [157]).....	75
Figure 4.6: Distributed architecture (source: Castanedo [157]).....	76
Figure 4.7. Homogeneous distributed data fusion for short-term traffic congestion prediction...	78
Figure 4.8: Heterogeneous distributed data fusion for short-term traffic congestion prediction..	79
Figure 4.9: Estimations from the heterogeneous data fusion model.....	80
Figure 4.10: Sample prediction based on RF trained on the Genetec traffic data.	82
Figure 4.11 Training with single hidden layer and multiple hidden layer.....	83
Figure 4.12 Regression plot with R values for NN.....	83
Figure 4.14: Example of calculated TTs from GBTTSE.....	86
Figure 4.15: Example of calculated TTs from RF, DBN, NN, and GBTTSE.....	90
Figure A.1: Traffic state diagram.....	125
Figure A.2: Cartesian map configured based on OSM servers, and Anylogic routing servers employing fastest routing method and integrated with source and sink nodes.....	126
Figure A.3: Measure of traffic flow (at the onset of rush hour)	127
Figure A.4: Measure of traffic flow with the presence of traffic signal controls.	128
Figure A.5: Measure of traffic flow (with emergency vehicle agents in congested traffic).....	130

List of Tables

Table 2.1: Machine learning methods for short-term traffic congestion	14
Table 2.2: <i>RMSE</i> Measurement	43
Table 2.3: Performance measure based on DBN and RF architecture	44
Table 2.4: Baseline performance of classifier algorithm on training data	47
Table 2.5: Baseline performance of classifier algorithm on testing data.....	47
Table 3.1: Traffic twitter sentiment analysis	64
Table 4.1: Data fusion algorithms for traffic congestion estimation	77
Table 4.2: MFRI with twitter sentiment distributed homogeneous model	81
Table A.1. Traffic agent block from AnyLogic Process modeling library.....	121

List of Acronyms

Name	Acronyms
Advanced Traffic Management Systems	ATMS
Automated Traffic Surveillance and Control	ATSC
Artificial Neural Network	ANN
Automated Traffic Recorder	ATR
Back Propagation	BP
Deep Belief Networks	DBN
Deep Learning	DL
Extended Kalman Filter	EKF
Fuzzy Logic	FL
Geographical Information System	GIS
Genetec Blufaxcloud Travel -Time System Engine	GBTTSSE
Global Positioning System	GPS
Input Variable Unit	IVU
Intelligent Transportation System	ITS
Kalman Filter	KF
Machine Learning	ML
Mamdani Fuzzy-Rule Base Inferencing	MRFI
Neuro Fuzzy	NF
Neural Network	NN
Number of Decision Trees	NDT
Part of Speech	POS
Predicted Travel Time	PTT
Random Forest	RF
Root Mean Square Error	RMSE
Stochastic Gradient Decent	SGD
Travel times	TT

List of Symbols

Symbol	Meaning
$f(w)$	Optimization formulation of the loss function
γ	Step size
$L'_{w,i} \in \frac{\partial}{\partial w} L(w, x_i, y_i)$	A member of sub-gradient of the loss function
$R'_w \in \frac{\partial}{\partial w} R(w)$	A member of sub-gradient of the regularizer function $R(w)$
w^{t+1}	Weight over t iteration
s	Initial step-size, s
$p(v, h \theta)$	Encode joint distribution function
$E(v, h; \theta)$	Energy function
$J(C_1, C_2)$	Jaccard similarity or index between set C_1 and C_2
$tf(t, d)$	Term frequency for term t and document d
$idf(t, D)$	Inverse document frequency of term t and whole document D

Chapter 1

Introduction

1.1. Foreword

The growth in transportation and economic activities due in most part to globalization has led to increased traffic flows in metropolitan regions around the world. For example, the traffic index measuring traffic congestion worldwide (TomTom [1]) reported that the city of Montreal could have a morning congestion peak of 47% and evening congestion peak of 57% and the city was ranked 3rd in Canada, and 81st in the world. In Omrani *et al.* [2], it is estimated that between 1985 and 2007 over 60% of workers in the city of Luxembourg and about 30% of cross-border workers commuted daily across the border by diverse travel modes: bike, bus, private car, train, or by foot. This emphasizes the need for better traffic information based on real-time traffic data from monitoring equipment like sensors, GPS broadcast etc. (Hamner [3]; Leshem and Ritov [4]). Traffic congestion, otherwise known as traffic jam, is generally defined as a condition in transport network in which the increased use of road by vehicles in traffic streams creates slower vehicle speeds, time delays, increased vehicular queueing and, sometimes, a complete paralysis of the traffic network. According to Jain *et al.* [5], traffic congestion can be conventionally categorized on the basis of four parameters: capacity, speed, delay/travel time and cost incurred due to congestion. The volume-by-capacity ratio (v/c) is a popular preliminary measure that compares the given traffic congestion with the limiting on-capacity congestion, and is used to assess the Level of Service (LOS) of the road. Speed based measures of congestion provide more effective explanation for the degree of congestion [1]. Lomax *et al.* [6] define congestion in

terms of the travel time or delay incurred in excess to that for free traffic flow. Traffic congestion is characterized not only by massive delays but by enormous cost incurred through increased fuel wastage and money losses, particularly, in cities of developing countries and in almost other cities around the world [7]. Complex, non-linear characteristics with cluster formation and shockwave propagation that deviate from the law of mechanics are widely observed in traffic. To address the problems of traffic congestion would require better traffic information systems with improved reliability of the traffic prediction and building of more infrastructure. Eisele *et al.* [8] observe that in spite of the Advanced Traffic Management Systems (ATMS) that typically monitor and provide information to passenger drivers on the basis of data mainly from passenger cars; there has yet been no means to statistically analyze the difference between the travel time estimates based on intelligent transportation systems (ITS) data of the passenger cars and that of commercial vehicle operations. Their approach seeks to know whether the accuracy in the travel times from the ITS data can be sufficient to replace current data collection techniques. Many researchers have employed Machine Learning (ML) and its variants to analyze traffic congestion data. Agent-based modeling has been used for real-world applications [Appendix B.1-2]. The successes recorded so far encourage further study towards improving the predictive accuracy of the methods.

Kumar *et al.* [9] identify the need to apply a society-wide consensus to resolve traffic related problems. This may require advanced computation and analytics on big data that has been generated in cities (Zhang *et al.* [10]) where building transportation infrastructures to resolve traffic issues can be for limited period only and can be insufficient to relief the traffic pressure with increase in number of urban road vehicles. Awasthi *et al.* [11] model traffic congestion on

the motorway networks using link as basic unit and extended it to network using microscopic traffic flow theory approach.

Kumar *et al.* [9] model traffic congestion using three kinds of data namely historical, real-time, and predicted (short-term forecasting) data. Real-time road traffic data can be obtained from surveillance systems composed of probe vehicles, incident detection systems, and magnetic loop detectors etc. Short-term traffic forecasting is the “process of directly estimating anticipated traffic at a future time, given continuous short-term feedback of traffic information”. These predictions involve seasonality (time series). The use of Artificial Intelligence (AI) technique for short-term traffic forecasting has gained much attention due to the stochastic nature of the traffic flow and the non-linear characteristics of short-term traffic forecasting. Another alternative source is simulation. Simulation allows modeling of real-world traffic situations using computer based models and assists in pro-active decision making.

Random variations in traffic flow create congestion that continuously affects the behavior and free-flow movements of vehicles on motorway networks (Awasthi *et al.* [11]). Rehborn and Klenov [12] presented two approaches to model the random variations in traffic flow namely: “data mining” and the “physics of traffic”. In the “data mining” approach, machine learning techniques like neural network, random forest, support vector machines are applied to generate the reproducible features of measured traffic data for the purpose of identification and analysis; on the other hand the “physics of traffic” tends to understand and explain these reproducible features of traffic as a model of the measured traffic data. The parameters often used for investigation are the rate of traffic flow, link-length, traffic density and average vehicle speed. Kerner [13] illustrates the relationship between these traffic variables using a fundamental diagram (see Figure 1.1a-d).

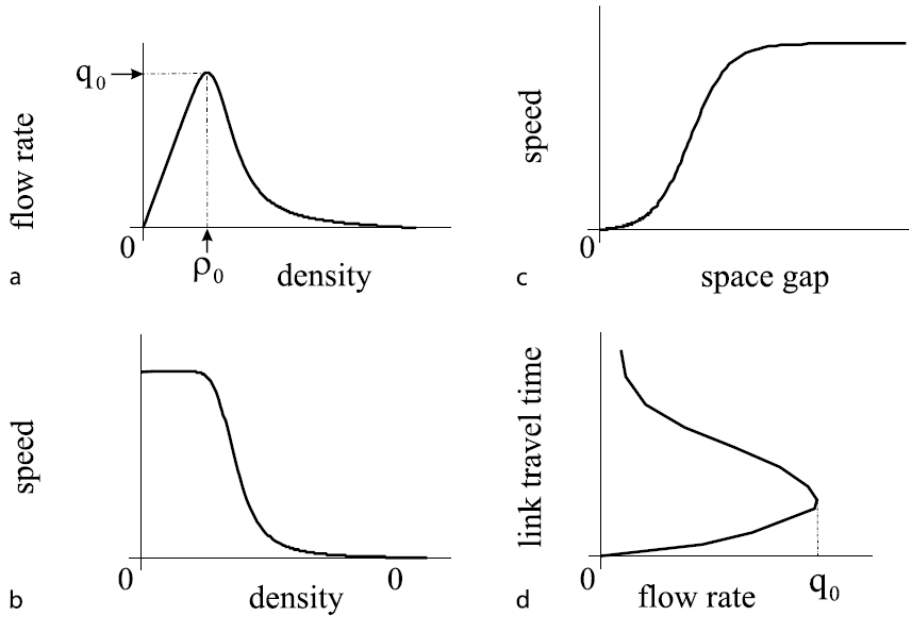


Figure 1.1: Qualitative example of fundamental diagram: (a) Flow-density relationship (fundamental diagram) (b)The speed-density (c) speed space- gap and (d) link-travel-time-flow (source: Kerner [14, 15])

In Figure 1.1, flow (q) is given by the number of vehicles passing a fixed point per unit of time (hrs). The density (k) is the number of the vehicles per unit length (km) of the roadway. The speed (v) is defined as the measurement of the link-travel distance or space gap covered per the unit of time. In practice, it involves measuring the average speed in terms of the time mean speed (v_t) and space mean speed (v_s). This is done by sampling vehicles in a given area over a period of time. The time mean speed is commonly measured at a reference point on the road way over a period of time using loop detectors spread over a reference area in order to track individual vehicle speed. On the other hand, consecutive videos or pictures from satellite, camera or both constitute the data used for calculating the space mean speed measured over the whole road way segment.

The mathematical formulations are given by:

$$k = \frac{1}{s} \quad (1.1)$$

$$q = kv \quad (1.2)$$

$$v_t = \left(\frac{1}{m}\right) \sum_{i=1}^m v_i \quad (1.3)$$

$$v_s = \left(\left(\frac{1}{n} \right) \sum_{i=1}^m \left(\frac{1}{v_i} \right) \right)^{-1} \quad (1.4)$$

And the relationship between v_s and v_t is given by:

$$v_t = v_s + \frac{\sigma_s^2}{v_s} \quad (1.5)$$

Where:

s – spacing between the vehicles.

m – number of vehicles passing the fixed point.

v_i – speed of the i^{th} vehicle.

σ_s^2 – variance of the space mean speed.

1.2. Data sources for traffic flow

Different data sources can be used to obtain traffic flow and congestion-related information such as social media, GPS, probe data, simulation models, sensors etc. These are described as follows:

1. Sensors: They are devices used to measure and analyze travel times in traffic in order to identify congestion patterns, time critical routes and other vital traffic information to optimize traffic flow. The BlipTrack sensors, illustrated in Figure 1.2 was first used in Zurich, Switzerland, to improve traffic with economic values through reduced travel times, less fuel consumption in relation to vehicle emissions. The BlipTrack solution has gained acceptance in the US, New Zealand, and UK (BlipTrack [16]).

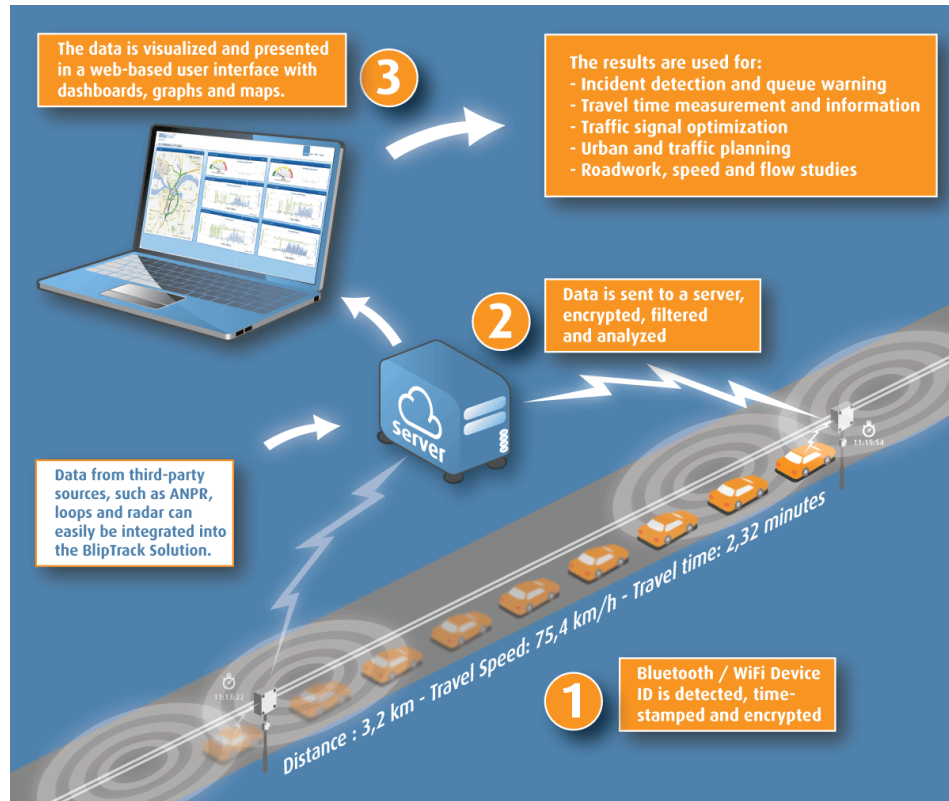


Figure 1.2: Smart traffic sensors for monitoring traffic congestion. Source: BlipTrack [16]

2. Simulation: It requires developing a model that imitates the process of a real-world system over time (Banks *et al.* [17]). It primarily helps to reduce cost and to understand the real world complexity before actual implementation and development of a solution. Simulation has been used extensively in traffic travel time and traffic flow prediction. The models can be created using software tools like: AnyLogic, ArcGIS, MATSIM, SUMO, Repast etc. Figure 1.3 illustrates a traffic simulation using Anylogic.



Figure 1.3: Traffic flow simulation using AnyLogic (Adetiloye, and Awasthi [18]) via <https://goo.gl/XySrJ4>

3. Probe vehicle: According to Young [19], “Vehicle probe technology is emerging as a means of monitoring traffic without the need for deploying and maintaining equipment in the right-of-way. In contrast to speed sensors, vehicle probes directly measure travel time using data from a portion of the vehicle stream.” As illustrated in Figure 1.4, commercial vehicle probe data services primarily include the use of cell phones and automated vehicle location (AVL) data (Young [19]).

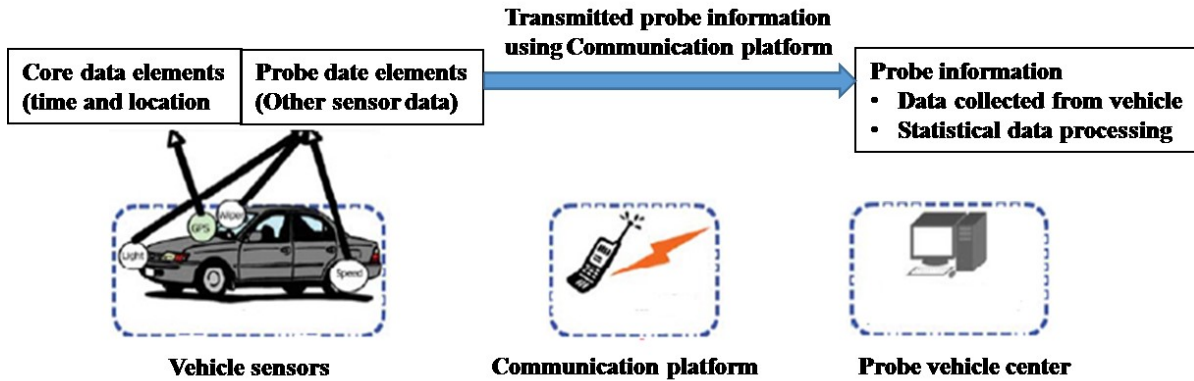


Figure 1.4: Example of probe vehicle systems (Adapted from Sato [20])

4. GPS: It is fully known as the global positioning system or simply Navstar (see Figure 1.5). It is a global navigation satellite system for providing location and travel times information from anywhere on the Earth surface; if there is an unobstructed line of sight to its remote space satellites. It can be used with or without telephonic or internet system in order to enhance its performance (Howell [21]).



Figure 1.5: Navstar: GPS Satellite Network. Source: Howell [21]

5. Social media: It is a good source for streaming real-time big data from online sources such as Twitter and Facebook. Figure 1.6 illustrates an example of stream processing pipeline architecture. It shows the stages involve in streaming of tweets from the input source to the destination using a bolt pipeline architecture. This is in order to achieve the tweets preprocessing, feature extraction, social network generation, sentiment analysis and so on.

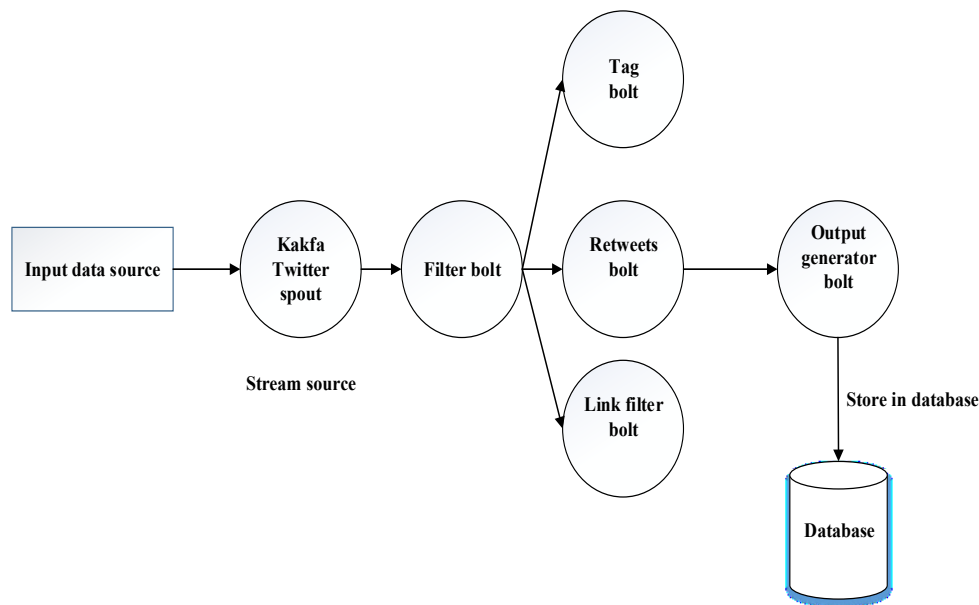


Figure 1.6: Streaming processing pipeline architecture. Adapted from Hortonworks [22] and EndoCode [23]

1.3. Thesis objectives and contributions

The objective of this thesis is to develop a multi-modal (data fusion) framework for short-term traffic congestion prediction on urban motorway networks. This involves:

1. Investigation of three machine learning algorithms namely back-propagation neural network (BP-NN), deep belief network (DBN) and the random forests (RF). A vehicle count traffic classification framework based Intelligent Road Traffic Information Retrieval system has been proposed in Chapter 2. The system can be used for extracting

the cyclic nature of the traffic volume of road vehicles and for the classification of traffic congestion using data mining algorithm. This work resulted in the following publication:

T. Adetiloye and A. Awasthi (2017), "Predicting short-term congested traffic flow on urban motorway networks", In P. Samui, S.S Roy, V.E. Balas(Eds.), Handbook of Neural Computation(pg. 145–165). doi: <https://doi.org/10.1016/B978-0-12-811318-9.00008-9> . Academic Press.

2. In Chapter 3, tweet mining of traffic delays and sentiment analysis and cluster classification to identify congestion pattern has been established. This is derived from a standard model methodology involving tweet crawling and preprocessing steps, feature extraction and social network generation as well as cluster classification. This work resulted in the following publication:

T. Adetiloye and A. Awasthi(2018), "Traffic condition monitoring using social media analytics", In S.S. Roy, P. Samui, R. Deo and S., Ntalampiras (Eds.), Big Data in Engineering Applications, Studies in Big Data. https://doi.org/10.1007/978-981-10-8476-8_13. Springer Nature Singapore Pte Ltd.

3. In Chapter 4, a multi-modal distributed big data fusion framework for predicting traffic congestion has been developed and experimentally validated with results obtained from the Genetec BlufaxCloud travel times' engine; and for the first time in the literature. This work resulted in the following publication:

T. Adetiloye and A. Awasthi(2018), "Multimodal big data fusion framework for traffic congestion prediction", In S.K. Seng, L. –m. Ang, A.W.C. Liew and J.Gao(Eds). Multimodal Analytics for Next-Generation Big Data Technologies and Applications, Springer.

1.4. Limitations

1. Our distributed data fusion architecture can be said to be in the initial application development stage. Like any software, minor, and major fixes are done overtime; sometimes this could take several years to reach a satisfactory software solution.
2. Issues such as latency due to increase in the system bandwidth and the computational runtime may be expected. Latency is widely defined as the time taken for a packet of data to travel to its destination. Also, there is the data quality from the source, such as from road cameras positioned on the various segments of the urban motorway network, which tend to affect the overall accuracy of prediction.
3. Generally, the computation speed when performing data analytics on single computer machine are often very slow because of its memory and disk size. Hence, distributed cloud computing using cluster(s) of machines are generally recommended for an advanced big traffic data analytics.

1.5. Thesis outline

The contents of this thesis are organized as follows:

Chapter 1 introduces the thesis.

Chapter 2 presents data mining algorithms for traffic congestion prediction.

Chapter 3 presents twitter data using framework for traffic congestion prediction.

Chapter 4 presents multimodal data fusion approach for traffic congestion prediction.

Chapter 5 draws the conclusions and provides directions for future works.

Chapter 2

Data mining models for traffic congestion prediction

2.1. Introduction

Traffic managers involved in maintaining or monitoring traffic systems need good predictive analytic models for quick evaluation of information they gather on drivers' naturalistic behaviors, traffic patterns, traffic origin of atmospheric pollutions and a whole lot more under real-practical situations. They are also concerned about the predictive accuracy when statistically compared to the actual occurrence in the future.

2.2. Problem definition

In this chapter, we investigate three data mining algorithms for modeling short-term traffic congestion on urban motorway networks. They can be classified into two main categories: neural networks, and random forest classifiers. The neural networks considered are back propagation neural network, and deep belief network. First, we develop various models based on these algorithms. Second, individual model is trained using the traffic input variables while comparatively evaluating their performances with the testing sets and also measuring their sensitivities. Based on preliminary experimental tests, we are of the opinion that these algorithms can offer a reliable and effective means of predicting short term traffic congestion towards better traffic management.

2.3. Literature review

The prediction of traffic congestion has been towards traffic management and traffic information systems using diverse predictive algorithms like neural network, ensemble algorithms e.g. random forests, or hybrid predictive algorithms. As presented in Eisele *et al.* [8], ITS technologies and infrastructures enhance accurate travel-time mean and variance estimates from reliable data sources. In traffic, of practical interest is the congestion in relation to the associated peak period that often forms the basis for traffic data collection.

Also, the prediction of traffic congestion aims to influence travel behavior, improve mobility, and save energy while serving as a vital component of ITS to assist drivers in averting potential traffic blocks or by traffic management and control systems to ensure free flow of traffic (Zhang *et al.* [24]). One aspect of the traffic management system as detailed in Baskar *et al.* [25] is monitoring the vehicle speed and the number of vehicles that enter and exit the highway segment on ramps and exits. Vehicles stay in one lane unless there is an accident blocking their assigned lane, in which case they go around the accident site. The traffic management model resolves three different cases: (1) an uncontrolled system as a reference case, (2) a controlled system with human drivers, and (3) a controlled system with intelligent vehicles, which is platoon based. In an automated highway system (AHS), every vehicle is presumed to be intelligent.

As described by Kerner [14], free flow traffic transitions to congested traffic when the average speed of vehicles is lower than the minimum average speed that is expected in free flow. Rehborn and Klenov [12] observe that the spatiotemporal congestion patterns develop mainly from freeway bottlenecks at motorways (e.g. on-off ramps, roadworks, decreasing freeway, lanes and so on) and that the patterns emerge from the onset of congestion in an initially free-flowing

traffic in space and time. Increased frequency and severity of congestion has led to increased investments in the development of traffic management techniques (Lyons *et al.* [26]). This has been greatly influenced by rapid improvement in fast computers and flexible mathematical methods (Kumar *et al.* [9]). A general review of network traffic analysis and prediction techniques can be found in Joshi and Hadi [27]. Table 2.1 summarizes the approaches for short-term traffic congestion.

Table 2.1: Machine learning methods for short-term traffic congestion

Approach	Author	Title	Comments and plausible limitations
Neural Networks(NN)	Dougherty and Cobbert [28]	Short-term inter-urban traffic forecasts using neural networks	The performance of NN in prediction of traffic flow and occupancy show some promising results; however its 'black box' attribute can make it difficult to interpret. Future work should look in the direction of adaptive neural network, for example, using recurrent back propagation NN and alternative methods.
Support Vector Machine(SVM)	Theja and Vanajakshi [29])	Short-term prediction of traffic parameters using support vector machines technique	SVMs is used for prediction of short-term traffic flow based on such variables: speed, volume, density, travel-time, headways etc. under mixed and less lane disciplined traffic congestion. A sensitivity analysis of SVM and ANN in terms of accuracy and runtime showed that SVM could be considered a viable alternative for prediction of traffic congestion.
Random Forest	Zarei <i>et al.</i> [30]	Road traffic prediction using context-aware random forest based on volatility nature of traffic flows	A scheme for differentiating between peak and non-peak traffic flow using context-aware RF is considered to be effective and scalable in short-time traffic prediction. Limitation: The time dependence of the traffic data should be investigated prior to inputting the data into the model
Dynamic Time Warping algorithms(DTWA)	Hiri-O-Tappa <i>et al.</i> [31]	Development of real-time short-term traffic congestion prediction method	Dynamic time warping algorithms may have better accuracy than traditional time-series predictive algorithms based on the evaluation on some time series traffic data. Limitation: Noise in the raw data causes low accuracy.
Genetic Algorithm(GA) and Cross-Entropy(CE)	Lopez-Grazia <i>et al.</i> [32]	A hybrid method for short-term traffic congestion forecasting using Genetic Algorithms and Cross Entropy.	The comparative evaluation showed that combination of both method is better than individual GA and CE in optimization of a Parallel Hierarchical Fuzzy Rule-Based System of short-term traffic congestion forecasting. Limitation: Unknown optimization performance when the method is used in combination with other techniques.

Moving average (MA), exponential smoothing (ES), autoregressive MA (ARIMA), and neural network (NN) models	Tan <i>et al.</i> [33]	An aggregation approach to short-term traffic flow prediction	The forecast of short-term traffic flow using aggregated approach know as Data Aggregation (DA) outperform individual algorithms: MA, ES, ARIMA and NN. Limitation: Further investigation may be needed to access accuracy of DA for specific applications.
--	------------------------	---	--

In the following sections, we will conduct literature review pertaining to the data mining techniques namely artificial neural networks, neuro fuzzy, deep learning and deep belief networks, as well as random forest. Thereafter, we will present our proposed methodologies, application results and discussion as well as the conclusion.

2.3.1. Artificial neural networks

The main idea behind artificial neural networks (ANN) is its architecture that mimics the way the brain works with interconnections of neurons. In machine learning and cognitive science, it is generally defined as the “collection of simple, nonlinear computing elements, whose inputs and outputs are tied together, to form a network” –Rumelhart *et al.*, [34]). There are many different types of ANN. One is the back propagation (BP) neural networks algorithm, widely used due to its fast computing power and that it can rapidly solve problems which had been previously insolvable (Nielsen [35]). Others are the hopfield networks, kohongen networks and the adaptive resonance networks – Barga *et al.* [36]. Hybrid models that comprise of mixture of NN and other algorithms like fuzzy logic, partial least squares and so on, have been proposed to improve predictive accuracy. According to Lyons *et al.* [26], the modeling technique offered by ANN (or simply NN) makes it distinctly different from more conventional approaches in respect of its accurate modeling; and especially its successes in solutions to problems hitherto lacking an appropriate modeling technique such as commonly found in the traffic data of urban network traffic environment. This makes it appealing to many researchers in diverse fields including

transportation. Moreover, NN are well-suited for modeling and predicting traffic parameters regardless of underlying non-linear relationships and prior knowledge of its functional form (Vlahogianni *et al.* [37]).

Gilmore and Abe [38] extended the work on the traffic management with neural network by Gilmore *et al.* [39]. They proposed two ATMS functions incorporated within neural networks for signal light control systems to adaptively optimize traffic in urban areas; and also, provide accurate prediction of traffic congestion. Furthermore, the system relied on information on street segment capacities, traffic flow rates, and potential flow capacities to enhance the performance of an Automated Traffic Surveillance and Control (ATSC) that uses responsive control in the designated areas of Los Angeles (Rowe [40]). Their approach seeks improvement of the systems to eliminate traffic jams and gridlocks with creation of intersection specific Hopfield energy functions (three-way and four-way intersections, four-way stops, etc.). The Hopfield energy functions are used to train the NN and proven to be more effective compared to the back-propagation, which is capable of good learning behavior but may be time-consuming for large systems in some cases. Dia [41] presented an object oriented neural network approach for short-term traffic forecasting with substantial improvements on conventional model performance and feasibility of the approach for traffic prediction. Theja and Vanajakshi [29] compared the performance between Support Vector Machine (SVM) and the BP-NN while using sensitivity analysis to determine optimum performance in terms of accuracy and runtime. Major issues to avoid in a NN are under-fitting and over-fitting of the data that may introduce bias such that in under fitting, the predictive factors become too complex for a small number of nodes to capture; whereas, in overfitting, there can be reduction in its generalization capability.

2.3.2. Neuro Fuzzy

Neuro Fuzzy (NF) refers to a combination of NN and Fuzzy Logic (FL). The idea of FL was advanced by Zadeh [42] and NF was proposed by Jang *et al.* [43]. The following subsections present the concepts of FL and NF. The NF, also widely called Fuzzy Neural Network, is composed of FL and NN.

FL is the approach of processing data based on “degree of truth” instead of the usual binary “1” or “0” (Truth or False) that modern scientific computing is based on. It has been very much used in the areas of control systems and artificial intelligence to handle the concepts of partial truth where the truth may range between two extremes of completely true and completely false; and when provided with linguistic variables, the degree can be managed by the membership functions (MF): a curve that defines how each points in the input space, known as the “Universe of Discourse” is mapped to degree of membership between 0 and 1. Linguistics variables as explained in Zadeh [44] refers to values whose variables are words or sentence in a natural or artificial language (e.g. ‘link’ could be more of linguistics variable rather than its value numerical). There are the trapezoidal, triangular, Gaussian and generalized bell membership functions among others. Figure 2.1 describes the common membership functions.

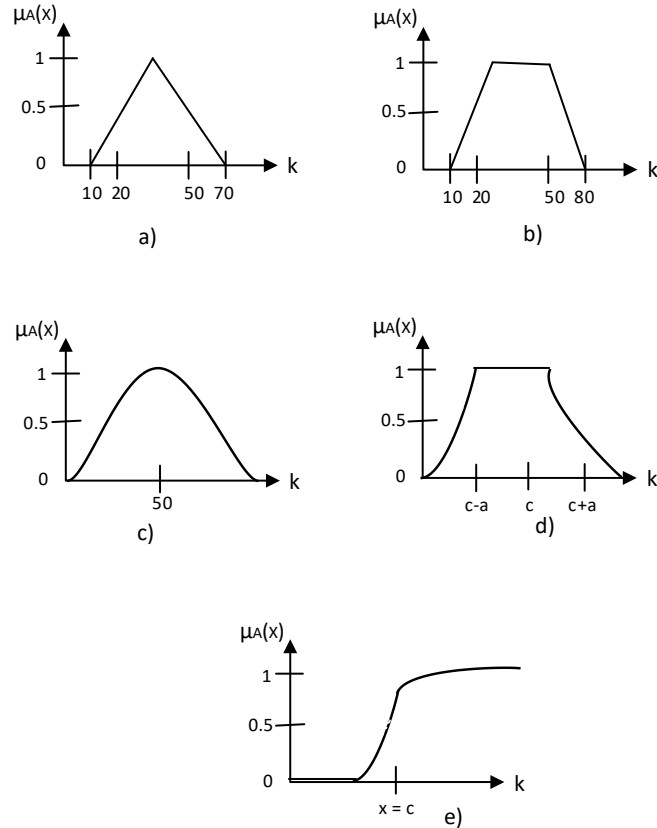


Figure 2.1: Common types of MFs: (a) Triangular MF (x ; 10, 50, 70); (b) trapezoidal (x ; 10, 20, 40, 80); (c) Gaussian (x ; 50, 20); (d) Generalized bell-shaped (x ; a , b , c), Sigmoidal $s(x; 0, 1, c)$. (Adapted from Nof [45])

The NF can be viewed as a 3-layer feedforward neural network that has the first layer consisting of the input variables, the middle layer, otherwise known as the hidden layer, having the fuzzy rules and the third layer being the output variables; and the fuzzy sets are encoded as (fuzzy) connection weights. Zhou and Quek [46] proposed a five layer NF architecture known as the Pseudo Outer Product-based Fuzzy Neural Network (POPFNN) where the fuzzy sets are represented in the units of the second and fourth layer. It consists of three phase of learning: fuzzy membership generation, fuzzy rule identification and the supervised fine-tuning.

Essentially, the research goal in NF is to achieve a good level of interpretability and accuracy. The Mamdani fuzzy structure (Takagi and Sugeno [47]) has the linguistic fuzzy model

that is focused on interpretability while Takagi-Sugeno-Kang –TSK (Takagi and Sugeno [47]; Sugeno and Kang [48]) structure is the precise fuzzy model mainly focused on accuracy. Schnitman *et al.* [49] explained that while the output of TSK fuzzy structure is computed with a very simple formula (weighted average, weighted sum); the Mamdani fuzzy structure requires higher computational effort because of the requirement to compute a whole membership function which is then de-fuzzified. This makes the TSK approach more useful despite the good interpretability when dealing with uncertainty by the Mamdani fuzzy reasoning. For the mathematical formulations of the TSK and Mamdani fuzzy structures, please refer to Schnitman *et al.* [49].

Ling *et al.* [50] propose a model for urban road network traffic congestion forecast based on probe vehicle technology, FL and BP-NN. Their idea seeks to combine multiple FL reasoning with a three layer BP-NN to estimate congestion possibility, level of congestion and the forming time of the link based on the road network topology. Li *et al.* [51] propose a Type-2 FL approach for short-term congestion prediction with the argument that it could be powerful in handling uncertainties especially due to measurements and data used to calibrate the parameters. Thus, they considered associating the membership function corresponding to a particular traffic state with a range of values that can be characterized by a function that reflects the level of uncertainty. Consequently, they were able to use the day-to-day traffic information with real-time traffic information to construct fuzzy rules for traffic prediction. Lu and Cao [52] introduce a FL method to evaluate the level of congestion (LOC) using traffic flow information. It utilizes a continuous variable to express the situation from free flow to traffic jam that is adaptable to their sensory evaluation. An inbuilt fuzzy inference system is implemented with mean velocity and density as inputs. For the evaluation of road traffic congestion, Ogunwolu *et al.* [53] present

a NF approach to vehicular traffic congestion prediction for a metropolis in a developing country. Park [49] put forward a hybrid NF application for short-term freeway traffic volume forecasting consisting of two components: a fuzzy C-means (FCM) method, which classifies traffic flow patterns into a couple of clusters, and a radial-basis-function (RBF) neural network, which develops forecasting models associated with each cluster.

2.3.3. Deep learning and deep belief network

Since 2006, deep learning has evolved as a class of machine learning methods with successful applications in various fields like automatic speech recognition, classification tasks, natural language processing, dimensionality reduction, object detection, motion modeling etc. (Bengio *et al.* [54]; Collobert and Weston [55]; Hinton and Salakhutdinov [56]; Hinton and Sejnowski [57]; Huval *et al.* [49]). Its algorithms are based on the architecture of hierarchical explanatory factors and distributed representations where a cascade of many layers of nonlinear processing units are used for the supervised or unsupervised learning of feature representations per layer, with the layers forming a hierarchy from low-level to high-level features; in the sense of feature extraction and transformation (Deng and Yu [58]). It is inspired by some loosely established interpretation of information systems and communication patterns formulated on the human nervous system; which, attempts to model high-level abstractions in data using a deep graph with multiple processing layers. Some notable architectures of deep learning include the deep belief networks, convolutional neural networks, and recurrent neural networks (Bengio *et al.* [59, 54]; Friedman [60]; Hinton [61]; Schmidhuber [62]). A detailed discussion of deep learning in neural networks can be found in Schmidhuber [62]. Lv *et al.* [63] presents a deep learning approach for traffic flow prediction with big data by means of accurate and timely

traffic flow information while considering the spatiotemporal correlation pattern. Ma *et al.* [64] introduce a large-scale transportation network congestion evolution prediction that relies on a deep Restricted Boltzmann Machine and Recurrent Neural Network architecture using Global Positioning System (GPS) data from taxi.

In machine learning, DBN is a multilayered probability generative model composed of simple learning modules, called Restricted Boltzmann Machines (RBM) (Hinton [61]) very similar to an autoencoders [65] where each subnetwork's hidden layer serves as the visible layer for the next (Bengio *et al.* [59]; Hinton et al [66]). An RBM implies the absence of the lateral connections in the visible and hidden layers such that the random variables encoded by the hidden units are conditionally independent given the states of the visible units, and vice versa (O'Connor *et al.* [67]). In Teh and Hinton [68], RBM is defined as “an undirected graphical model in which visible variables (v) are connected to stochastic hidden units (h) using undirected weighted connections”. The architecture of DBN can be much more efficient than shallow architectures as contained in the single latent layer of feedforward and BP-NN with many levels of non-linearity and highly-varying functions. Figure 2.2 illustrates the DBN architecture.

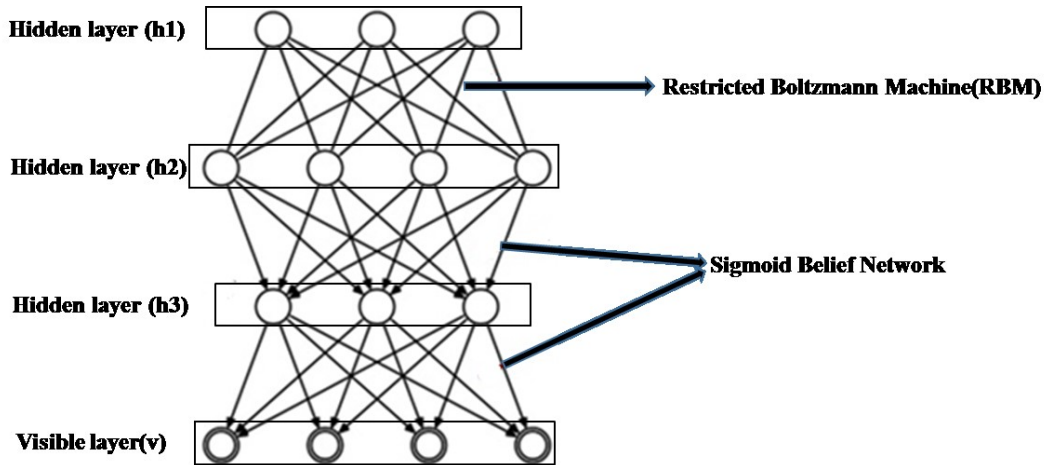


Figure 2.2: Deep belief network (Adapted from Hinton *et al.* [66]).

Training the deep layer networks one layer at a time using greedy algorithm rather than the gradient-based optimization often used with BP-NN can guarantee a good local optimal (Bengio *et al.* [59]) though with the BP-NN, feasible solution can be reached if starting in the neighborhood of a good local optimal. Also, BP-NN is liable to poor performance and prone to overfitting (Hochreiter *et al.* [69]). Moreover, despite the number of NN that exist, finding one that precisely model a given training set can be an NP complete problem (Schmidhuber [62]; Blum and Rivest [70]; Judd [71]).

In traffic congestion prediction, Huang *et al.* [72] investigate the use of deep learning approaches for features extraction and selection without any prior knowledge. Its stack architecture used for traffic flow prediction of a single road is comprised of RBM stacks constituting the DBN for the unsupervised feature learning having sigmoid regression layer on top. This came out of concern about the shallow architectures of neural network attributed to the single hidden layer; hence introducing the key idea of using greedy layer-wise training with stacked RBM and subsequent fine tuning according to the architecture of the DBN guaranteed near 3% improvements.

2.3.4. Random forest

Breiman [73] initially introduced random forest as an ensemble classifier tree learner. This algorithm consists of many decision trees such that each tree depends on the values of random vector grown from a bootstrap aggregated sample in the original training set with the same distribution for the individual trees (Liaw and Wiener [74]). It is used mainly because it is not overly influenced by noise and for its additional features such as effectiveness in multiclass classification and regression tasks, is comparatively fast to train and predict, has fewer tuning parameters, generalizes well with its built-in error estimator, framework for high-dimensional problems, handling of missing values and easy to implement in parallel computation and, from the statistical viewpoint, provides good measures of variable importance and visualization (Cutler *et al.* [75]).

In Hamner [3], an ensemble of RF was trained using different preprocessing techniques to predict future automated traffic recorder (ATR). It posits that performance could be hampered by noise, faulty ATR measurements and temporary or permanent obstruction to traffic flow. It recommends other measures to improve traffic congestion prediction such as data on time of day, the weather, real-time GPS data and the status of local stoplights; in addition, to finding appropriate number of ATRs, which are computationally feasible as with the number of RFs trained on the increasing feature sets. Thus, having proper feature selection techniques, and alternative approaches could make computation of thousands of datasets, like the ATR, feasible. The approach by Leshem and Ritov [4] details the use of hybrid method that combines adaboost algorithm with random forests as weak learner. Applying such hybrid approach helps to improve the overall performance of the hybrid algorithms; with resultant improvements in the quality of traffic prediction. This became evident in their promising results with significant reduction of the

error rate on both simulation and real-world environment. In future, some optimization mechanisms might be necessary for seamless integration of such hybrid algorithms in order to minimize their individual complexity.

2.4. Solution Approach

All the selected data mining models have the same input variable unit (IVU) and traffic flow estimation (TFE). One way to model traffic congestion patterns is by predicting the short term traffic flow based on the volume or number of vehicles (V) passing a road section within a specific travel time interval(t) usually measured in hours(hr) or minutes(mins) e.g Smith and Demesky [76]. Figure 2.3 illustrates the generalized diagrammatic representation of our data mining model with the IVU having input variables which consist of $V(t)$, $V(t - 15)$, $V_{hist}(t)$ and $V_{hist}(t + 15)$. These independent input parameters are defined as current volume and historical (subscript *hist*) average volumes with respect to the time under observation. In this case, the TFE predicts traffic congestion as high, medium or low congestion by estimating the model output within 15mins intervals. Our classification models using NN, RF and DBN seek to estimate the traffic congestion based on the number of vehicles or volume in a future time. Previous approach (Smith and Demesky [76]) seek to predict short term traffic flow using NN and non-parametric regression approaches. In order to generate volume predictions, the mapping model instinctively assume that there is a functional relationship between the given input variables and the prediction variable $V(t + 15)$, while using a clustering model to extract the cyclic nature of the traffic volume by utilizing the past cases having much similarity to the current one. Our contribution goes beyond the current approach by exploring the cyclic patterns while mapping the predictive value to either of the following class labels: low, medium and high

congestion such that the volume range $[25\%, C]$ – high congestion, $[10\%C, 25\%C]$ – medium congestion and $[0, 10\%C]$ – low congestion, where C represents the capacity of the road.

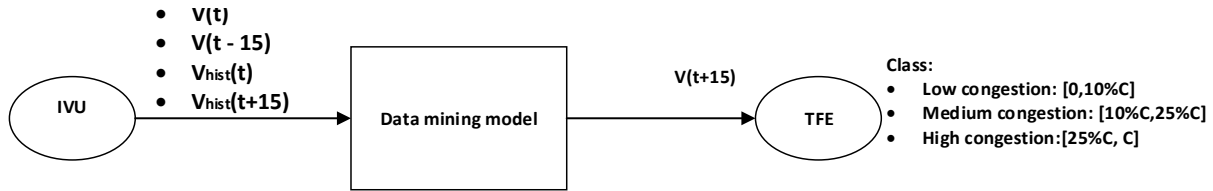


Figure 2.3: Representation of the data mining model with the IVU and TFE.

The following equations by Smith and Demesky [76]) provide a formal statement of the problem:

$$\begin{aligned}
 & \text{Predict } V(t + 15) \\
 & \text{Given } V(t) \\
 & \quad V(t - 15) \\
 & \quad V_{hist}(t) \\
 & \quad V_{hist}(t + 15)
 \end{aligned}
 \tag{2.1}$$

Our revised version of this formal statement is defined as:

$$\begin{aligned}
 & \text{Predict } V(t + 15) \\
 & \text{Given } V(t) \\
 & \quad V(t - 15) \\
 & \quad V_{hist}(t) \\
 & \quad V_{hist}(t + 15)
 \end{aligned}
 \tag{2.2}$$

Predict class label
 Low
 Medium
 High

While, regression models are often used for various mapping applications, it can be difficult for defining the highly non-linear functional form of traffic flow. Hence, the main reason for the non-attractiveness of regression models (Zhang *et al.* [77]) and the need to consider other modeling techniques such as classification and clustering as well as investigate the accuracy of different algorithms to improve the short term traffic flow prediction within these modeling frameworks.

In the following subsections, we discuss:

- **Experimental setup:** This section describes the data and introduces the complete setup of an Intelligent Road Traffic Information Retrieval (IRTIR) system. The system helps in extracting traffic data from the camera images and videos which are processed, retrieved and persisted to the computer system storage for data analysis.
- **Selected data mining models:** Here, we discuss the NN, RF and DBN with details of the features and configurations.

2.4.1. Experimental setup

2.4.1.1. Data description

We relied on the Quebec511 [78] website to obtain the traffic data comprising of images and videos for our study. The data collection starts from Monday to Sunday during the time period of 10:00am to 11:00am over a month period. We focus on the Montreal region. Figure 2.4 illustrates areas within the region with cameras on the road sections.



Figure 2.4: Motorway network of Montreal region (source: Quebec 511 [78])

The sections are the Montreal Island, North shore, and South shore. The Montreal Island has cameras covering Autoroute Decarie (aut. 15), Metropolitaine (aut. 40), Transcanadienne (aut. 25), autoroute 20, Ville-Marie (aut. 720) and Route 112. In the North Shore, the cameras are positioned along autoroute des Laurentides, Papineau, Felix-Leclerc, Chomedey and Route 138. For the section of the South shore, the cameras cover autoroute des Cantons-de-l'Est (aut. 10), Jean-Lesage (aut. 20), autoroute 15 and boulevard Taschereau (route 134). These cameras record real-time traffic while the website updates its information every 5mins. However, this web service system lacks the predictive intelligence to understand the data.

2.4.1.2. IRTIR Framework

Figure 2.5 illustrates the Intelligent Road Traffic Information Retrieval (IRTIR) system composed of four main units for the traffic data analysis. The system's input traffic data is composed of images and videos from camera positioned on the road segments.

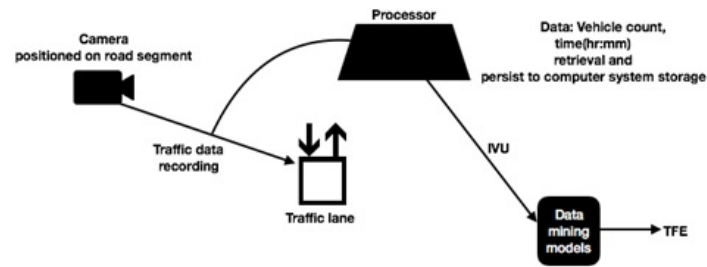


Figure 2.5: IRTIR composed of units for the traffic data analysis

The IRTIR also has a processor algorithmic unit that detects vehicle objects and obtains the counts within the view area of the camera. This was implemented in MATLAB [79] using a Simulink vehicle counter called *viptraffic*. It employs Gaussian mixture models (GMMs) to estimate the background and produce a foreground mask using foreground detector model. This is in order to estimate the video sequence while highlighting the moving vehicle. The GMMs serve as a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. This is on the basis of incorporating information about the covariance structure of the data and the center of the latent Gaussians [80]. Full application details of this model for detection and counts of vehicles can be found on Mathworks [81]. Figure 2.6 shows the *viptraffic* Simulink diagrammatical representation. Figure 2.7 illustrates the detection and counts of moving vehicles on the traffic lane.

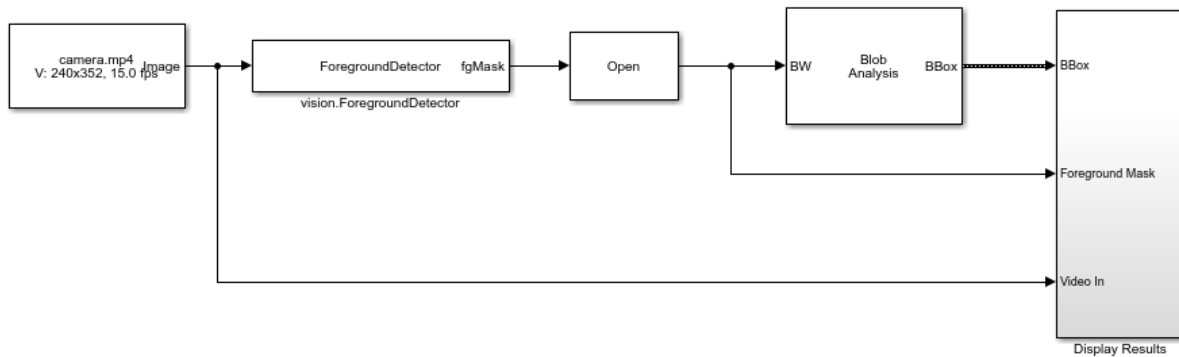


Figure 2.6: *viptraffic* model (adapted from [81])

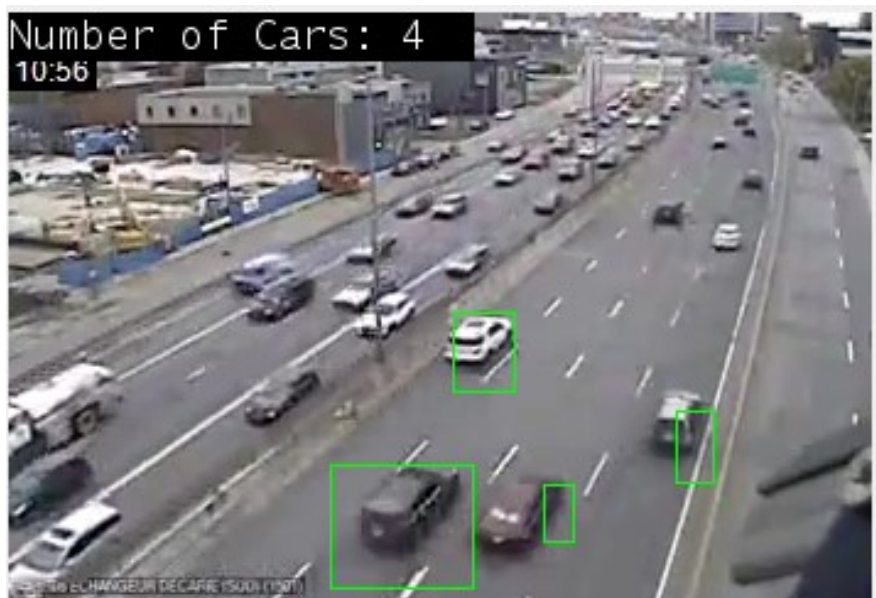


Figure 2.7: Detection and counts of moving vehicles on the traffic lane.

Thereafter, traffic data obtained from the processor unit serves as input variables to the data mining models. Note that data was split in the ratio of 70 to 30 to make the training and test data.

2.4.2. Selected data mining models

2.4.2.1. Backpropagation neural network

Figure 2.8 shows the BP-NN pipeline model with the main traffic input parameters assigned for processing by the stochastic gradient descent (SGD) optimization method. The weight adjustment over t iteration eventually decreases value of the stochastic sub-gradient, $f'_{w,i}$, to good local minimum value taking into considerations $L'_{w,i} \in \frac{\partial}{\partial w} L(w, x_i, y_i)$ a member of sub-gradient of the loss function. The lost function gives a measure of reaching the optimal solution to the problem. This is defined based on the gradient descent with regards to the regularization $\gamma R'_w$ consisting of the step size γ and the sub gradient of the regularizer function $R(w)$.

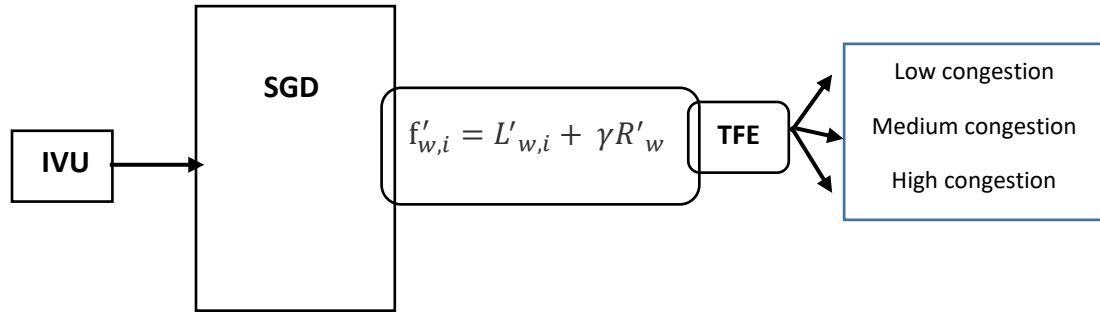


Figure 2.8: BP-NN Traffic Congestion Pipeline Model

By experimentation, an optimal architecture with proper parameter settings without time-consuming process and with less memory usage, which depends on the data size, can be obtained with adequate momentum for convergence (Lu *et al.* [82]). For our training set, we set the number of neuron units in the hidden layer to 7, with decay rate of 0.3. The value chosen for the maximum iteration is based on experimentation. The maximum iteration is set to 860 to allow for adequate momentum for convergence and in order to find a global optimal solution. The SGD is

selected as the optimization method used with the back-propagation to minimize the loss function. A forward computation utilizing an activation sigmoid function from the input layer towards the output is performed in order to find a best fit between the expected and actual output. In instance where the bias as measured by the Root Mean Square Error (RSME) is not well reduced, forward and backward computations may be required over number of iterations. The optimization formulation of the loss function based on Bach and Moulines [83] is defined as:

$$f(w) := \gamma R(w) + \frac{1}{n} \sum_{i=1}^n L(w, x_i, y_i) \quad (2.3)$$

The function $f(w)$ can be representative of at least one data point $i \in [1 \dots n]$ choosing uniformly at random with step size γ upon which we can obtain a stochastic sub-gradient defined with respect to the weight, w :

$$f'_{w,i} := L'_{w,i} + \gamma R'_w \quad (2.4)$$

where $L'_{w,i} \in R^d$; while, considering the weight and the parameter space. And $R'_w \in \frac{\partial}{\partial w} R(w)$ being the sub-gradient of the regularizer function $R(w)$, which is independent of the data point $i \in [1 \dots n]$. Given the parameter settings for the step size with the gradient and regularizer, it is possible to achieve a negative $f'_{w,i}$ such that the weight over t –iteration:

$$w^{t+1} := w^t - \gamma f'_{w,i} \quad (2.5)$$

The SGD is limited by the appropriate choice for the parameter setting. However, default implementation of the step-size can be gotten from $\gamma := \frac{s}{\sqrt{t}}$ decreasingly in t^{th} -iteration with the initial step-size, s , as input parameter. Notice that the step-size is computed with the square root of the iteration counter, t , in order to reduce the speed towards the direction of the steepest descent and to avoid missing the desired local minimum of the function (Kiwiel [84]). Although, it has the limitation of being relatively slow to reach a local optimum; its ability to solve not only linear problems but non-linear ones makes it an ideal optimization method which together with back-propagation helps in minimizing the loss function by adjustment of weights in the network.

2.4.2.2. Deep belief network

While considering the DBN in the classification tasks of traffic congestion with the binary RBM, we assigned the visible input unit, v_i assigned to the IVU, as elements of the encoded joint distribution function (See Figure 2.9). We used three separate hidden layers with each layer connected to the other such that number of neuron units is 32, with decay rate of 0.1.

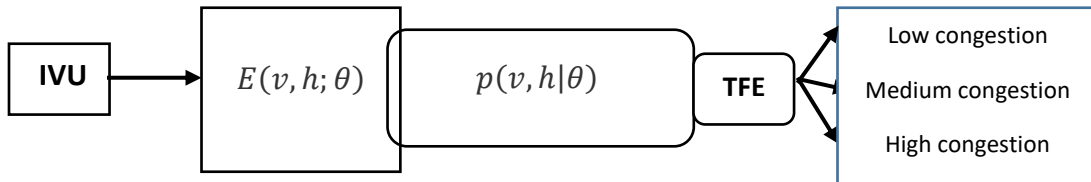


Figure 2.9: DBN Traffic Congestion Pipeline Model

This is based on Hinton and Sejnowski [57], which has been further illustrated by O'Connore *et al.* [67]. The encoded joint distribution function can be defined as:

$$p(v, h|\theta) = \frac{\exp(-E(v, h; \theta))}{\sum_{v'} \sum_{h'} \exp(-E(v', h'; \theta))} \quad (2.6)$$

where the energy function is given by:

$$E(v, h; \theta) = - \sum_i \sum_j w_{ij} v_i h_j - \sum_i b_i^{(v)} v_i - \sum_j b_j^{(h)} v_j \quad (2.7)$$

with the model parameters, $\theta = (w, b^{(v)})$, h_j is the states of the hidden units, w_{ij} represents the states connecting these units: namely the visible input and hidden units; and $b_i^{(v)}$ and $b_j^{(h)}$ are the biases in the visible and hidden units respectively. Hinton *et al.* [66] propose an effective way to learn θ using contrastive divergence. With the TFE, it establishes the following stochastic update rules for the state at which lower energy state eventually attains an equilibrium given by:

$$p(v_i = 1, h|\theta) = \sigma\left(\sum_j w_{ij} h_j + b_i^{(v)}\right) \quad (2.8)$$

$$p(h_j = 1, v|\theta) = \sigma\left(\sum_i w_{ij} v_i + b_j^{(h)}\right) \quad (2.9)$$

Where $\sigma(t) = \frac{1}{1+e^{-t}}$ denotes the sigmoid function with the capability of having its unit state change to 0 and the flexibility of the network to generate samples over all possible states (v, h) based on the joint probability distribution, $p(v, h|\theta)$.

2.4.2.3. Random forest

As illustrated in the RF pipeline model in Figure 2.10, the RF algorithm is used for drawing the number of decision trees (NDT), $n_{tree} = 112$ to grow, such that for each of the bootstrap samples for unpruned analysis it randomly samples the number of variables, $m_{try} = 1.73$, as candidates at each split of the predictors with the best split chosen from among those traffic variables at the IVU. The m_{try} is taken by default to be \sqrt{p} where p is the number of predictor variables found in the IVU. The prediction-type for new data is then performed based on majority voting by aggregating the predictions from the number of decision trees. This is needed to determine the level of congestion.

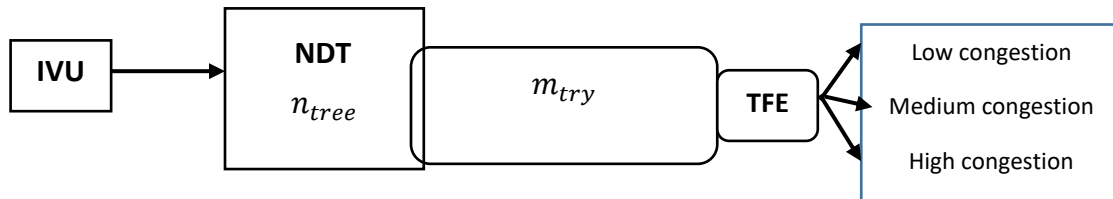


Figure 2.10: Traffic Congestion RF Pipeline Model

The estimation of the error rate as the tree is grown with the bootstrap sample, otherwise known as the ‘out-of-bag’ data involves the tuning of the following hyper parameters: m_{try} and the n_{tree} . While there are other hyper parameters, the m_{try} and the n_{tree} have the most important effect on the final accuracy obtained using RF with bagging i.e. sampling with replacement; otherwise known as the aggregate bootstrapping.

2.5. Application results

Our selected data mining models were implemented using Tensorflow [85] and SparkMLlib [86]. We used Tensorflow to develop our DBN model for image object classification and also implemented our RF, BP-NN and DBN using the SparkMLlib. Tensorflow [87] was originally developed by Google Brain Team and research engineers within Google's Machine Intelligence research organization.

Tensorflow is widely used for conducting machine learning research involving deep neural networks and has increasingly gain acceptance in wide variety of other domains. It is an open source library for numerical computation using data flow graphs in which the graph nodes represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. More information on the use of Tensorflow for training deep neural network for ML classification can be found under the topic: ‘Tensorflow and Wide Deep Learning Tutorial’ [88]. It has application programming interface (API) in Python and C++. In our case, we made use of its Python API to write our codes.

Figure 2.11 illustrate our example of traffic congestion prediction using Tensorflow. The experiment is performed on a window’s terminal. It is on the basis of applying deep neural network on input image data coming from real-time camera recording. This is to tell if the congestion is low, medium or high and to determine the model accuracy based on the ‘score’. The main steps developed with (Python) Tensorflow based on Adetiloye [89] are described as follows:

1. Clone git repository and cd into the directory:

```
git clone https://github.com/taiwotman/TensorflowPredictCongestionTypes.git
cd TensorflowPredictCongestionTypes
```

2. Set up virtualenv with directory *venv*

```
virtualenv venv
```

3. Activate *venv* using:

```
source venv/bin/activate
```

4. Install tensorflow using:

```
pip install tensorflow
```

5. Use traffic congestion image(supports only jpeg/jpg format) e.g

```
python run.py test_image/Aut10_010.jpg
```

6. Example output:

```
high congestion (score = 0.70454)
```

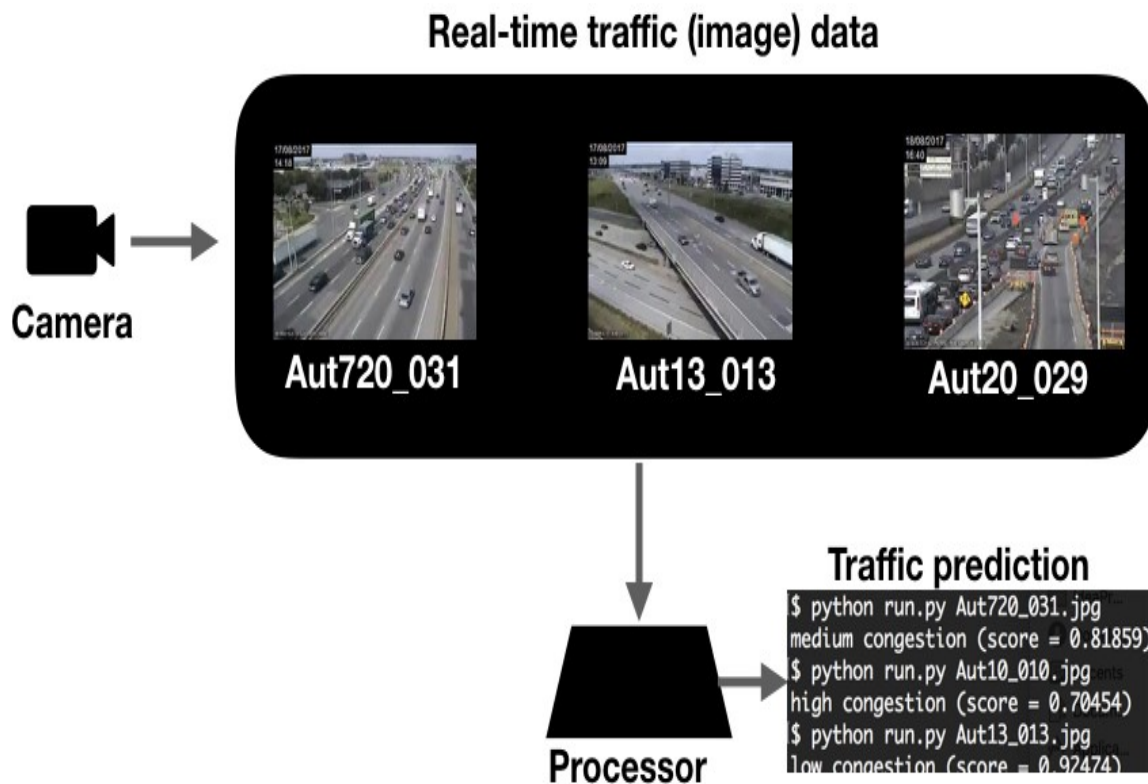


Figure 2.11: Example of predicting traffic congestion using Tensorflow.

(Adapted from Adetiloye [89] <https://github.com/taiwotman/TensorflowPredictCongestionTypes>)

Figure 2.12 gives sample predictions of high way traffic using traffic image data taken from some selected regions. The model predicts traffic congestion as low, medium or high for the three instances.

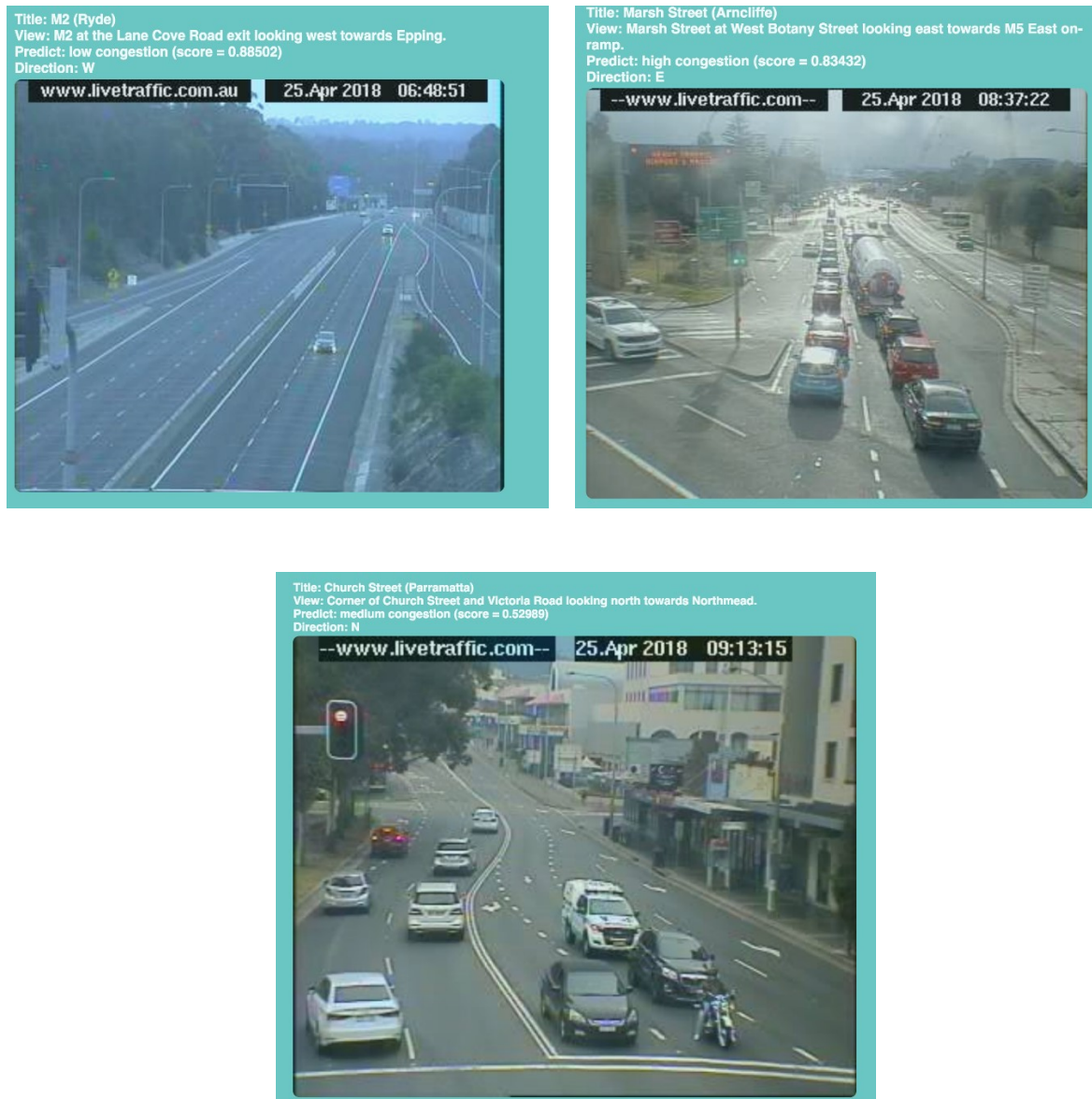


Figure 2.12: Sample predictions of high way traffic using traffic image data taken from some selected regions of New South Wales, Australia.

The SparkMLlib, an Apache Spark's scalable machine learning library (commonly abbreviate as MLlib), has API support for the following programming language: Java, Scala, SQL, Python, and R. It has high performance with fast iterative computation, implements in-memory processing while promoting caches and persistence as well as lazy evaluation in which transformation is delayed until the final action of computing the results. Also, it contains many algorithms and utilities for wide range of ML tasks. Full details of this can be found in [90]. We selected Scala as our programming language of choice to code the ML pipelines of our RF and NN classifiers. Figure 2.13 shows an excerpt of the vehicle count data. Sample of the vehicle count data and source codes can be found in Appendix *A.1, A.2 and A3*.

Day	Length(kn	Time	v(t-15)	v(t)	vhist(t)	vhist(t+15)
1	213	10:05	17	19	13	17
1	213	10:05	12	19	10	19
1	213	10:05	19	8	12	8
1	213	10:05	9	13	13	8
1	213	10:10	14	19	11	17
1	213	10:10	14	18	10	15
1	213	10:10	17	13	17	11
1	213	10:10	18	17	9	16
1	213	10:15	9	13	11	13
1	213	10:15	11	9	19	12
1	213	10:15	12	11	15	11
1	213	10:15	17	13	13	8
1	213	10:20	14	15	23	14
1	213	10:20	12	22	22	11
1	213	10:20	8	9	16	25
1	213	10:20	20	13	18	16
1	213	10:05	17	16	13	14
1	213	10:05	10	11	17	10
1	213	10:05	19	18	14	13
1	213	10:05	17	14	8	17
1	213	10:10	19	14	17	12
1	213	10:10	13	16	12	15
1	213	10:10	13	16	13	19
1	213	10:10	15	13	11	11
1	213	10:15	9	12	8	17
1	213	10:15	16	19	12	19
1	213	10:15	9	8	15	16
1	213	10:15	13	25	16	18
1	213	10:20	11	9	13	24
1	213	10:20	18	23	22	18
1	213	10:20	10	19	24	18
1	213	10:20	17	8	16	20
1	213	10:05	14	17	36	35
1	213	10:05	15	15	19	14

Figure 2.13: Excerpt of vehicle count data

2.5.1. Trend visualization

Figures 2.14 - 2.16 illustrate the trend visualization of the DBN, NN and RF derived from the predictive results of $V(t + 15)$. Virtualization of the trends over time is required when the order of the categories or classes is important. Also, traffic trends may be similar on consecutive weekends except on occasions due to more pronounced variations arising from change in weather conditions (e.g. from winter to summer). For instance, it can be observed that high congestion occurs at 10:05am which marks the peak of the morning rush hour period. It can also be seen that the classification models are able to map the cyclic pattern of the traffic volume in order to make good prediction for previously unseen test data. This is supposing that the performance of the models would depend on number of factors such as the data quality, size and feature selection.

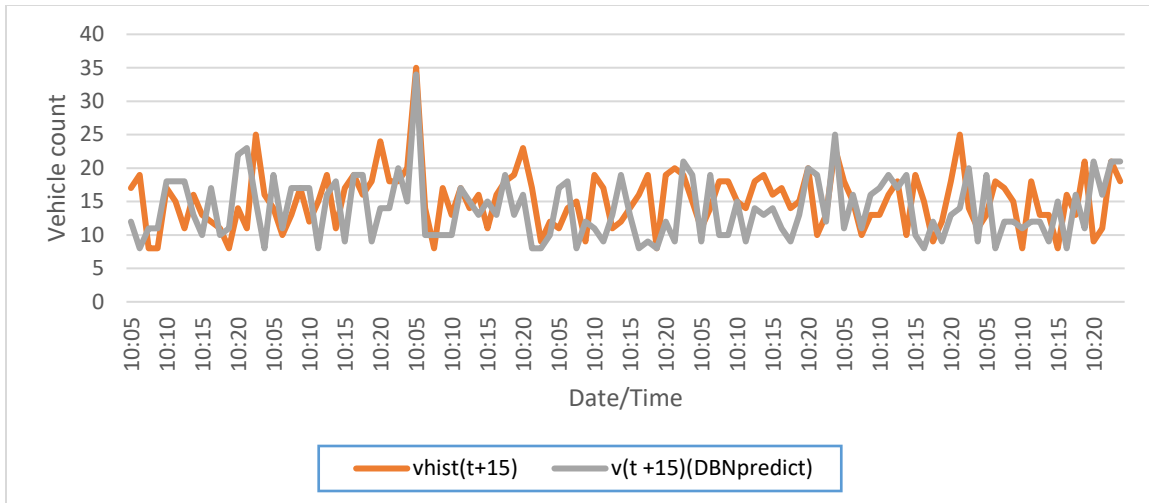


Figure 2.14: DBN traffic congestion model

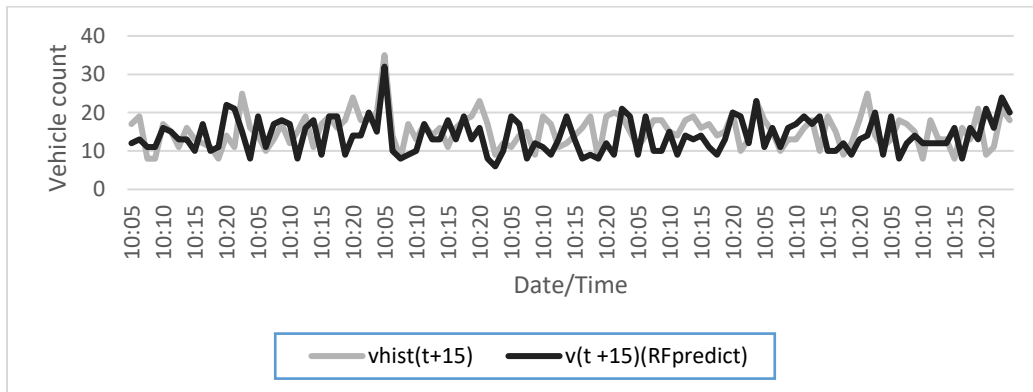


Figure 2.15: RF traffic congestion model

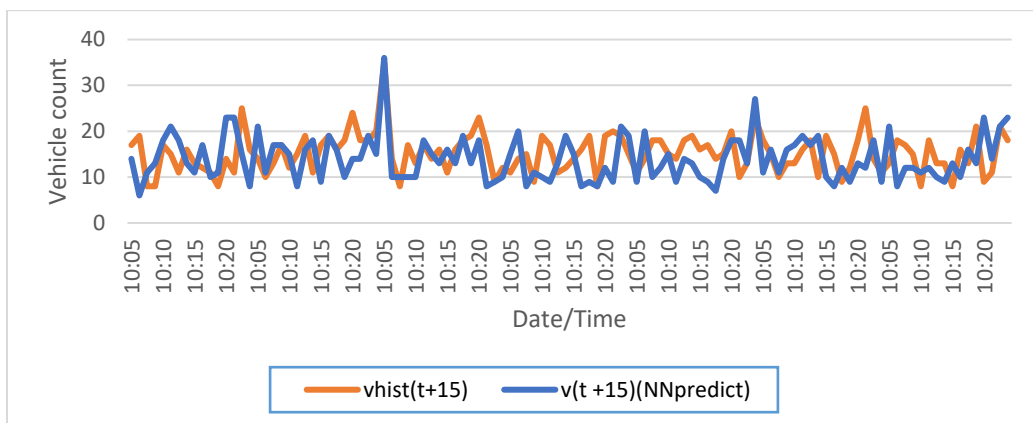


Figure 2.16: NN traffic congestion model

Figure 2.17 illustrate the predictive non-linear graph of these main algorithms with individual date times covering over two weeks.

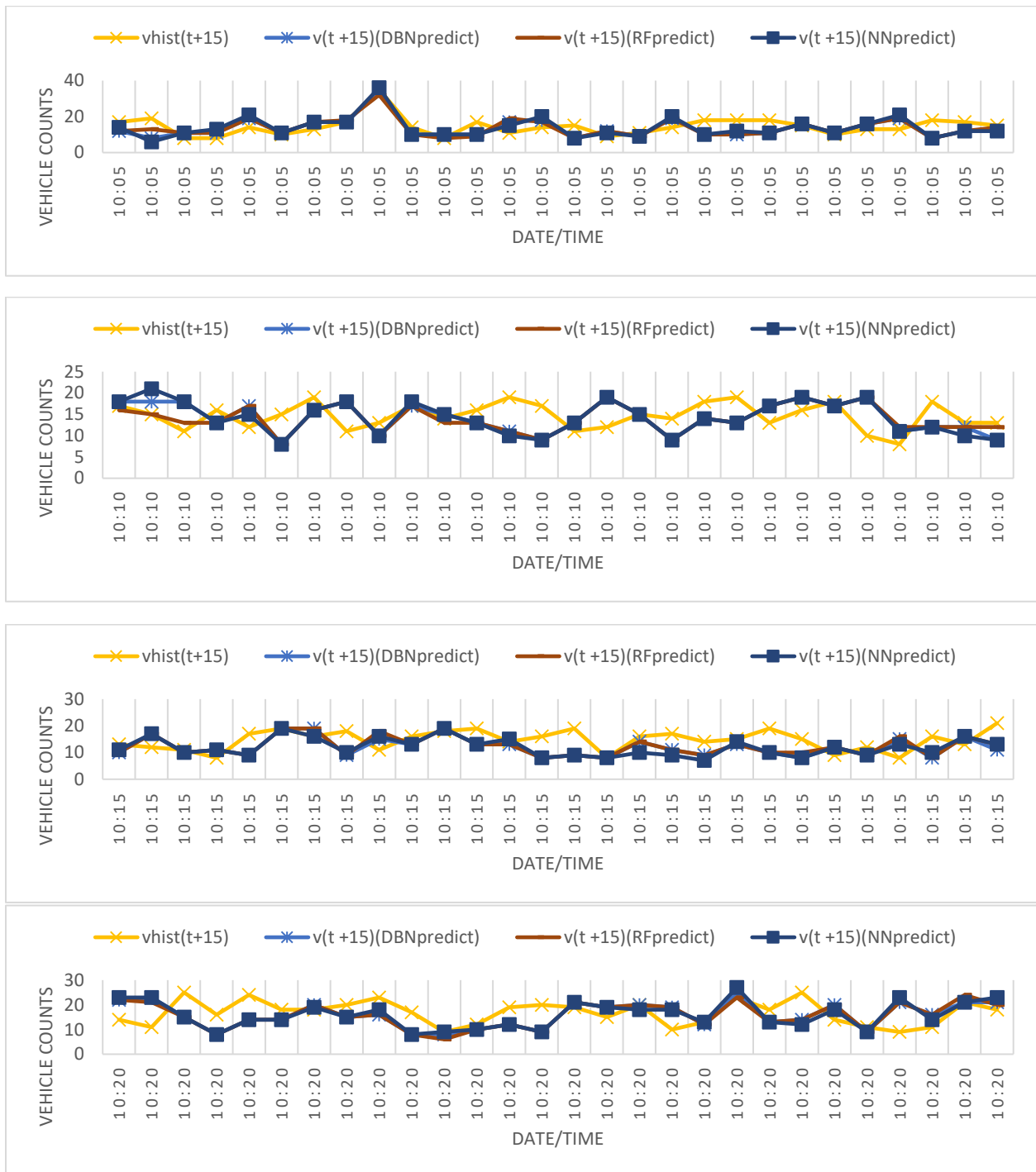


Figure 2.17: Overlaying the predictive models with different date/time.

Figure 2.18 shows the overall predictive non-linear graphs of our selected models. Here, the predictions of DBN and NN are closer in values compared to RF. This might be due to their densely (or fully) connected layers such that every node in one layer is connected to every node in the preceding layer thereby increasing their learning rate and enhancing classification.

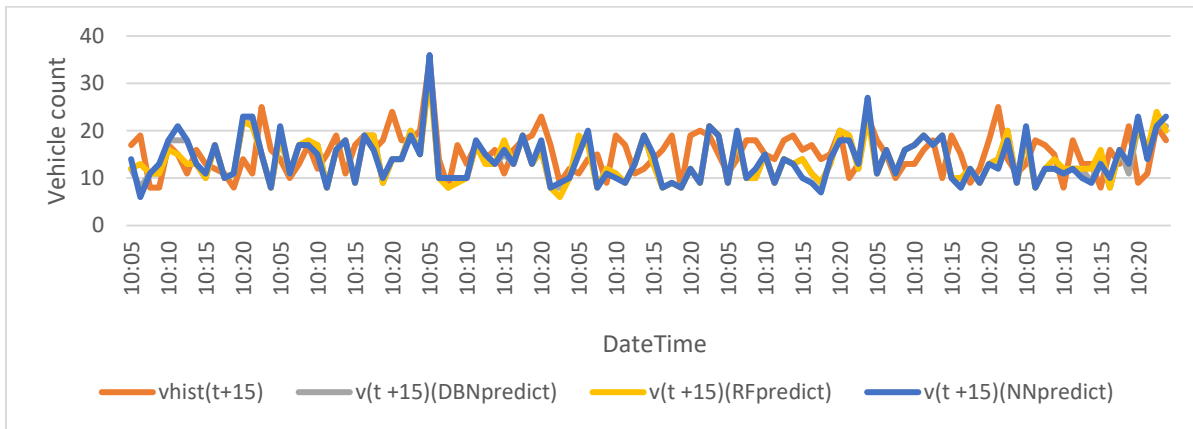


Figure 2.18: Overall predictive models

2.5.2. Model verification

The evaluation metric is Root Mean Square Error, *RMSE*. The *RMSE* represents the sample standard deviation of the differences between predicted (\hat{y}_t) and observed (y) flow data where n is the number of observations. It is calculated using:

$$RMSE = \sqrt{\sum_{t=1}^n (\hat{y}_t - y_t)^2 / n} \quad (2.10)$$

Table 2.2 presents the *RMSE*.

Model		Prediction time interval(date time)			
		10:05	10:10	10:15	10:20
DBN	RMSE	0.5246	0.2430	0.3564	0.4375
NN		0.8411	0.6531	0.8641	0.6465
RF		0.6423	0.7531	0.4017	0.3129

On average, DBN had the lowest *RMSE* compared to the RF and NN while RF was generally better than the NN. We investigate the most effective architecture by varying the number of hidden layers of the DBN and the number of decision trees of the RF as illustrated in Table 2.3.

Table 2.3: Performance measure based on DBN and RF architecture

DBN	Hidden layers	Hidden units	Epoch
(t + 15) prediction	2	100	14
	4	140	24
	6	200	40
	8	250	60
	10	300	70
RF	No of decision trees	mtry	Out of bag error estimation
(t + 15) prediction	10	1.73	200
	100	1.73	250
	200	1.73	270
	400	1.73	350
	600	1.73	350

DBN performance improves disproportionately with the increase in the hidden layer and units. This is on the basis of the number of forward pass and backward pass of all the training instances defined by the epoch. The epoch is a measure of the number of times all of the training vectors are used once to update the weights. For the RF, we observe similar improvement in performance considering that the out of bag error estimation tends to remain steady at 350. Figure 2.19-2.20 illustrates that the best validation performance can be attained at 4-epoch for the NN and DBN (with 10 hidden layers) respectively. This is for a training, validation, and test set split ratio [70:15:15]. The Mean Square Error (MSE) defined as the mean square of the errors is given by:

$$\text{MSE} = \frac{1}{n} \sum_{t=1}^n (\hat{y}_t - y_t)^2 \quad (2.11)$$

\hat{y}_t is the predicted and y_t is the observed flow data where n is number of observations.

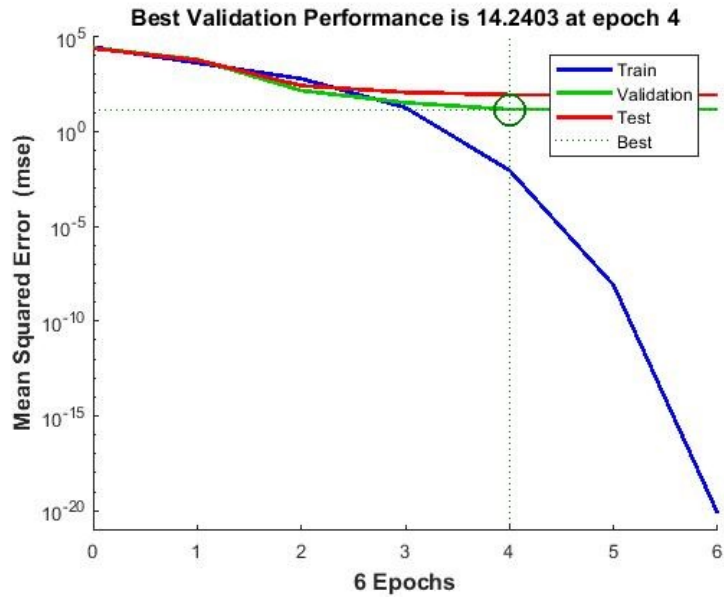


Figure 2.19: Best validation performance: $v(t + 15)$ (NN-predict)

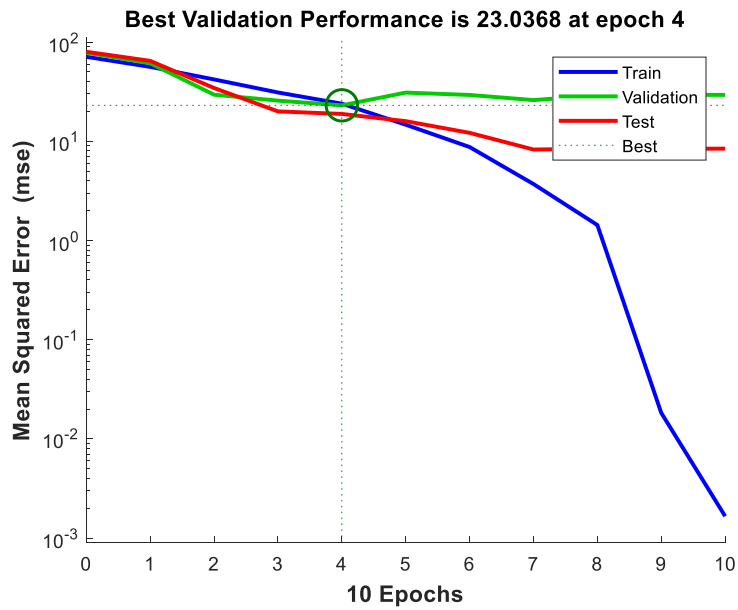


Figure 2.20: Best validation performance: $v(t + 15)$ (DBN-predict)

2.6. Results validation

In order to estimate the baseline performance of our ML models, we compared them with few classified algorithms found in Weka [91] namely ZeroR [92], RandomizableFiltered [93] and RandomTree [94]. As found in the Weka documentation [93-95], they are typically black box algorithms designed to provide a point of reference to predict the majority category or the most observation in the training dataset. For instance, the ZeroR is similar to the NF using sets of rules. The RandomizableFiltered is a metaheuristics that is able to execute classification on data that has been passed through an arbitrary filter. The RandomTree constructs a tree that considers K randomly chosen attributes at each node. This is accomplished without pruning with the option to estimate the class probabilities based on a hold-out set (back fitting). In addition to our earlier defined evaluation metric: $RMSE$, we calculated the correlation coefficient(r) and MAE as well as obtain the execution time.

$$MAE = \frac{1}{n} \sum_{t=1}^n |\hat{y}_t - y_t| \quad (2.12)$$

$$r = \frac{\sum_{t=1}^n (\hat{y}_t - \bar{\hat{y}})(y - \bar{y})}{\{[\sum_{t=1}^n (\hat{y}_t - \bar{\hat{y}})^2 \sum_{t=1}^n (y - \bar{y})^2]\}^{\frac{1}{2}}} \quad (2.13)$$

For the predicted value (\hat{y}_t) and observed value(y), r measures the strength and direction of a linear relationship between the predicted and observed value. The value of r is always between +1 and -1 and the model optimality can be known by how close r is to +1.

The results in Tables 2.4 - 2.5 show that performances of the algorithms could vary based on the execution time. For instance, longer time of execution might result in better performance of the model and sometimes there could be poor model performance depending on several factors such as noise, outliers, data quality and so on. Moreover, as often the case with heuristics, a number of

experimentation vis-à-vis trial and error would be required to find the most feasible solution. It is worthy of note that the results in Table 2.4-2.5 are derived from the training and testing data; hence, cannot be generalized.

Table 2.4: Baseline performance of classifier algorithm on training data				
Classifier Algorithms	RMSE	MAE	Correlation coefficient	Execution time(seconds)
ZeroR	0.2818	0.1126	0.0898	0.06
RF	0.2231	0.0627	0.5478	0.33
NN	0.3009	0.1674	0.0576	0.39
DBN	0.2521	0.1449	0.8453	0.42
RandomizableFiltered	0.1861	0.0529	0.7708	0.61
RandomTree	0.0627	0.2231	0.6880	0.20

Table 2.5: Baseline performance of classifier algorithm on testing data				
Classifier Algorithm	RMSE	MAE	Correlation coefficient	Execution time(seconds)
ZeroR	0.1895	0.1447	0	0.02
Random Forest	0.1409	0.0606	0.8775	0.28
NN	0.2781	0.1471	0.0576	0.39
DBN	0.2818	0.1056	0.9378	0.50
RandomizableFiltered	0.0042	0.0042	0.6898	0.58
RandomTree	0.4231	0.0627	0.5254	0.13

2.7. Conclusions

In this chapter, we investigated the application of two ML models for short term traffic congestion prediction on urban motorway networks. From the trend visualization, we see that random variations may be expected due to cyclic (non-linear) characteristics of the traffic flow congestion pattern. We present a vehicle count traffic classification framework built on our IRTIR system. The system can be used for extracting the cyclic nature of the traffic volume. This is in order to adequately predict traffic congestion in a future time while utilizing the past cases having much similarity to the current one.

The results of our measure of accuracy showed RF to be better than the NN. However, DBN had the best accuracy as measured by the *RMSE* values. This can be attributed to the shallow architecture of the NN. Using a good algorithm like the DBN with a minimum *RMSE* can be at the expense of the model complexity. This is considering the fact that increasing the number of hidden layers with the longer computation time will likely slow the momentum for convergence towards obtaining the best feasible solution. Evidence of this can be gathered from the training, test and validation performance of the DBN based on increase in the epoch

Further study will be needed to identify what solution can reduce the computation time and speed for an excellent algorithm like DBN. This can be measured by evaluating the model performance on the basis of the learning curves.

Chapter 3

Twitter data analysis for traffic congestion prediction

3.1. Introduction

Perhaps the emergence of big data technology could not have been more disruptive anywhere else than in transportation and traffic engineering systems. This is considering that daily traffic flow from human transportation holds vast big data yet to be fully harnessed for real time estimation and prediction. Lu *et al.* [95] observed that such rapid development of urban “informatization”, in the era of big data, offers several details entrenched in some spatio-temporal characteristics, historical correlations and multistate patterns. Undoubtedly, big data have increasingly been used for discovering subtle population patterns and heterogeneities that are not possible with small-scale data (Villars *et al.* [96]). For these reasons amongst others academia, governments, federal and state agencies, industries, and other organizations continue to seek innovations to manage and analyze big data; providing them the prospect of increasing the accuracy of predictions, improving the management and security of transportation infrastructures while enabling informed decision-making to gain better insight into their transportation and traffic engineering phenomena (Vlahogianni *et al.* [97]).

The practical significance of real-time traffic flow state identification and prediction using big data lies in the ability to identify and predict traffic flow state efficiently, timely and precisely (Lu *et al.* [95]). Various articles (Vlahogianni *et al.* [97]; Stopher and Greaves [98]; Wang and Li [99]) have employed big data resources to examine traffic demand estimation, traffic flow prediction and performance as well as integration, and validation with existing

models. A noteworthy aspect is that the rapidly increasing (big data) volume of leading social media microblogging services such as Twitter (twitter.com) can be pragmatically challenging, and nearly impossible to manually analyze (Philander and Zhong [100]). Nevertheless, the huge volume of data derived from Twitter makes it ideal for machine learning,

A few decades ago, researchers developed sentiment and cluster analysis to monitor twitter messages, identify followers and followings, find word resemblances and examine the nature of the comments i.e. positive, negative or neutral. Such promising twitter analytic tools appear to be sufficient in solving the aforementioned traffic flow problems. In this chapter, our objective is tweet mining of the twitter traffic delays and to perform sentiment analysis and cluster classification for traffic congestion prediction based on a model methodology involving tweet crawling, preprocessing steps, feature extraction and social network generation and cluster classification.

3.2. Problem definition

In this chapter, we analyze traffic twitter data with sentiment analysis and cluster classification for traffic flow prediction. Firstly, we examine some key aspects of big data technology for traffic, transportation and information engineering systems. Secondly, we consider Parts of Speech tagging utilizing the simplified Phrase-Search and Forward-Position-Intersect algorithms. Then, we use the k -nearest neighbor classifier to obtain the unigram and bigram; and use Naïve Bayes Algorithm to perform the sentiment analysis. Finally, we use the Jaccard Similarity and the Term Frequency-Inverse Document Frequency for cluster classification of traffic tweets data.

3.3. Literature review

3.3.1. Traffic twitter sentiment analysis

Following the launch of twitter in 2006, sentiment analysis has been applied to various areas of interests e.g. extracting adverse drug reactions from tweets (Korkontzelos *et al.* [101]), news coverage of the nuclear power issues (Burscher *et al.* [102]), and in the tourism sector for capturing sentiment from integrated resort tweets (Philander and Zhong [100]). Terabytes of twitter data could be from traffic road users expressing their opinions on traffic jam, road accidents and other information which constitute general traffic news update. The question, of course, is how to determine traffic flow state based on the weight as measured by the opinion contained in a twitter message (called “tweet”) –a short message that a sender post on twitter that cannot be longer than maximum 140 characters? According to Abidin *et al.* [103], certain special characters including @, RT, and # symbols used in a tweet creates a collective snapshot of what people are saying about a given topic. An in-depth process of computationally identifying and automatically extracting opinions from a writer’s piece of text to determine whether the attitude or emotions towards a topic is positive, negative or neutral is known as sentiment analysis (Pak and Paroubek [104]; Go *et al.* [105]). The technique of sentiment analysis is generally expected to yield a high accuracy rate of roughly 70% to 80% in training-test data matching tasks (Wang *et al.* [106]), while objectively seeking useful insights from a large quantity of aggregated data instead of achieving perfect classification of all data points (Philander and Zhong [100]). Sentiment mining using corpus based and dictionary based methods for semantic orientation of the opinion words in tweets has been presented by Kumar and Sebastian [107].

In drawing the relevance of twitter sentiment analysis to traffic flow state prediction, He *et al.* [108] consider improving long-term traffic prediction with tweet semantics; and, then,

analyze the correlation between traffic volume and tweet counts with various granularities. Finally, an optimization framework to extract traffic indicators based on tweet semantics using a transformation matrix, while integrating them into the traffic prediction using linear regression is proposed. Real-time traffic improvement by semantic mining of social networks has been captured by Grosenick [109]. Abidin *et al.* [103] introduce the use of Twitter API to retrieve traffic data serving as input to Kalman Filter [110] models for route calculations and updates while fine-tuning the output for new, accurate arrival estimation.

3.3.2. Traffic twitter cluster classification

Tweets have a hashtag which consist of any word that starts with “#” symbol. Hashtags help to search messages containing a particular tag. Also of interest is the Part of Speech (POS) tagging in tweets, which has been applied by Elsafoury [111] to monitor urban traffic status. The main idea of POS tagging, also known as word-category disambiguation, is to mark up a word in a corpus and to assign it to a corresponding POS based on its definition and its context. The former is an example of exact term search while the latter, POS, can be considered a typical example of full-text search, which is usually thorough in its search process but can be more challenging to perform when compared to the exact text search. One instance of such text search is classification of tweets into positive and negative sentiments using multinomial Naïve Bayes’ unigram with mutual information based on n -grams and POS that has been presented by Go *et al.* [105]. It outperforms other classifier approaches under consideration. In between the exact and full-text search is the phrase text search for searching a particular word phrase. For instance, an exact term search might be required to search the term “delay” in a tweet stream. This would bring out only tweets containing the term “delay”. On the other hand, a phrase term search could

be a phrase like “Traffic delay” in which there are more details of the search term. Phrase text search is often more useful when performing cluster classification than the other text search methods. It is noteworthy that using a particular search operation is based on measuring the relevance of the query to efficiently match the terms appropriately. Azam *et al.* [112] present the functional clustering details of their tweets mining approach which has the following steps:

- 1) *Tweet crawling*: It is the process of retrieving tweets from twitter server using Twitter Application Program Interface (API). The crawled tweets are stored on local machine for further processing.
- 2) *Tweets pre-processing and tokenization*: It involves the filtering of the crawled tweets of non-entirely textual items like emoticons, URL, special character, stop words etc. A common tokenization method known as the n -gram technique can then be applied to tokenize the tweets into bag-of-works ($n = 1$, known as a unigram is recommended for such tweets tokenization by Broder *et al.* [113]).
- 3) *Feature extraction and social network generation*: It is the process of extracting important features from the preprocessed and tokenized tweets while transforming the feature sets into a social network generation comprising a term tweet matrix A of order $m \times n$, where m is the number of candidate terms and n is the number of tweets. The resulting matrix A is used to compute the weight $w(t_{i,j})$ using the following two equations:

$$w(t_{i,j}) = tf(t_{i,j}) \times idf(t_i) \quad (3.1)$$

$$idf(t_i) = \log \frac{|D|}{|d_j: t_i \in d_j|} + 1 \quad (3.2)$$

Where $tf(t_{i,j})$ is the number of times t_i occurs in j^{th} tweet.

$|D|$ is the total number of tweets and $\{d_j: t_i \in d_j\}$ represents the number of tweets with term, t_i . The objective is to normalize matrix A such that the tweet vectors' length equals to 1.

- 4) *Social network clustering*: After generating the social network for the complete set of tweets, Markov clustering is used to achieve the social network clustering by crystallizing the network into various cluster each representing individual events. The Markov clustering algorithm (introduced by van Dongen [114]) is a fast and scalable unsupervised cluster algorithm for graphs (also known as *networks*). It serves as an iterative method for interleaving of the matrix expansion and inflation steps based on simulation of (stochastic) flow in graphs.

More details on the above steps can be found in Azam *et al.* [112]. For traffic flow prediction using big data analysis and visualization, McHugh [115] considered among other approaches the use of traffic tweets to test the effectiveness of geographical location of vehicles to determine the location of an incident. Tejaswin *et al.* [116] introduce a continuous traffic management dashboard automated system to generate real-time city traffic insights and predictions using social media data. Figure 3.1 illustrates the twitter traffic analytic system.

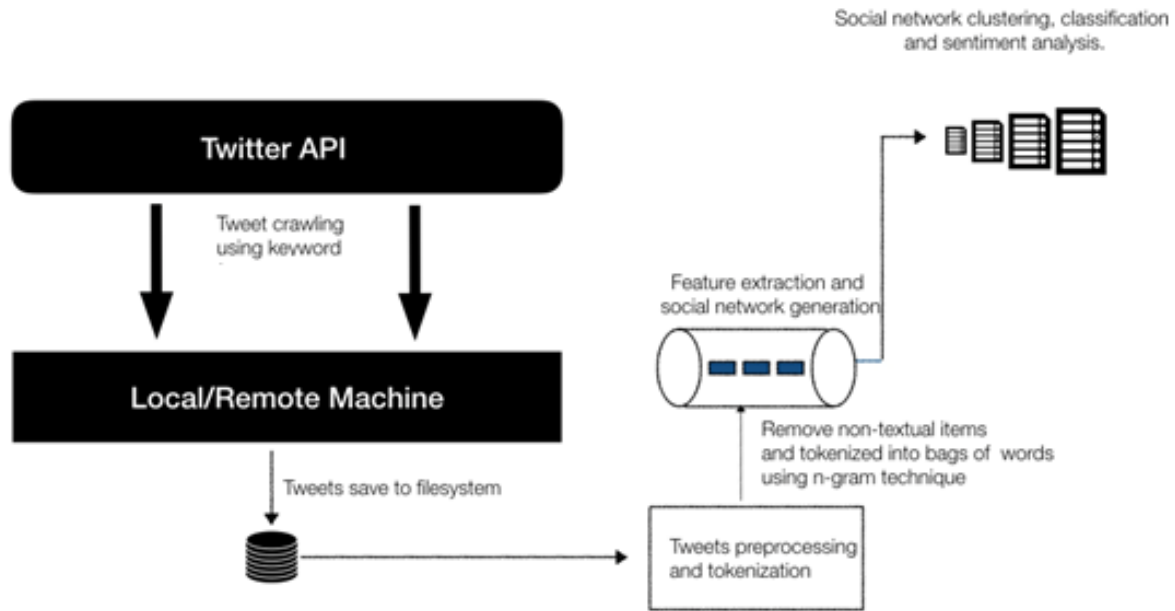


Figure 3.1: Twitter traffic analytic system

3.4. Solution approach

Figure 3.2 presents the schematic diagram of our traffic twitter data analysis. To obtain logs of twitter traffic data for the sentiment analysis and cluster classification, we use twitterR package to connect to the Twitter API and perform OAuth authentication using the ROAuth package all in RStudio. The plyr and stringr packages are used to crawl a number of tweets into RStudio while ensuring they are clean of unwanted symbols. More detail of this twitter text mining can be found in Rais [117] and detailed documentation of the widely used twitter data mining statistical program can be found in cran.r-project.org [118]. We perform a phrase search based on the phrase using a POS tag: *Montreal traffic*. This is made possible with a simplified phrase search algorithm derived from Eckert [119](see *Algorithm 1*), with the original simplified version by Manning *et al.* [120].

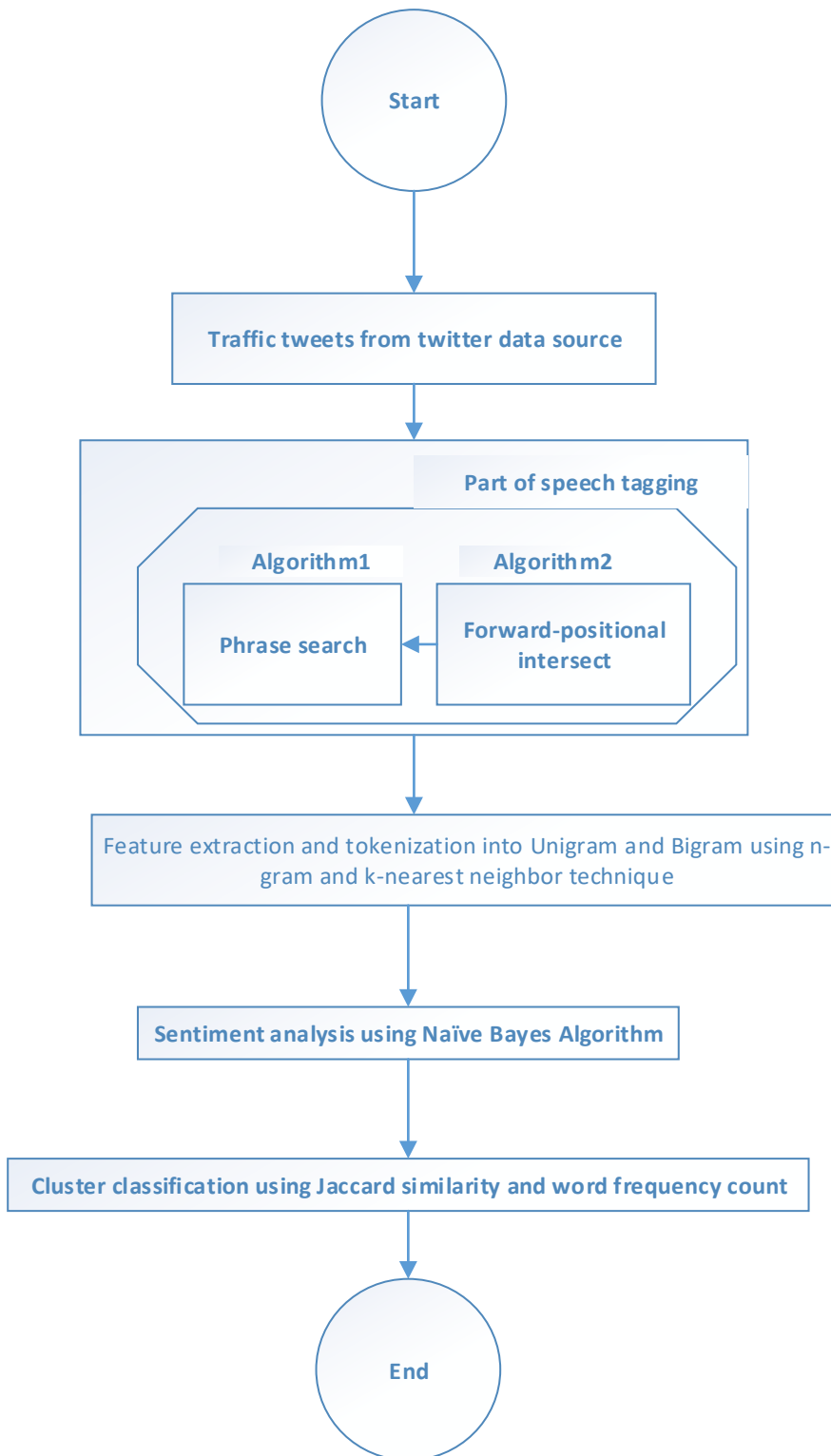


Figure 3.2: Schematic diagram of our traffic twitter data analysis.

Algorithm 1

Phrase-Search(index , phrase)

1. $t \leftarrow \text{Terms}(\text{phrase})$
2. $k \leftarrow 1$
3. $\text{answer} \leftarrow \text{Index-Get}(\text{index}, t)$
4. $t \leftarrow \text{next}(t)$
5. while $t \neq \text{NIL}$ and $\text{answer} \neq \{\}$
6. do $\text{nextTweet} \leftarrow \text{Index-Get}(\text{index}, t)$
7. $\text{answer} \leftarrow \text{Forward-Positional-Intersect}(\text{answer}, \text{nextTweet}, k)$
8. $k \leftarrow k + 1$
9. $t \leftarrow \text{next}(t)$
10. return answer

In order to apply the above algorithm to tackle our problem, a positional *index* containing a list of data mined tweets with a list of positions is used to indicate the search phrase. The *Terms* is taking to be a split-normalization tokenizer that splits the *phrase* into list of tokens, normalizing them and assigning as a bag of words. We consider the weighted k -nearest neighbor classifier (Samworth [121]) which assigns a weight $1/k$ and other as k weights. This is by finding the vector of nonnegative weights that is asymptotically optimal while minimizing the misclassification error rate, R_R (Samworth [121]). Essentially, the asymptotic expansion is needed to ensure strong consistency in the search. This is subject to a regularity class distribution:

$$R_R(C_n^{wnn}) - R_R(C^{Bayes}) = (B_1 s_n^2 + B_2 t_n^2) \{1 + o(1)\}, \quad (3.3)$$

Let us take C_n^{wnn} to be the weighted nearest classifier with weights $\{w_{ni}\}_{i=1}^n$ where B_1 and B_2 are some complicated constants to be determined as defined by:

$$B_1 = \int_S \frac{\bar{f}(x_o)}{4\|\bar{\eta}(x_o)\|} dVol^{d-1}(x_o) \quad (3.4)$$

$$B_2 = \int_S \frac{\bar{f}(x_o)}{\|\bar{\eta}(x_o)\|} dVol^{d-1}(x_o),$$

Vol^{d-1} denotes the natural $(d - 1)$ dimensional volume with measure inherent in $S \in \mathbb{R}^d$ while $\bar{f}(x_o)$ denotes the first derivative of the initial point x_o ; $s_n^2 = \sum_{i=1}^n w_{ni}^2$ and $t_n = n^{-2/d} \sum_{i=1}^n w_{ni} \{i^{1+\frac{2}{d}} - (1-i)^{1+\frac{2}{d}}\}$ represent variance and squared bias contributions. C^{Bayes} denotes the Bayes classifiers, minimizing the risk over R . Both are given by:

$$C_n^{wnn}(x) = \begin{cases} 1, & \text{if } w_{ni}_{i=1}^n \geq 1/2 \\ 2, & \text{otherwise} \end{cases} \quad (3.5)$$

$$C^{Bayes}(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 1/2 \\ 2, & \text{otherwise} \end{cases}$$

Therefore, there is the interpretation that for the point $x \in \mathbb{R}^d$, $\eta(x)$ belongs to class $C(x)$ with value of 1 in the sense of the weighted nearest neighbor classifier if $w_{ni}_{i=1}^n \geq \frac{1}{2}$; and in the sense of the bayesian classifier, if the regression function $\eta(x) = P(Y = 1|X = x) \geq \frac{1}{2}$ and; otherwise, both have a value of 2. Further interpretation of the asymptotic behavior towards optimal classification can be found in Samworth [121]. Subsequently, provided that a single term t from the index is not empty based on the resulting *answer* from the positional *index*, we can iterate over the number of incoming tweets while adapting the document list Forward-Position-Intersect algorithm (Eckert [119]; Manning *et al.* [120]) as follows:

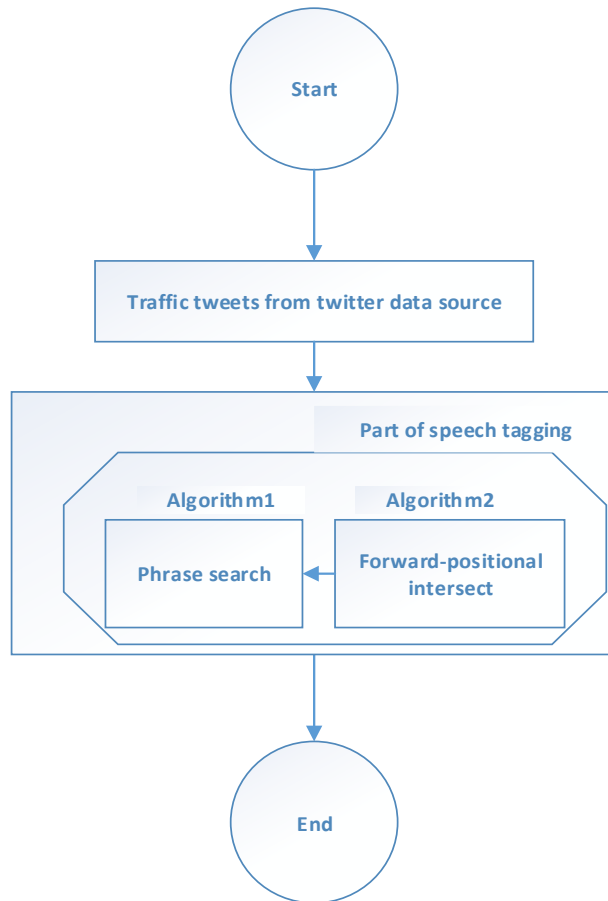
Algorithm 2

Forward-Positional-Intersect(p_1, p_2, k)

```
1. answer  $\leftarrow \{\}$ 
2. while  $p_1 \neq \text{NIL}$  and  $p_2 \neq \text{NIL}$ 
3. do if  $\text{tweetId}(p_1) = \text{tweetId}(p_2)$ 
4.   then:
5.      $pp_1 \leftarrow \text{positions}(p_1)$ 
6.      $pp_2 \leftarrow \text{positions}(p_2)$ 
7.     while  $pp_1 \neq \text{NIL}$  and  $pp_2 \neq \text{NIL}$ 
8.     do if  $\text{pos}(pp_2) - \text{pos}(pp_1) = k$ 
9.       then Add(answer,  $\text{tweetId}(p_1)$ ,  $\text{pos}(pp_1)$ )
10.       $pp_1 \leftarrow \text{next}(pp_1)$ 
11.       $pp_2 \leftarrow \text{next}(pp_2)$ 
12.      elseif  $\text{pos}(pp_2) - \text{pos}(pp_1) > k$ 
13.      then:  $pp_1 \leftarrow \text{next}(pp_1)$ 
14.      else:  $pp_2 \leftarrow \text{next}(pp_2)$ 
15.      $p_1 \leftarrow \text{next}(p_1)$ 
16.      $p_2 \leftarrow \text{next}(p_2)$ 
17.   elseif  $\text{tweetId}(p_1) > \text{tweetId}(p_2)$ 
18.     then  $p_1 \leftarrow \text{next}(p_1)$ 
19.   else  $p_2 \leftarrow \text{next}(p_2)$ 
20. return answer
```

Re-defining the variables in Eckert [119], let p_1, p_2, pp_1 and pp_2 be the pointers to tweet lists and let p_1 and p_2 reference the tweet lists of the two terms to be intersected while pp_1 and pp_2 reference the inner position lists for each tweet with tweetId and pos dereferencing the pointers to their actual value in the list. Let positions extract the inner position list from an entry

in the tweet list. *Add* adds a list identifier and a position to the resulting tweet list. The tweet list represents the tweets logs of traffic information saved into file. Figures 3.3 illustrates the functional relationship with these two algorithms.



Figures 3.3. Functional relationship between the Phrase search and Forward-Positional Intersect

For our sentiment analysis, we consider the approach of Hu and Liu [122]) lexicon of opinion words (LOWs). With our earlier derivations, we posit that the index of sentiments word would require correct interpretation of the word context in relevance to the topic of traffic delay and congestion by scoring the opinion contained in the traffic tweets based on the contextual polarity: positive, negative and neutral. The first method of the improved Naïve Bayes

Algorithm (INB-1) by Kang *et al.* [123] was helpful in computing the score for the crawled filtered traffic tweets based on the following conditional probability:

$$Class(t_i) = \arg \max R_1(p_{ij})P(c_j) \prod_{i=1}^d P(p_i|c_j) \quad (3.6)$$

$$R_1(p_{ij}) = \frac{\sum_{p_{ij} \in L_j} C(p_{ij})}{\sum_{p_{ij} \in L} C(p_{ij})} \quad (3.7)$$

Where $Class(t_i)$ denotes the function that determines whether a traffic tweet (t_i) is positive, negative or neutral. The probability of class c_j is calculated by $P(c_j)$ while $P(p_i|c_j)$ computes the probability that p_i belongs to c_j . $R_1(p_{ij})$ denotes the ratio of number of patterns $C(p_{ij})$ present in the class j of LOWs when the number of patterns $|L|$ is counted over number of patterns $C(p_{ij})$ present in the class j of LOWs when the number of patterns $|L|$ is uncounted. The pattern essentially an n -gram, dwells on the form of $n - 1$ Markov model, representing contiguous sequence of n items from a corpus widely known as shingles. We used the Jaccard index to know the extent of similarity between sample sets of shingles irrespective of the ordering. This is given by:

$$J(C_1, C_2) = \frac{|C_1 \cap C_2|}{|C_1 \cup C_2|} \quad (3.8)$$

$J(C_1, C_2)$ denotes the similarity between set C_1 and C_2 . It follows that when item C_1 and C_2 are unrelated then $J(C_1, C_2) = 0$; otherwise $0 \leq J(C_1, C_2) \leq 1$. The cluster formation provides enough evidence to support the interrelations between traffic incidents with regards to the trending causatives of traffic congestions. Furthermore, we employ the term-frequency-inverse-document-frequency, *tfidf* (Spärck Jones [124]) to classify each term in the traffic congestion

clusters based on the frequency of occurrence. This is performed by invoking the TF log-normalization with the smooth *tdidf* weight-schemes as follows:

$$tf(t, d) = 1 + \log(f_{t,d}) \quad (3.9)$$

$$idf(t, D) = \log \frac{N}{n_t} \quad (3.10)$$

Such that tweet document term weight is given by:

$$tdidf(t, d, D) = tf(t, d).idf(t, D) \quad (3.11)$$

With $N = |D|$ denoting the total number of document in the corpus; $n_t = 1 + |\{d \in D: t \in d\}|$ representing number of times term t appears in document d which belongs to D in the corpus. Notice that the addition of 1 to $|\{d \in D: t \in d\}|$ ensure that infinity value $idf(t, D)$ is avoided.

3.5. Numerical application

3.5.1. Discussion of results

We perform our analysis in Rstudio [125] – an integrated development environment (IDE) for coding with R programming language. R is widely used among statisticians and data miners for developing statistical computing, graphics and data analysis in its free software environment. The versions update is supported by the *R Foundation for Statistical Computing* [126]. Hence, using R, a sample of over 1000 tweets were retrieved based on the phrase search *Montreal traffic*. Figure 3.3 shows an excerpt of the tweets on *Montreal traffic*. The data was cleaned of irrelevant symbols. Beyond the steps of tweets crawling, preprocessing, tokenization and feature extraction explained in our methods, we obtained the sentiment analysis as presented in Table 3.1. Also, for text analysis and visualization we used Lexalytics [127]. As illustrated in Figure 3.4, 16.9% are negative, 82.8% are neutral, 0.5% are positive using a grand total of 1049

Montreal traffic tweets studied. The Jaccard similarity and *tdidf* is used to generate the relevant traffic trending events contributing to the cluster classification index shown in Figure 3.5.

Montréalnorthbound AutDecarie ExpyAut Decarie between Queen Mary and AutRteAut slow traffic
Montréalwestbound Aut after Boul Gouin stalled vehicle in the right hand laneCLEAR
Montréalsouthbound Aut off ramp Souigny vehicle with flat tireCLEAR
Montréalsouthbound Aut between Rue Sherbrooke and Souigny slow traffic
Montréalsouthbound Aut off ramp Souigny vehicle with flat tire
Montréalwestbound Aut between Boul Brien and Boul Gouin slow traffic
Montréalleastbound TranscanadienneTCan between Boul CavendishSortieand Aut slow traffic
Montréalsouthbound AutAutoroute Chomedey between Boul SteRoseSortieand Pont LouisBisson slow traffic
Montréalwestbound Aut after Boul Gouin stalled vehicle in the right hand lane
Montréalsouthbound AutAutoroute Chomedey at AutTranscanadienneTCansortiestalled vehicle in the right hand laneCLEAR
Montréalsouthbound AutAutoroute Chomedey at AutTranscanadienneTCansortiestalled vehicle in the right hand lane
Montréalnorthbound AutTurcot off ramp AutAutEch Turcot stalled vehicleCLEAR
Montréalleastbound Aut off ramp Rue Guy stalled vehicle in the left hand laneCLEAR
Montréalleastbound Aut off ramp Rue Guy stalled vehicle in the left hand lane
Montréalnorthbound Pont Mercier between RteRte and Rue Airlie slow traffic
Montréalsouthbound AutAutoroute Chomedey between Boul Cleroux and Pont LouisBisson slow traffic
Montréalnorthbound AutTurcot off ramp AutAutEch Turcot stalled vehicle
Montréalleastbound Aut at Boul Gouin stalled vehicle in the right hand laneCLEAR
Montréalleastbound Aut at Boul Gouin stalled vehicle in the right hand lane
Montréalwestbound Aut off ramp Boul Angrignon stalled vehicleCLEAR
Montréalwestbound Aut off ramp Boul Angrignon stalled vehicle
Montréalleastbound Aut off ramp Boul Des Sources stalled truckCLEAR
Montréalleastbound Aut off ramp Boul Des Sources stalled truck
Montréalleastbound TranscanadienneTCan at Boul StJeanSortiestalled truckCLEAR
Montréalleastbound MétropolitaineMetropolitan at Deslauriers stalled vehicle in the right hand laneCLEAR
Montréalsouthbound AutAutoroute Chomedey at Boul De La Cotevertu stalled vehicle in the right hand laneCLEAR
Montréalsouthbound AutAutoroute Chomedey at Boul De La Cotevertu stalled vehicle in the right hand lane
Montréalleastbound TranscanadienneTCan at Boul StJeanSortiestalled truck
Montréalsouthbound Aut between Hochelega and Lafontaine Tunnel slow trafficCLEAR
Montréalleastbound Aut on ramp Boul BourgetSortiedebris RLCLEAR
Montréalleastbound MétropolitaineMetropolitan on ramp Blvd St Laurent stalled vehicle in the right hand laneCLEAR
Montréalleastbound Aut on ramp Rue Morgan stalled vehicleCLEAR
Montréalleastbound Aut on ramp Rue Morgan stalled vehicle
Montréalleastbound MétropolitaineMetropolitan on ramp Blvd St Laurent stalled vehicle in the right hand lane
Montréalwestbound Aut between Boul StLaurentRue Berri and Tunnel Ville Marie slow trafficCLEAR
Montréalnorthbound AutDecarie ExpyAut Decarie between Plamondon and AutRteAut slow trafficCLEAR
Montréalleastbound Aut at Boul BourgetSortiestalled vehicle in the right hand laneCLEAR
Montréalsouthbound AutAutoroute Chomedey between Boul StMartin WRteSortieand Pont LouisBisson traffic flowing freely
Montréalleastbound Aut at Boul BourgetSortiestalled vehicle in the right hand lane

Figure 3.3: Excerpt of the Montreal traffic tweets

Table 3.1: Traffic twitter sentiment analysis

Phrase	Negative	Neutral	Positive	Total
Slow traffic	162			162
Stalled traffic	4	128		132
ExpyAut Decarie		119		119
slow traffic	2	102		104
stalled vehicle	2	92		94
right hand lane		83		83
right hand Clear		70	6	70
Montréaleastbound Transcanadienne Can		72		72
MontréalSouthbound AutAutoroute Chomedey		74		74
Montréalnorthbound AutDecarie		71		71
MontréalSouthbound AutDecarie		62		62
Total	170	869	6	1049

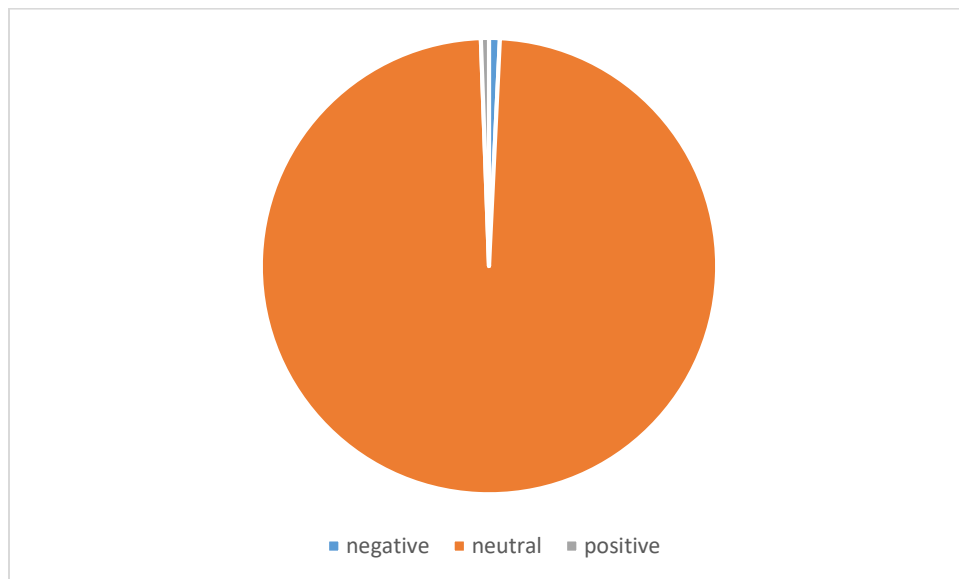


Figure 3.4: Pie charts showing proportion of the total sentiment

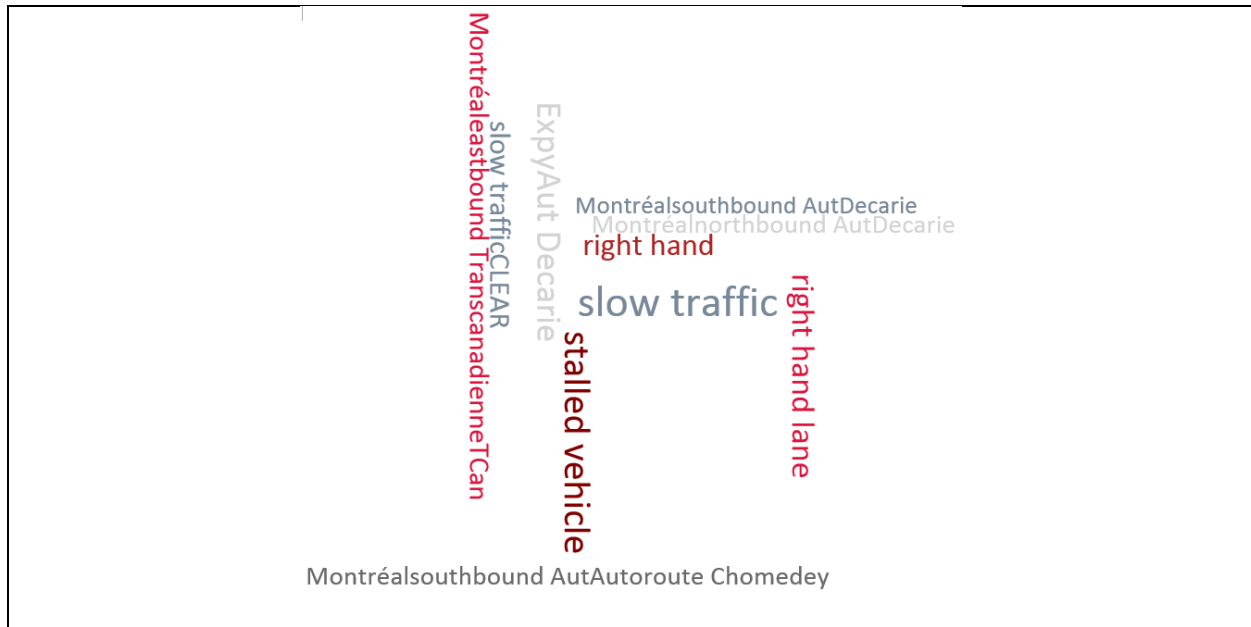


Figure 3.5: Traffic delay trending events

3.6. Conclusions

Sentiment and cluster classification of twitter big data continue to influence different areas of information technology – one of which is traffic information and transportation engineering managements. Applying the proposed data mining techniques on different strata of the traffic delay tweets yields interesting results that could help inform better decision-making on traffic congestion, incidents and control.

Chapter 4

Data fusion for traffic congestion prediction

4.1. Introduction

In this chapter, we propose a big data fusion framework based on homogenous and heterogeneous data for traffic congestion prediction. The homogeneous data fusion model fuses data of same types (quantitative) estimated using machine learning algorithms: back propagation neural network, random forest, and deep belief network. In heterogeneous traffic data fusion, the quantitative and qualitative data obtained from twitter sentiment analysis are interpreted using the Mamdani Fuzzy rule inferencing system [47]. The proposed approaches are demonstrated through application on Genetec Blufaxcloud travel-time system engine (GBTTSE) [128]. Figure 4.1 shows the map of the Montreal motorway network under consideration with each node on the map being either a start or end node. Figure 4.2 presents the (near) real-time traffic data for Montreal motorway network.

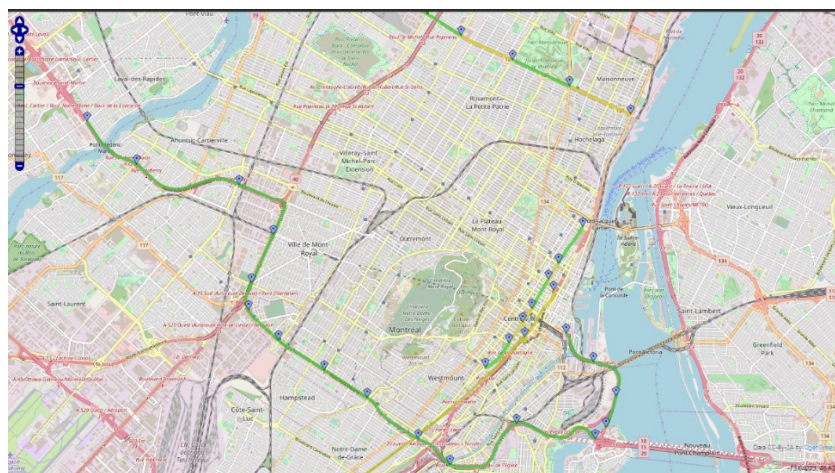


Figure 4.1: Map of the Montreal motorway network (source: GBTTSE [128])

Network	Description	Speed		Travel Time (mins)		
		(mph)	(km/h)	Current	Predicted	Historical
Montreal	MTL A15N 1 MTL A15N Champlain-Atwater north [1.47 mi / 2.37 km] View Details	30.4	48.9	2.9	2.9	
Montreal	MTL A15N 2 MTL A15N Atwater-SIPatrick WARNING: *** Geometry Issue Detected *** north [1.49 mi / 2.4 km] View Details	38.3	61.7	2.33	2.33	
Montreal	MTL A15N 3 MTL A15N SIPatrick-UpperLachine north [0.67 mi / 1.08 km] View Details	38.3	61.6	1.05	1.95	
Montreal	MTL A15N 4 MTL A15N UpperLachine-SILuc WARNING: *** Geometry Issue Detected *** north [1.08 mi / 1.74 km] View Details	36.0	57.9	1.8	1.95	
Montreal	MTL A15N 5 MTL A15N SILuc-CoteStCatherine north [0.83 mi / 1.34 km] View Details	32.5	52.3	1.53	1.95	
Montreal	MTL A15N 6 MTL A15N CoteStCatherine-JeanTal north [0.89 mi / 1.43 km] View Details	15.6	25.0	3.43	3.43	
Montreal	MTL A15N 7 MTL A15N JeanTalon-Duncan north [0.69 mi / 1.11 km] View Details	8.7	14.1	4.73	4.73	
Montreal	MTL A15N 8 MTL A15N Duncan-Lucerne north [0.47 mi / 0.76 km] View Details	15.5	25.0	1.82	1.95	
Montreal	MTL A15N 9 MTL A15N Lucerne-Dunkirk (A06) north [0.85 mi / 1.37 km] View Details	14.2	22.8	3.6	3.6	
Montreal	MTL A15N 10 MTL A15N Dunkirk-Sauve north [1.35 mi / 2.17 km] View Details	25.9	41.6	3.13	3.13	
Montreal	MTL A15N 11 MTL A15N Sauve-Salaberry north [1.85 mi / 2.98 km] View Details	35.1	56.4	3.17	3.17	
Montreal	MTL A15N 12 MTL A15N Salaberry-Cartier	50.6	81.4	1.32	1.95	

Figure 4.2: (Near) real-time traffic data for Montreal motorway network (source: GBTTSE [128])

4.2. Problem definition

Data fusion can be defined as the process of merging data from homogeneous or heterogeneous data sources like GPS, probe vehicle, social media, simulation etc. so as to give multi-dimensional information and knowledge in a more accurate, and reliable representation than that of raw input data from a single data source. Homogenous data sources provide data with same characteristics, which can be either structured (data table), semi-structured (XML, JSON) or unstructured form (social media data). On the other hand, heterogeneous data sources provide data with varying characteristics, which can be two or all of the structured, semi-structured and unstructured form. The problem addressed in this chapter is how to perform multi-modal (homogenous and heterogeneous) data fusion.

4.3. Literature review

Over the past decade, traffic data fusion has increasingly been adopted due to its many advantages which include reduced ambiguity, increased robustness, increased confidence factor, enhanced spatial and temporal coverage and decreased cost (Anand *et al.* [129]; Dailey *et al.* [130]; Bachmann [131]). It is said to be one of the best approaches for accurate estimation and prediction of traffic parameters using data from many sources (Anand *et al.* [129]). For instance, a GPS integrated navigation system which utilize multi-data fusion was developed based on decentralized data fusion. This is to eliminate the obvious errors of the GPS traffic data during the estimation of traffic density (Bin *et al.* [132]). A Kalman filter (KF) model using simulated loop detector and probe vehicle data has been used to estimate travel time (Chu *et al.* [133]). Hence, Kalman filtering and other data fusion techniques utilizing Bayesian inference, Dempster-Shafer evidential reasoning, artificial neural networks, and fuzzy logic rule-based membership continue to gain wide acceptance in the area of ITS (El Faouzi *et al.* [134]).

Wang *et al.* [135] raise some concerns over serious traffic congestion causing great economic loss and environmental problems. This brings the urgent need for best travel path that would be independent of a systemic data source failure due to lack of backup and alternative data plan. In fact, high congestion may persist for longer hours as a consequence of drivers missing the travelling path because of faulty traffic information equipment like sensors which sometimes malfunction. They further argue that less reliance on a single data source is the solution to address this traffic problem of ambiguity; thus, necessitating fusing of various traffic data. Angela Aida *et al.* [136] conducted experiment in Tanzania using floating car data collected and processed by a centralized server. The information gathered is communicated to road users via several interfaces including web, radio, television, and mobile phone, after estimation is

performed with the MFRI. In Kim and Kang [137], an adaptive navigation system was applied for scalable route determination using EKF; which, had better accuracy than traditional prediction methods. Similarly, Peng *et al.* [138] used a KF method to fuse the information of urban road sections in order to obtain speed information without GPS sampling signals. Their experimental results show that the method can be effective, more precise and able to provide detailed information suitable for road traffic managers as well as offering simple and easy engineering implementation at very small computation costs.

The experimental results from above studies are promising, however, in the fusion process, there could be aggregated latency due to increase in the system's bandwidth, noisiness and longer runtime. Our multimodal traffic data fusion framework addresses some of these concerns using distributed traffic data fusion architecture. This involves our homogeneous data fusion with the MFRI for good interpretation of the heterogeneous data fusion for traffic flow prediction.

The following subsections highlight the major research in terms of the following:

- Data sources for modelling of traffic congestion
- Levels of data fusion
- Data fusion architecture

4.3.1. Data sources for modeling of traffic congestion

- *GPS data*

Lwin and Naing [139] present the estimation of traffic congestion states from road GPS trajectories data collected from mobile phones on vehicles using the Hidden Markov model (HMm). Necula [140, 141] perform the analysis of traffic patterns on street segments based on

GPS data. Kaklij [142] present the data mining of GPS data using clustering and classification algorithms.

- *Simulation data*

Ito and Hiramoto [143] propose a process simulation-based approach for Electronic Toll Collection System (ETC) traffic expressway problems at toll plazas. Kim and Suh [144] use VISSIM, a microscopic multi-modal traffic flow simulation package, to analyze the difference between standard traffic flow inputs and the trip chain method in overcapacity conditions and the sensitivity of the model to this parameter. Metkari *et al.* [145] develop a simulation model for heterogeneous traffic with no lane discipline. He [146] perform the analysis of traffic congestion degree based on spatiotemporal simulation.

- *Sensor data*

Nellore *et al.* [147] use wireless sensor networks for traffic congestion evaluation, control and survey on urban traffic management systems, respectively. Aslam *et al.* [148] present a congestion-aware traffic routing system using system data.

- *Twitter data*

Mai and Hranac [149] present the use of twitter data source to examine the interactions between the tweets and the traffic using the relationships weighted by its relevance to traffic incidents measurement. Elsafoury [111] recommends the use of Part of speech (POS) tag systems to analyze traffic information from micro-bloggers data source like twitter. Chen *et al.* [150] perform road traffic congestion monitoring via social media with Hinge-Loss Markov Fields.

- *Probe data*

Hofleitner *et al.* [151] present a dynamic Bayesian network that provides a flexible framework to learn traffic dynamics from historical data and to perform real-time estimation with streaming data from a probe vehicle. Wang *et al.* [152] develop a hidden Markov model for urban vehicle estimation using probe data from a floating car.

4.3.2. Levels of data fusion

Data fusion in our study context involves multi-sensor fusion. It is used for consolidation of various unstructured, structured and semi-structured data. Using data fusion has the benefit of larger “degree of freedom” within the internal state that contribute to improved estimation on observed measurement. The uncertainty in the data via fusion is also expected to be drastically reduced since it is more accurate, complete or more dependable (Elmenreich [153]). Other types of fusion such as homogeneous images captured from a 360-degree camera remains a prospect for prediction of congested traffic flow. According to Klein [154], various level of data fusion processes comprising of low, intermediate or high will depend on the processing stage at which fusion occurs as found in Steinberg and Bowman [155]; and Blasch and Plano [156] namely:

- Level 0 – Data alignment
- Level 1 – Entity assessment (e.g. signal/feature/object) i.e. tracking and object detection/recognition/identification
- Level 2 – Situation assessment
- Level 3 – Impact assessment
- Level 4 – Process refinement (i.e. sensor management)
- Level 5 – User refinement

Figure 4.3 presents a tree diagram of the above data fusion framework. It is the most popular conceptual model in the data fusion community consisting of five processing levels including an associated database and an information bus that connects the components. Castanedo [157] explains its use based on the Joint Directors of Laboratories (JDL) and the America Department of Defense (DoD) (JDL, Data Fusion Lexicon [158]).

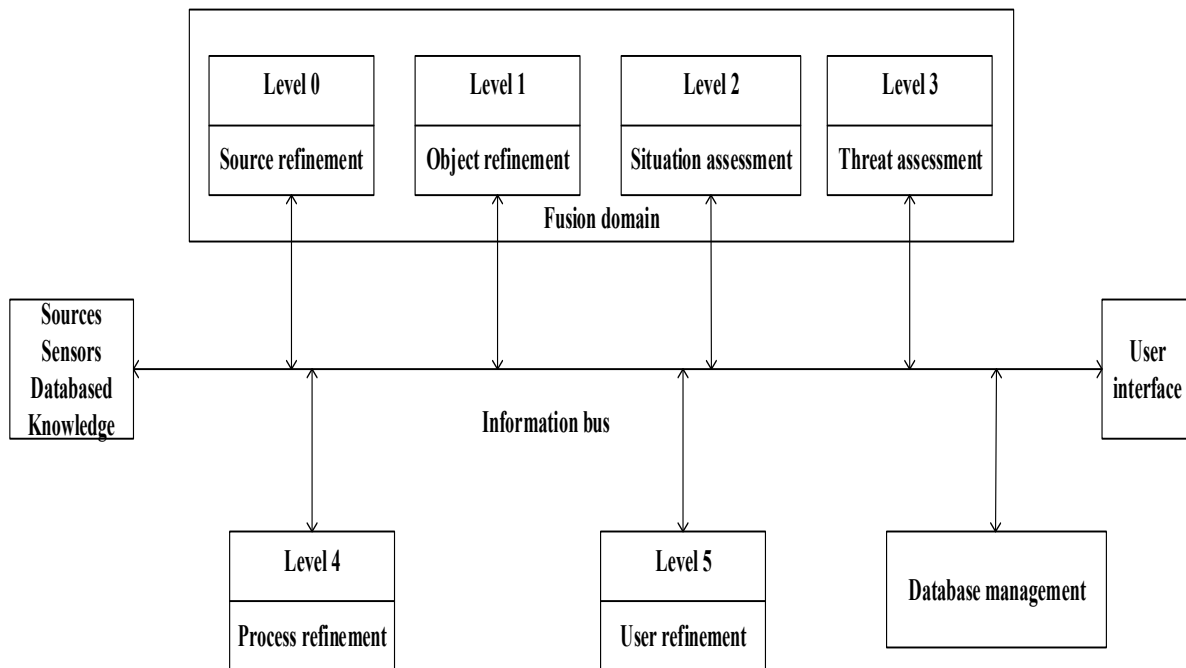


Figure 4.3: Data fusion framework (adapted from the JDL, Data Fusion Lexicon [158])

In addition, accurate and reliable estimation and prediction of traffic flow congestion using data fusion techniques are often done with powerful algorithms such as Bayesian network, Kalman filter (KF), and Dempster-Shafer. It is noteworthy that the main distinction between the Dempster-Shafer theory and KF is that in the Dempster-Shafer theory, each state equation or observation is considered a special case of a linear belief function and the KF is a special case of

combining linear belief functions on a join-tree or Markov tree. Additional approaches include belief filters which use Bayes or evidential updates to the state equations. Anand *et al.* [159] applied KF to fuse the spatial and location-based data. This emphasize that the use of traffic density obtained from location based data is insufficient for making estimation and prediction without including the spatial variation in respect of predicting density for future time intervals using a time-series regression model (Anand *et al.* [159]). Zhanquan *et al.* [160] identify the information fusion levels as data level, feature level, and decision level based on data obtained from loop vehicle detector and GPS floating car and use bayesian and entropy-based weighted methods for the traffic data fusion estimation and prediction. Discussions on the general applications of KF to traffic management vis-à-vis traffic flow prediction can be found on Antoniou *et al.* [161].

4.3.3. Data fusion architectures

There are various data fusion architectures which could be centralized, decentralized, distributed and hierarchical architecture. In Castanedo [157], they posited that data fusion architecture are practically comparable because the selection would depend on the requirements, demands, existing networks, data available, node processing capabilities, and organization of the data fusion system. Crowley and Demazeau [162] presented the principles and techniques of sensory data fusion.

In traffic flow management, the need to fuse traffic data from multiple information sources such as induce loop vehicle detector, video detector, GPS floating car has been supported by Zhanquan *et al.* [160] and Kuwahara and Tanaka [163]. Also, Ben-Akiva *et al.* [164] have recommended the use of real-time Dynamic Traffic Assignment (DTA) systems, for

implementing two important functions: traffic state estimation and traffic state prediction. The DTA is supported by two main modules: a demand simulator that fuses data surveillance information with historical information for the estimation and prediction of the evolving demand patterns; and the second is a supply simulator that is based on high-level (mesoscopic or macroscopic) models that represents traffic dynamics characteristics such as speed-density relationships, kinematic representation of traffic elements of queueing theory. The following describes the most common architectures as found in Castanedo [157] for data fusion:

- **Centralized**

The fusion nodes are located in the central processor that collects all the raw data and uses the provided raw data measurements from the sources to send instructions to the respective sensors. Figure 4.4 illustrates the centralized architecture.

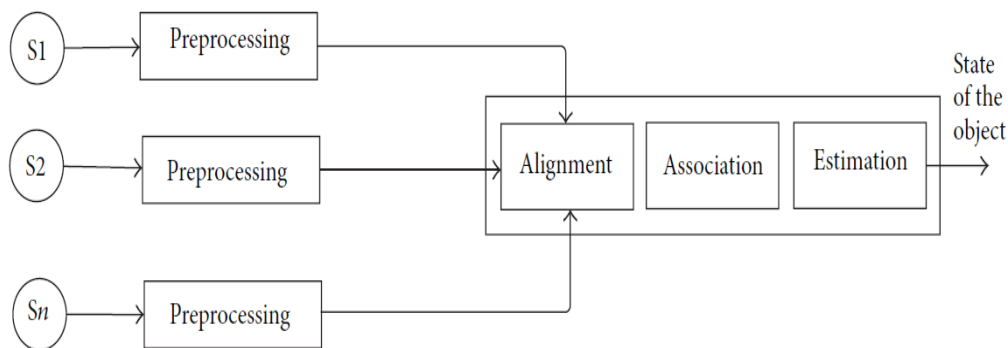


Figure 4.4: Centralized architecture (source: Castanedo [157])

The centralized scheme is theoretically optimal if we assume that the central processor's tasks such as association, filtering, and tracking are performed correctly and data transfer time is insignificant.

The limitation however is that in real systems, large amount of raw data are transferred through the network which requires large amount of bandwidth. This can lead to time delays during transfer of information from different sources.

- ***Decentralized***

The architecture (see Figure 4.5) consists of a network of nodes in which each node has its own central processor. Hence, there is no single point of data fusion such that each node fuses its local information with other information it received from its peers.

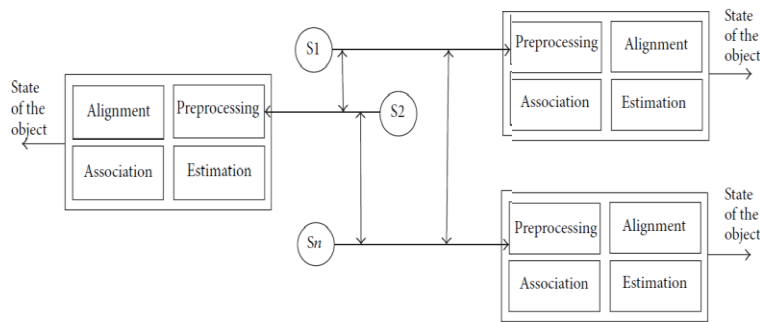


Figure 4.5: Decentralized architecture (source: Castanedo [157])

The decentralized scheme is performed anonymously in order to improve the level of data security. Durrant-Whyte and Stevens [165] explained that the transfer of information from the sources typically employ the Fisher and Shannon measurements instead of the object state. In terms of the disadvantages, high communication cost which is $O(n^2)$ at each communication step is expected, where n is the number of nodes. Also, scalability becomes an issue when the number of nodes is increased where each node communicates with all of its peers.

- **Distributed**

It is an extension of the centralized architecture where measurements from each source node are processed independently before the information is sent to the fusion node. As illustrated in Figure 4.6, the source node is responsible for the data association and state estimation before the information is communicated to the fusion node.

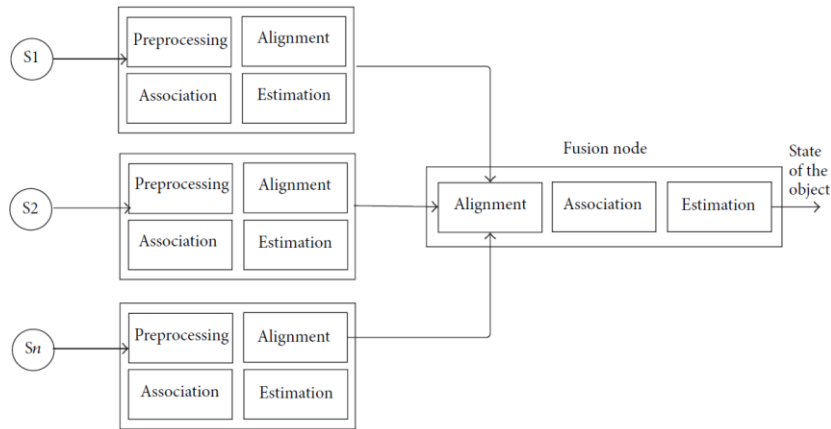


Figure 4.6: Distributed architecture (source: Castanedo [157])

It has an advantage that each node accounts for the estimation of object state that are based on only their local views which become a fused global view when the information is sent as input to the fusion node. The main disadvantage lies in the different options and variations that range from one fusion node to several intermediate fusion nodes. This may result in large amount of bandwidth usage with time delay as found in the centralized architecture.

- **Hierarchical**

This architecture generates hierarchical schemes that combine decentralized and distributed nodes in which data fusion process is performed at different level in the hierarchy.

4.3.4. Data fusion algorithms for traffic congestion estimation

Table 4.1 summarizes few data fusion algorithms for traffic congestion.

Table 4.1: Data fusion algorithms for traffic congestion estimation

Approach	Author	Title	Brief summary
(Discrete-time) Kalman Filter	Yang [166]	Travel time prediction using the GPS test vehicle and Kalman Filtering techniques	A recursive, discrete-time KF is used on historic and real-time data to improve performance monitoring, evaluation, planning, and for efficient management of special events related traffic flow.
	Anand <i>et al.</i> [159]	Data fusion-based traffic density estimation and prediction	They used KF to fuse spatial and location-based data for the estimation of traffic density to future time intervals using a time-series regression model.
Extended Kalman Filter	Kim and Kang [137])	Congestion avoidance algorithm using Extended Kalman Filter	They used KF algorithm for accurate, scalable and adaptable traffic flow prediction for near future congestion based on historical and real traffic information. The user's route preferences is improved using the adaptive traffic route conditions with scalable routing services.
	Guo <i>et al.</i> [167]	Kalman Filter approach to speed estimation using single loop detector measurements under congested conditions	Traffic data from single loop and dual loop station are fused while employing Extend KF to relate the ratio of flow rate over occupancy and the speed. This resulted in more accurate estimation than the traditional g-factor approach.
Bayesian and Neural Network	Pamula and Krol [168]	The traffic flow prediction using bayesian and neural networks	Comparative evaluation of the performance of bayesian and neural networks on short-term traffic congestion models as well as comparing with bayesian dynamic model. The study showed that there is prospect in the use of artificial intelligence methods for forecasting traffic congestion and incorporating them into modules of intelligent traffic management systems.
	Zheng <i>et al.</i> [169]	Short-term freeway traffic flow prediction: bayesian combined neural network approach	For accurate and stable predictions of short-term free traffic flow, two singular neural network predictions were compared with bayesian combined neural network known as BCNN. The results showed that the (hybrid) BCNN outperforms the singular predictors for more than 85% time intervals.

4.4. Solution approach

The distributed architecture as an extension of the centralized data fusion architecture offers measurements from each source node that are processed independently before the information is sent to the fusion node. In our method, while using the distributed architecture, we consider the following data sources: highway traffic API, and Twitter data source. This can be classified into homogeneous and heterogeneous based on the similarity in the data structure and characteristics.

4.4.1. Homogeneous traffic data fusion

The homogeneous traffic data fusion has a distributed architecture with same relational structure and with numeric attributes. This data are obtained from the twitter and highway traffic API. The API traffic data comes from GPS and Sensor data sources. For instance, similar data obtained from the API undergo preprocessing, alignment, association i.e. aggregation and estimation using the ML classification algorithms: NN, RF (random forest), and DBN. Thereafter, data fusion is performed on the outputs with the alignment, association and estimation without further preprocessing. Figure 4.7 illustrates the homogeneous distributed data fusion for short term traffic flow prediction.

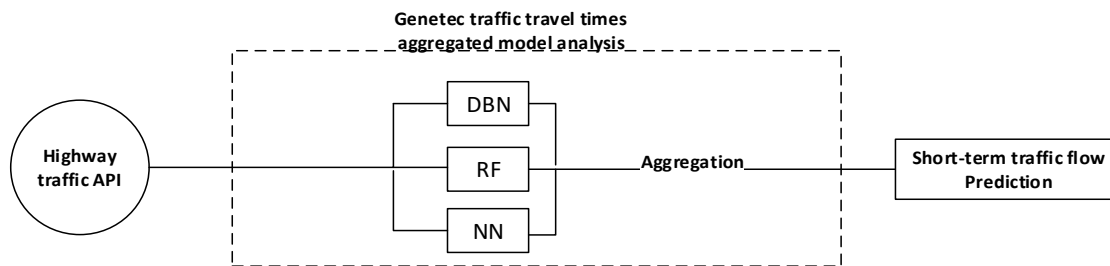


Figure 4.7. Homogeneous distributed data fusion for short-term traffic congestion prediction.

4.4.2. Heterogeneous traffic data fusion

The heterogeneous traffic data fusion uses the distributed architecture with the homogeneous traffic data fusion which now includes the twitter data source. Recall that in Chapter 3, we presented the twitter sentiment analysis and cluster classification. This is on the basis of traffic delay tweets obtained in near real-time with unstructured data having categorical characteristics. Figure 4.8 illustrates heterogeneous distributed data fusion for short-term traffic congestion prediction.

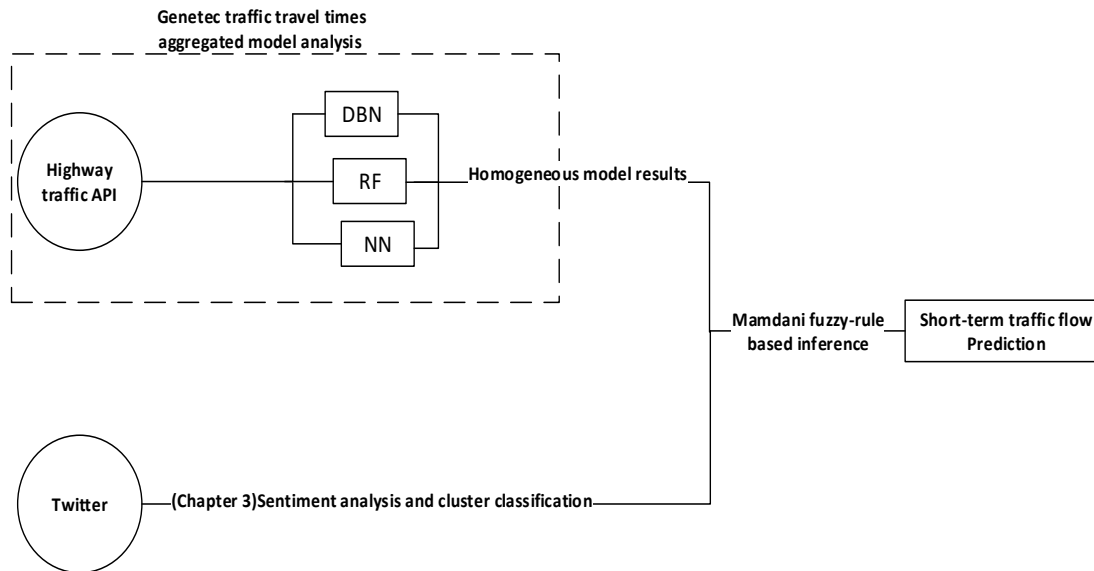


Figure 4.8: Heterogeneous distributed data fusion for short-term traffic congestion prediction.

4.5. Numerical application

We defined set of fuzzy-rules for the traffic delays data tweets using Mamdani inferencing while relying on existing distributed homogeneous model. The MFRI provides an effective fuzzy logic to monitor real-time traffic level detection and decision-making for real-time traffic information so as to reduce congestion based on time, mode and route alternatives (Angela-Aida *et al.* [136]). The fuzzy rules obtained are presented in Figure 4.9 and Table 4.2. The use of the MFRI system is motivated by Awan and Awais [170] and its application for predicting weather events. It involves alignment of the traffic delays Part of Speech (POS) tags based on the sentiment and cluster classification and associating them with the homogenous predictive model.

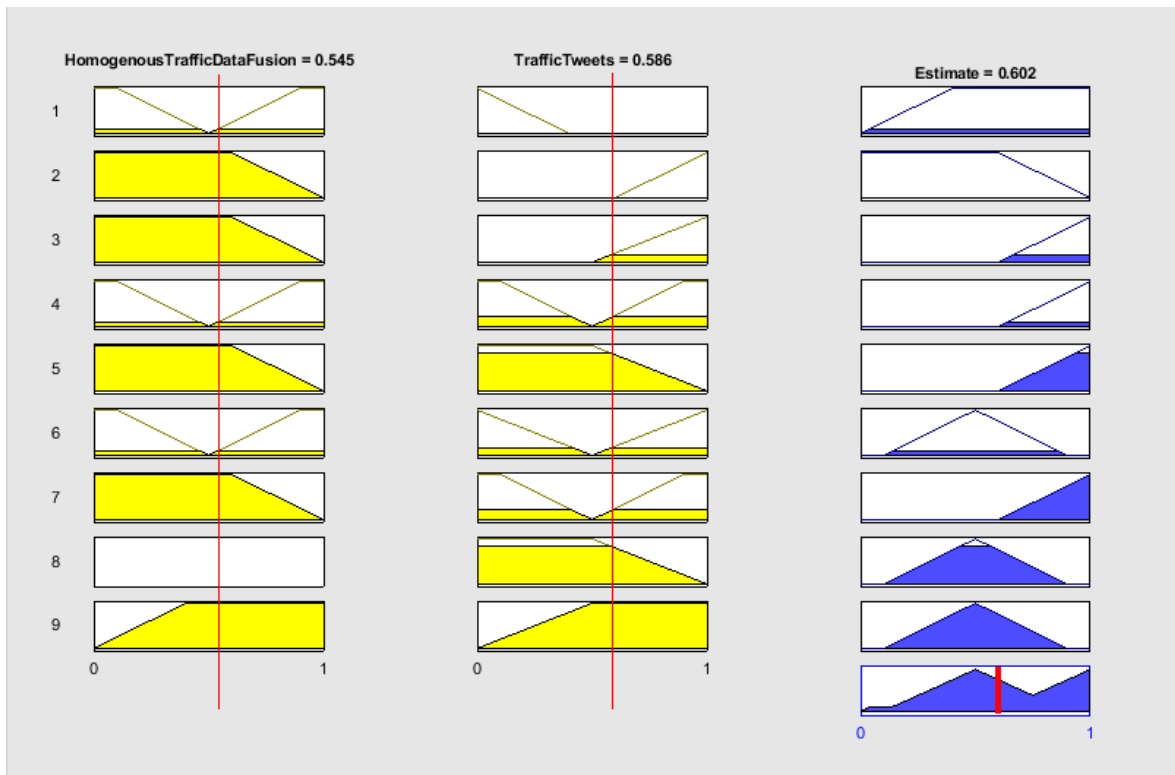


Figure 4.9: Estimations from the MFRI data fusion model

Table 4.2: MFRI data fusion model for twitter sentiment and aggregated data mining model results(ADM MR)

Rule 1:

If (tweets sentiment is positive and ADM MR__Has_High_Congestion_Output)
Then(Possible_High_Congestion)

Rule 2:

If (tweets sentiment is positive and ADM MR__Has_Medium_Congestion_Output)
Then (Possible_MediumCongestion)

Rule 3:

If (tweets sentiment is positive and ADM MR__Has_Low_Congestion_Output)
Then (Possible_Low_Congestion)

Rule 4:

If (tweets sentiment is Neutral and ADM MR__Has_High_Congestion_Output)
Then (High_Congestion)

Rule 5:

If (tweets sentiment is Neutral and ADM MR__Has_Medium_Congestion_Output)
Then (Medium_Congestion)

Rule 6:

If (tweets sentiment is Neutral and ADM MR__Has_Low_Congestion_Output)
Then (Low_Congestion)

Rule 7:

If (tweets sentiment is Negative and ADM MR__Has_High_Congestion_Output)
Then (High_Congestion)

Rule 8:

If (tweets sentiment is Negative and ADM MR__Has_Medium_Congestion_Output)
Then (Medium_Congestion)

Rule 9:

If (tweets sentiment is Negative and ADM MR__Has_Low_Congestion_Output)
Then (Low_Congestion)

We used MATLAB [79] for the Genetec traffic data analytics. This is done in order to predict travel times based on historical data. We consider the RF, NN and DBN regression learner.

Figure 4.10 is an example of the prediction derived from a RF.

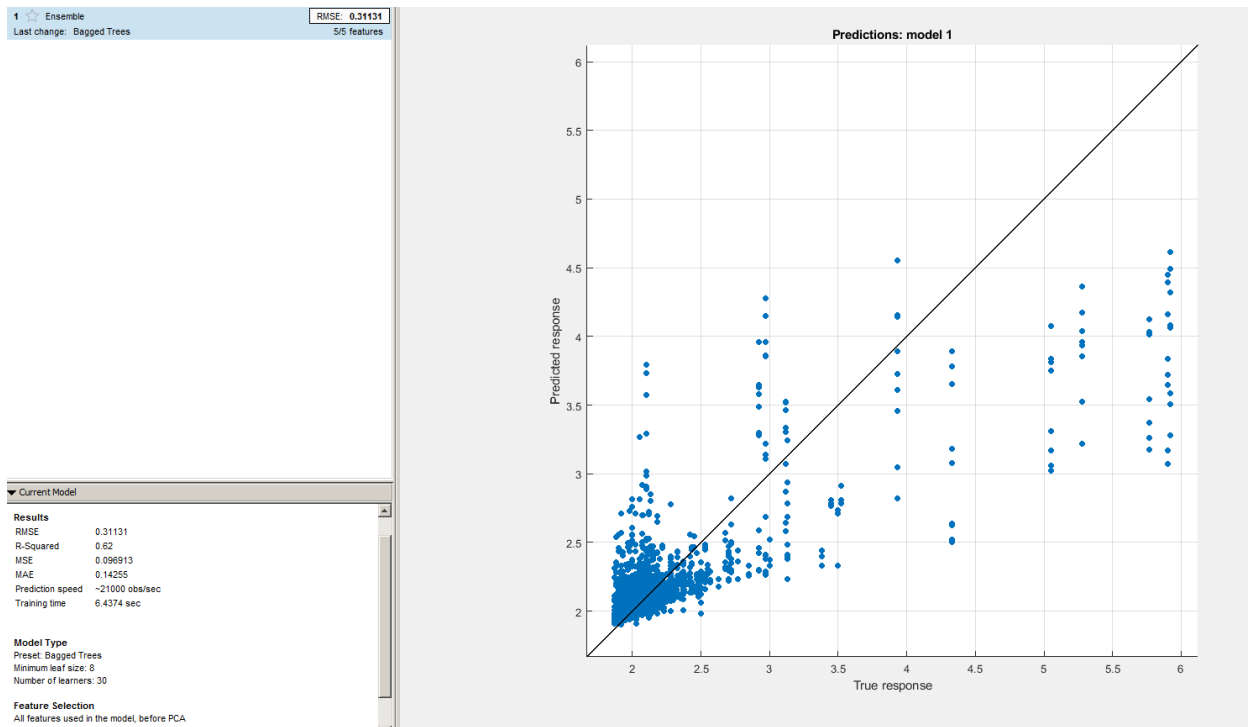


Figure 4.10: Sample prediction based on RF trained on the Genetec traffic data.

From the diagram we see that the $RMSE = 0.31131$, $MAE = 0.096913$ and the prediction speed is approximately 21000 observations/sec. Next, MATLAB [79] *nntraintool* is used to perform the Neural Network training with the same Genetec traffic data. Figure 4.11 illustrates the training instance with single hidden layer for the NN and ten hidden layers for the DBN, respectively.

Figure 4.12-4.13 gives the respective R values for the two cases in 4.11



Figure 4.11 Training with single hidden layer and multiple hidden layer

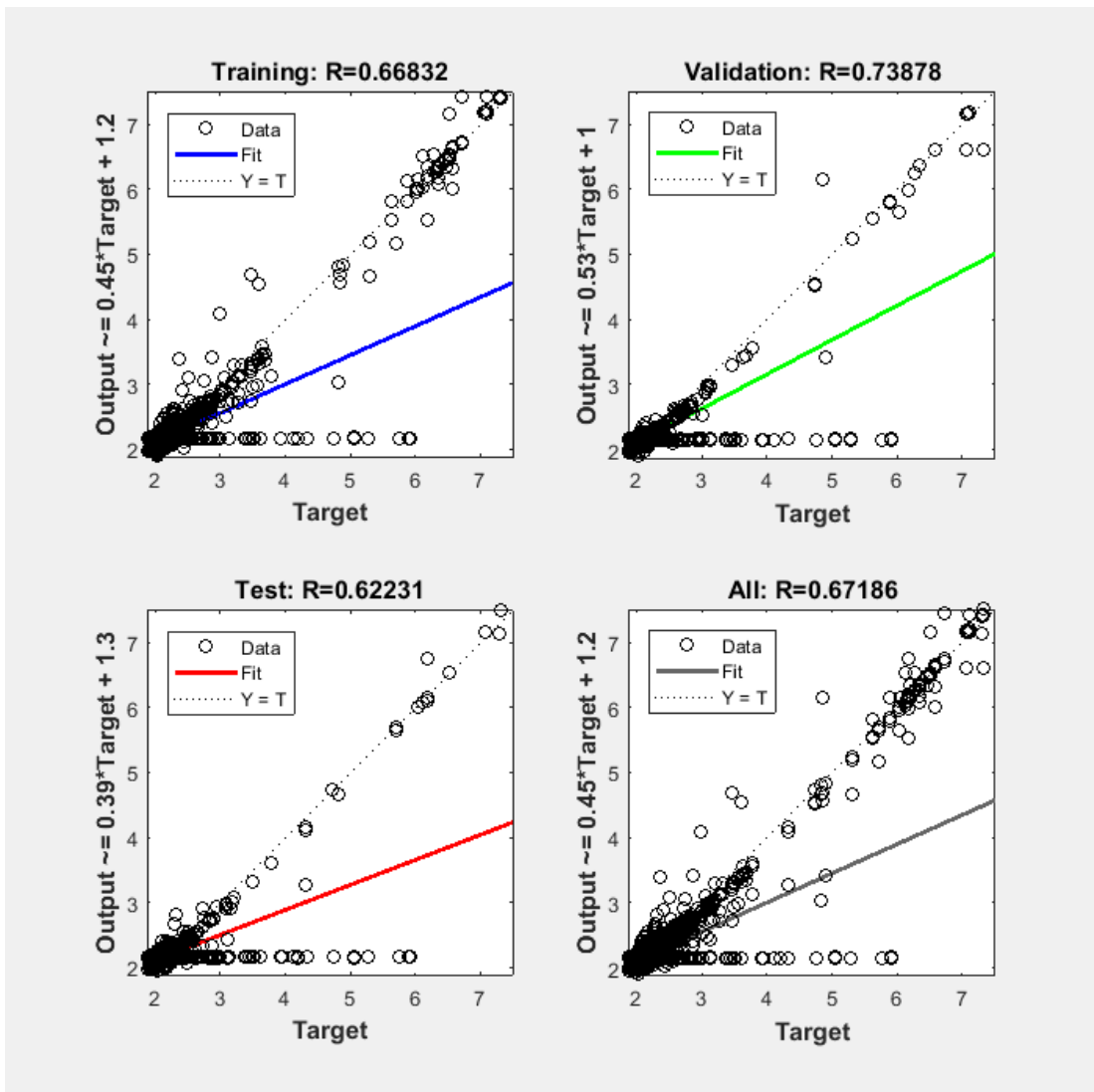


Figure 4.12 Regression plot with R values for NN

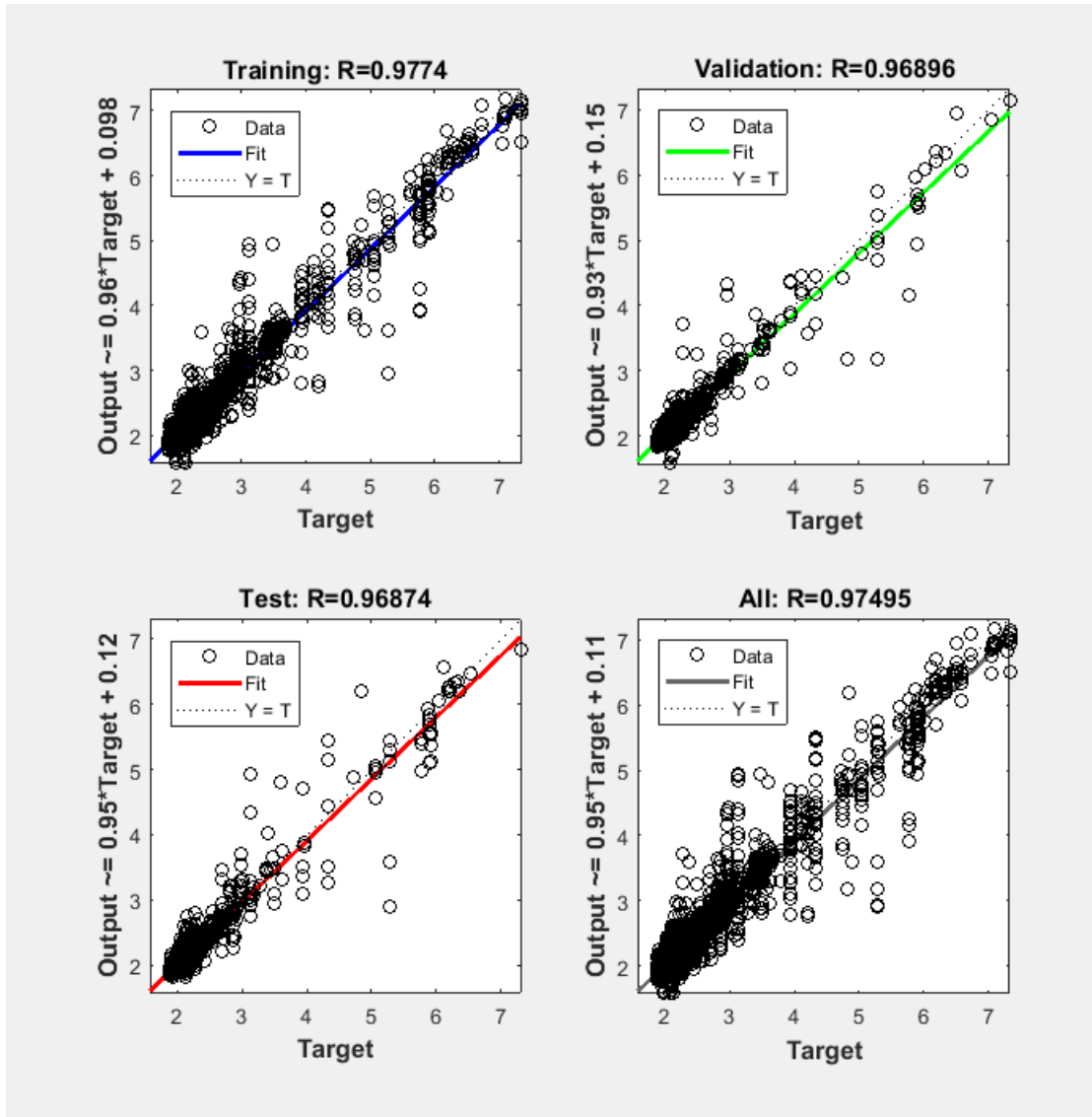


Figure 4.13 Regression plot with R values for DBN

We see that the R (i.e. the correlation coefficient) is high for both the training, testing and validation sets in DBN when compared to the NN. Its value ranges from -1.0 to +1.0. The closer R is to +1 or -1, the better the model optimality.

4.6. Results validation

Our model validation is done by measuring the predicted travel time (*PTT*) required for the vehicles' traversal of segment of the road networks against the *PTT* estimates obtained from GBTTSE. Table 4.3 presents the results.

A sample of the *PTT* for *St Patrick-Upper Lachine MTL A15N* is presented in Figure 4.14. The *mean travel time* is the mean of the travel times between the two nodes e.g. from St. Patrick to Upper Lachine. The calculated travel times provide the various travel time used for calculating the traffic congestion predictions on the road network.

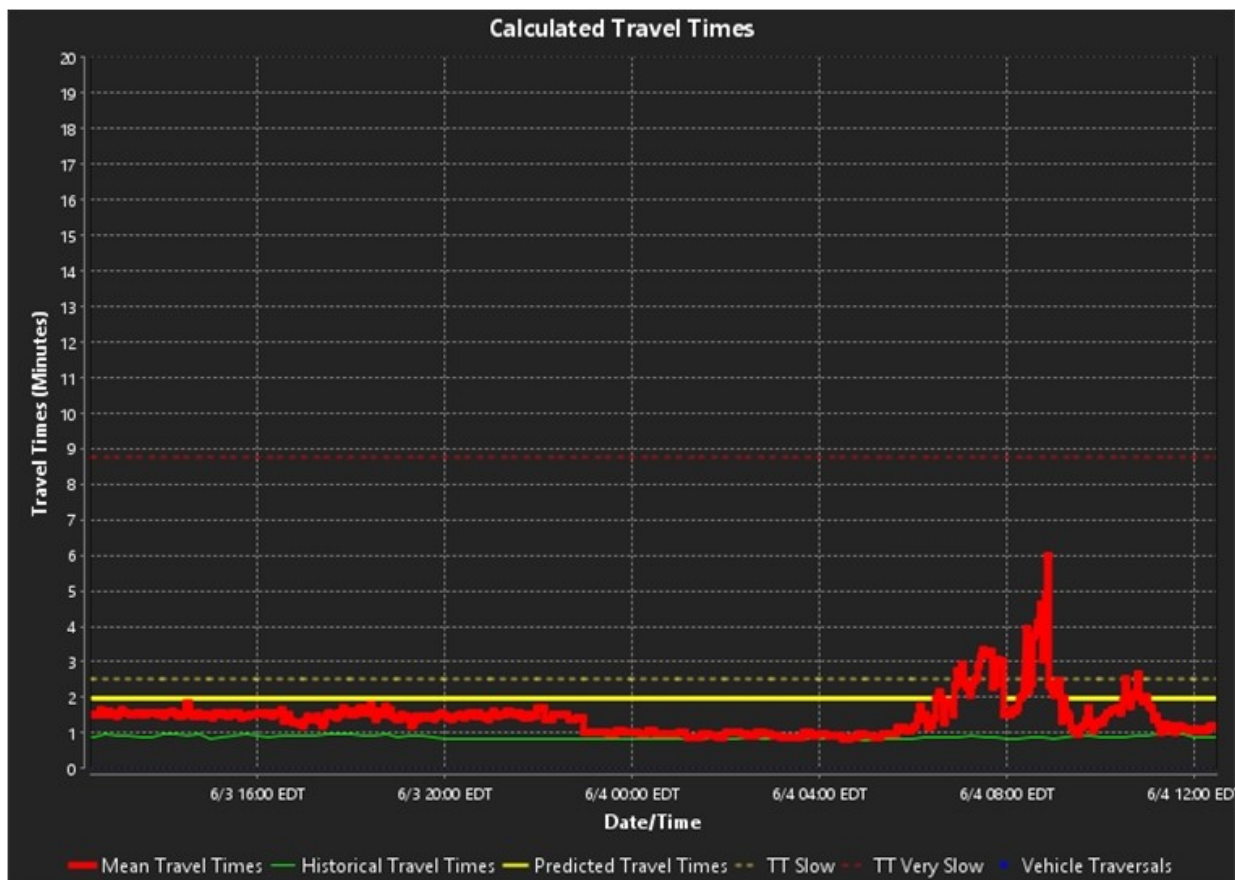


Figure 4.14: Example of calculated TTs from GBTTSE

MTL Motorway networks	Source node	Destination node	Current TT(mins)		Historical TT(mins)		PTT(mins)	PTT(mins)
			t	$t - 1$	t_{hist}	$t_{hist} + 1$	t_{pred} GBTSE	t_{pred} Homogeneous data fusion model(i.e. ADMMR)
							Accuracy (%)	Accuracy (%)
A15N	Champlain	Atwater	2.15	2.12	1.18	2.23	2.00	1.81
							93.02	84.19
	St Patrick	Upper Lachine	2.28	2.32	1.95	1.87	1.96	1.85
							85.96	81.14
	Upper Lachine	St Luc	1.05	1.02	1.45	1.65	1.85	1.66
							23.80	41.90
	St Luc	Cote St Catherine	1.62	1.58	1.47	1.53	1.61	1.78
							99.34	90.12
	Cote St Catherine	Jean Talon	1.05	0.62	0.92	1.03	1.08	0.99
							97.22	94.29
	Jean Talon	Duncan	1.95	1.95	2.03	1.68	2.08	1.98
							93.33	98.46
	Duncan	Lucerne	4.48	4.48	3.03	4.06	3.04	4.05
							67.85	90.40
	Dunkirk	Sauve	2.05	1.97	1.93	2.02	2.12	2.09
							96.59	98.05
	Sauve	Salaberry	2.17	2.17	2.14	2.21	2.13	2.40
							98.16	89.4
Salaberry	Cartier	1.17	1.95	1.02	1.10	1.86	1.75	
						58.03	50.43	
AL100	IlleSoeur	FernandSeng	2.82	2.80	2.83	2.86	2.78	2.89
							98.58	97.52
	FernandSeguin	Irlandais	1.45	1.45	1.56	1.50	1.53	1.50
							94.48	96.55
	Irlandais	Wellington	1.18	1.20	1.06	1.08	1.08	1.11

¹ These results are based on the sample or test data and hence, cannot be generalized

							91.52	94.07
ReneLevW-	Papineau	St Denis	2.85	2.83	2.62	2.60	2.72	2.98
							95.44	95.44
	St Denis	St Laurent	1.62	2.72	1.65	1.71	1.68	1.56
							96.30	96.30
	St Laurent	University	2.45	2.45	1.86	1.95	1.62	2.40
66.12							97.96	
Peel	Guy	1.60	1.60	1.30	1.30	1.52	1.55	
						95.0	96.88	
Guy	Atwater	1.73	1.73	1.52	1.50	1.64	1.59	
						94.80	91.91	
PieIXN1	Notre-Dame	Sherbrooke	3.8	3.8	3.15	3.12	3.50	2.38
							92.11	62.63
PieIXN2	Sherbrooke	Rosemont	3.05	2.81	2.5	3.02	2.61	3.01
							85.57	98.69
PieIXN3	Rosemont	Jean Talon	2.72	2.70	2.30	2.41	2.71	2.68
							99.63	98.53
PieIXN4	Jean Talon	Jarry	1.87	1.84	1.68	1.70	1.43	1.50
							76.47	80.21
PieIXN5	Jarry	HenriB	8.87	8.87	9.02	10.50	9.82	8.73
							89.29	98.42
RTM	Lachine	LucienAllier	22.18	22.18	21.87	20.56	21.69	20.89
							97.79	94.18
	LucienAllier	PalaisCongres	12.03	15.0	16.68	18.26	19.00	18.63
							42.06	45.14

The accuracy of the prediction is defined as the degree of closeness of a measured or calculated value to its actual value. The formula is $(1 - \textit{percentage error})$ where the percentage error is given by:

$$\textit{percentage error} = \frac{|\textit{current TT} - \textit{predicted TT}|}{\textit{current TT}} \times 100\% \quad (4.14)$$

We see that the *PTT* for our model is better than that of GBTTSE in 12 cases while the GBTTSE is better in 11 cases out of the total. In two instances, both models performed equally well in *PTT*.

Figure 4.15 provides a sample plots of the selected algorithms applied on the test data.

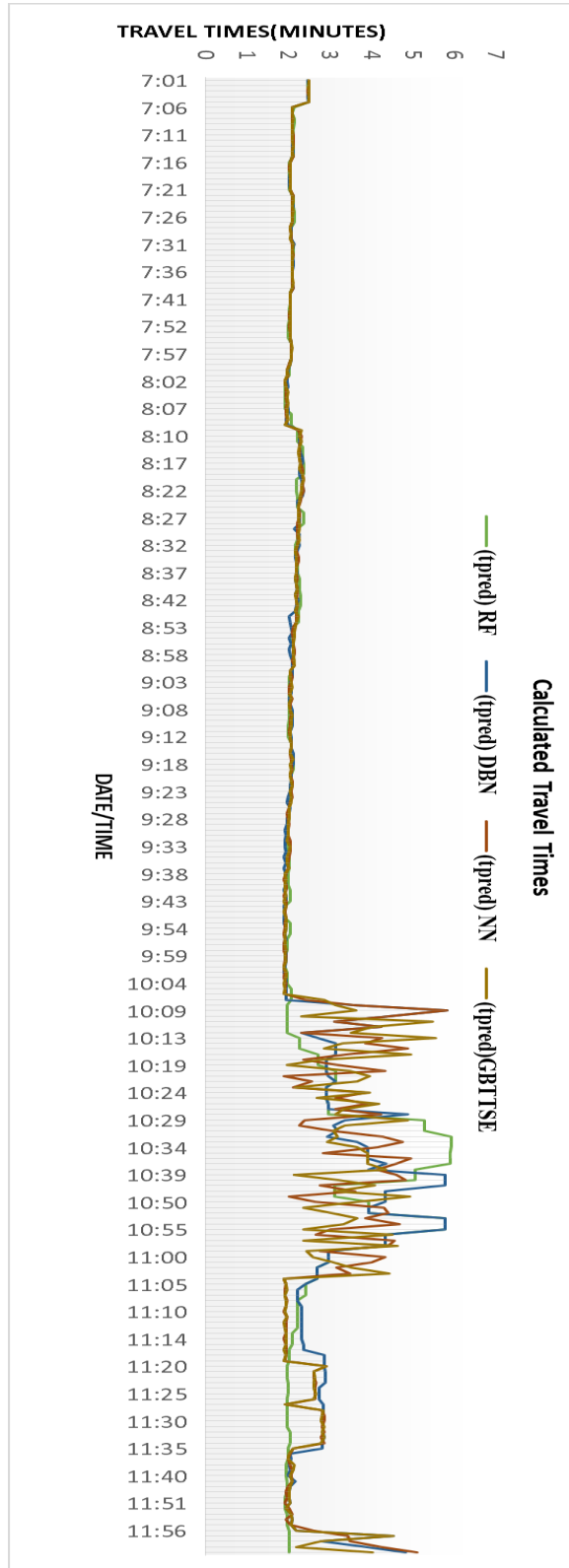


Figure 4.15: Example of calculated TTs from RF, DBN, NN, and GBTTSE

4.7. Conclusions

In this chapter, we propose a multi-modal big data fusion framework for traffic congestion prediction. It involves distributed traffic data fusion architecture with homogenous (numeric only) and heterogeneous (categorical and numeric) data. For the homogenous data fusion, using highway traffic dataset, we predict the traffic travel times using data mining algorithms comprising of DBN, NN and RF. For the heterogeneous data, we integrate the homogeneous fusion information with twitter traffic data and applied MFRI for good interpretability. Our results emphasize the improvements made in the prediction of travel times using traffic big data of vehicles' traversing various road network nodes in the city of Montreal. The model validation is done with the GBTTSE.

The strength of this research study is the use of big data fusion for traffic congestion prediction in near real-time traffic congestion states (i.e. low, medium and high). This is on the basis of the traffic travel times of road vehicles on urban motorways. The limitation is lack of adequate system tools to seamlessly integrate data from various sources for real-time traffic information.

As future work, one could also consider the use of real-time traffic image, video and text data to improve traffic congestion prediction while integrating them to our existing framework. Secondly, the study can be extended while considering various geographical contexts, connected locations, merges, ramps etc.

Chapter 5

Conclusions and future works

5.1. Conclusions

In this thesis, we address the problem of short-term traffic congestion prediction on urban motorway networks. A multi-modal data fusion framework is proposed. Three categories of models are developed. The first category comprises of data mining or machine learning models for traffic congestion prediction. The second category comprises of social media traffic data analysis (i.e. twitter) using sentiment analysis and classification. The third and the last category comprises of data fusion models for homogenous (quantitative) and heterogeneous (quantitative, qualitative) data types. The homogeneous data fusion method aggregates the results of various data mining algorithms. The heterogeneous data fusion method extends the homogenous method using the MFRI for the model interpretability. The results obtained are promising and provides useful insights on near real time traffic congestion prediction.

The strength of our data mining model is the use of deep belief network for traffic congestion prediction. This is chosen based on the state of the art algorithm. The performance, in terms of accuracy and runtime, is compared with the back propagation neural network (a traditional algorithm) and the random forest. While, there is little contribution made with the approach using twitter sentiment and classification for traffic congestion predict; it provides evidential support vis-à-vis the interpretability of our distribution traffic data fusion framework with the MFRI. Lastly, the major strength of this work is the development of the distributed homogenous and heterogeneous data fusion framework for traffic congestion prediction.

The usefulness of our research to academia and industry includes better intelligent route planning, monitoring and mitigation of traffic congestion on urban motorways by traffic management systems, reduction of traffic delays, waiting times, air pollution and noise in cities. In addition, it would help the road vehicle drivers to avoid congested traffic route. It would also help the first responders to determine the root cause of traffic congestion based on the sentiment analysis and classification of traffic congestion prediction from twitter data source; as well as assist the traffic managers to design better road traffic infrastructure.

5.2. Future works

Several extensions can be made to the various methods proposed in this thesis.

- For data mining models, further solutions can be investigated to reduce the computation time for DBN.
- For twitter data analysis, precision of our cluster classification algorithm can be improved. Also, investigating the reliability for seamless integration with well-known traffic management software system tools should be explored.
- Lastly, for data fusion work, improvement of the predictive computation for optimum performance is recommended. Also, more investigation of the cause of latency should be explored.

References

- [1] TomTom, "TomTom Traffic and Historical Traffic data," 2015. [Online]. Available: http://www.tomtom.com/en_ca/trafficindex/list. [Accessed 29 March 2016].
- [2] Omrani H., Charif O., Gerber P. , Awasthi, A. and Trigano, P., "Prediction of individual travel mode using evidential neural network model," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2399, no. 1-8, 2014.
- [3] Hamner, B., "Predicting Future Traffic Congestion from Automated Traffic Recorder Readings with an Ensemble of Random Forests," in *IEEE International Conference on Data Mining Workshops*, DOI 10.1109/ICDMW.2010.169, 2010.
- [4] Leshem, G. and Ritov. Y., "Traffic flow prediction using adaboost algorithm with Random Forests as a weak learner," *World Academy of Science, Engineering and Technology, International journal of Mathematical, Computational, Statistical, Natural and Physical Engineering*, vol. 1, no. 2, pp. 1-6, 2007.
- [5] Jain, S.,Jain S.S., and Jain, G., "Traffic Congestion Modelling Based on Origin and Destination," in *10th International Scientific Conference Transbaltica 2017: Transportation Science and Technology*, Elsevier Ltd, 2017.
- [6] Lomax. T., Turner, S.,Shunk, G. , Levinson, H. S. , Pratt, R. H. , Bay, P. N. , Douglas, G. B., "Quantifying congestion," in *Transportation Research Board 1&2, NCHRP Report 398*, Washington DC, 1997.
- [7] Jain, V., Sharma, A., and Subramanian, L., "Road traffic congestion in the developing world," in *ACM DEV '12 Proceedings of the 2nd ACM Symposium on Computing for Development*, Atlanta, Georgia , 2012.

- [8] Eisele W.L, Rilett, L.R., Mhoon, K.B. and Spiegelman, C., "Using intelligent transportation systems travel-time data for multimodal analyses and system monitoring," *Transportation Research Record*, 1768, Paper No. 01-2834, 2001.
- [9] Kumar, K., Parida, M. and Katiyar, V.K., "Short term traffic flow prediction for a non urban highway using Artificial Neural Network," in *2nd Conference of Transportation Research Group of India (2nd CTRG)*, Procedia - Social and Behavioral Sciences, Elsevier, 2013.
- [10] Zhang, F., Li J.L, and Zhao Q.X, "Single-lane traffic simulation with multi-agent system," in *In: Intelligent Transportation Systems, Proceeding*, Vienna, 2005.
- [11] Awasthi, A., Parent, M., and Proth, J-M., "Case-based modelling and simulation of traffic flows on motorway networks," *International Journal of Modelling and Simulation*, vol. 26, no. 3, pp. 251-260, 2006.
- [12] Rehborn, H., and Klenov, S.L., "Traffic Prediction of Congested Patterns," in *Encyclopedia of complexity and systems science*, R. A. Meyers, Ed., 2009, p. 9536.
- [13] Kerner, B.S., "The Physics of Traffic," Springer, Berlin, New York, 2004.
- [14] Kerner, B.S., "Traffic congestion, modelling approaches to," in *Encyclopedia of Complexity and Systems Science*, R. A. Meyers, Ed., Springer New York, 2009.
- [15] Lint, J.W.C.V., "Reliable traffic time prediction for freeways," Unpublished PhD thesis, Netherlands TRAIL Research School, 2004.
- [16] Blip systems, "Smart traffic sensors help alleviate traffic congestion in switzerland," 16 February 2016. [Online]. Available: <http://blipsystems.com/smart-sensor-switzerland/>. [Accessed 13 August 8].
- [17] Banks J., Carson J. , Nelson, B. ; and Nicol, D., "Discrete-Event System Simulation," Prentice Hall, 2001, p. 3.

- [18] Adetiloye, T. and Awasthi, A, "Part 2: Simulated Traffic Congestion Visualization and Analytics," 1 March 2016. [Online]. Available: <https://www.youtube.com/watch?v=UMZK11AzD3I>. [Accessed 13 August 2016].
- [19] Young, S., "Real-Time Traffic Operations Data Using Vehicle Probe Technology," 2007.
- [20] Sato, M., "Privacy concerns with big data from probe vehicle systems," *Coordinates*, April 2013. [Online]. Available: <http://mycoordinates.org/privacy-concerns-with-big-data-from-probe-vehicle-systems/>. [Accessed 13 August 2016].
- [21] Howell, E., "Navstar: GPS Satellite Network," 14 February 2013. [Online]. Available: <http://www.space.com/19794-navstar.html>. [Accessed 13 August 2016].
- [22] Hortonworks, "Realtime event processing in Hadoop with Nifi, Kafka and Storm," [Online]. Available: <http://hortonworks.com/hadoop-tutorial/realtime-event-processing-nifi-kafka-storm/>. [Accessed 15 August 2016].
- [23] EndoCode, "Building a stream processing pipeline with Kafka, Storm and Cassandra – Part 1: Introducing the components," April 2013. [Online]. Available: <https://endocode.com/blog/2015/04/08/building-a-stream-processing-pipeline-with-kafka-storm-and-cassandra-part-1-introducing-the-components/>. [Accessed 15 August 2016].
- [24] Zhang, X., Onieva, E., Perallos, A., Osaba, E., Lee, V. CS, "Hierarchical fuzzy rule-based system optimized with genetic algorithms for short term traffic congestion prediction," *Transportation Research Part C*, vol. 43, pp. 127-142, 2014.
- [25] Baskar, L.D., De Schutter, B.D. and Hellendoorn, H., "Traffic Management System for Automated Highway Systems Using Model-Based Predictive Control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 838-846, June 2012.
- [26] Lyons, G.D, McDonald, M., Hounsell, N.B., Williams B., Cheese, J., and Radia, B, "Urban traffic management: the viability of short term congestion forecasting using artificial neural networks,"

Association for European Transport, 1996.

- [27] Joshi, M.R. and Hadi, T.H., "A Review of Network Traffic Analysis and Prediction Techniques," School of Computer Sciences, North Maharashtra University, 2015.
- [28] Dougherty, M.S. and Cobbert, M.R., "Short-term inter-urban traffic forecasts using neural networks," *International Journal of Forecasting*, vol. 13, pp. 21-31, 1997.
- [29] Theja, P.V.V.K., and Vanajakshi, L, "Short term prediction of traffic parameters using support vector machines technique," in *Proceedings of Third International Conference on Emerging Trends in Engineering and Technology*, 2010.
- [30] Zarei, N., Ghayour, M.L. and Hashemi, S., "Road traffic prediction using context-aware random forest based on volatility nature of traffic flows," in *Lecture Notes in Computer Science*, vol. 7802 of the series Lecture Notes in Computer Science, N. A. a. H. H. Selamat A., Ed., Kuala Lumpur, Malaysia, 5th Asian Conference, ACIIDS 2013, Kuala Lumpur, Malaysia, March 18-20, 2013, Proceedings, Part I, 2013, pp. 196-205.
- [31] Hiri-O-Tappa, K., Pan-ngum, S., Narupiti, S. and Pattara-Atikom, W., "Development of real-time short-term traffic congestion prediction method," [Online]. Available: <http://www.thaitransport.org/journal/index.php/Path/article/viewFile/62/65>. [Accessed 2016 7 2016].
- [32] Lopez-Grazia, P., Onieva, E. and Osaba, E., Masegosa, A.D., and Perallos, A., "A Hybrid Method for Short-Term Traffic Congestion Forecasting Using Genetic Algorithms and Cross Entropy," *IEEE Transactions on Intelligent Transportation Systems* , vol. 17, no. 2, pp. 557-569, 2016.
- [33] Tan, M.C., Wong, C.M., Xu, J.M., Guan, Z.H., and Zang P., "An Aggregation Approach to Short-Term Traffic Flow Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 60-69, 2008.
- [34] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back-

- propagating errors," *Nature*, 323, 533--536, 1986.
- [35] Nielsen, M.A., *Neural networks and deep learning*, Determination Press, 2015.
- [36] Barga, R., Fontama, V., and Tok, W.H., *Predictive Analytics with Microsoft Azure Machine Learning: Build and Deploy Actionable Solutions in Minutes*, California: Apress Media LLC, 2014.
- [37] Vlahogianni, E.I., Karlaftis, M.G., and Golias, J.C., "Optimized and meta-optimized neural networks for traffic flow prediction: A genetic approach," *Transportation Research Part C*, vol. 13, pp. 211-234, 2005.
- [38] Gilmore, J.F. and Abe, N. , "Neural network models for traffic control and congestion prediction," *IVHS Journal*, vol. 2, no. 3, pp. 231-252, 1995.
- [39] Gilmore, J.F., Elibiary, K.J. and Abe, N., "Traffic management application of neural networks," *AAAI Technical Report WS-93-04*, pp. 85-95, 1993.
- [40] Rowe, E. , "The Los Angeles Automated Traffic Surveillance and Control(ATSAC) System," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 2, 1991.
- [41] Dia, H., "An object oriented neural network for short term traffic forecasting," *European Journal of Operational Research*, vol. 13, pp. 644-654, 2001.
- [42] Zadeh L.A, "Fuzzy Sets," *Information and Control*, pp. 338-353, 1965.
- [43] Jang, J-S R., Sun C-T, Mizutani, E, *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*, prentice hall, 1997.
- [44] Zadeh, L.A, "The Concept of a Linguistic Variable and its Application to Approximate Reasoning-I," *Information Sciences*, vol. 8, pp. 199-249, 1975.
- [45] Nof, S.Y. , *Handbook of Industrial Robotics*, 2nd Edition, John Wiley and Sons, 1999.
- [46] Zhou, R. W.,and Quek, C. , "POPFNN: A Pseudo Outer-product Based Fuzzy Neural Network," *Neural Networks*, vol. 9, no. 9, pp. 1569-1581, 1996.

- [47] Takagi, T. and Sugeno, M., "Fuzzy Identification of Systems and its Applications to Modeling and Control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 51, no. 1, p. 116–132, 1985.
- [48] Sugeno, M. and Kang, G.T., "Structure Identification of Fuzzy Model," *Fuzzy Sets and Systems*, vol. 28, no. 1, pp. 15-33, 1988.
- [49] Schnitman, L., Felipe de Souza, J.A.M., and Yoneyama, T., "Takagi-Sugeno-Kang Fuzzy Structures in Dynamic System Modeling," 2001.
- [50] Ling H., Peiqun, L., and Jianmin, X., "Urban road network traffic congestion prediction model based on probe vehicle technology," *Journal of Highway and Transportation Research and Development*, pp. 88-92, 2011.
- [51] Li, L., Lin W.-H. and Liu H., "Type-2 fuzzy logic approach for short-term traffic forecasting," *IEEE Proceedings Intelligent Transportation System*, vol. 153, no. 1, pp. 33-40, 2006.
- [52] Lu, J. and Cao, L., "Congestion Evaluation from Traffic Flow Information based on Fuzzy Logic," *Intelligent Transportation Systems, 2003. Proceedings*, vol. 1, pp. 50 - 53, 2003.
- [53] Ogunwolu, L., Adedokun, O., Orimoloye, S. and Oke, S.A., "A neuro-fuzzy approach to vehicular traffic flow prediction for a metropolis in a developing country," *Journal of Industrial Engineering International*, vol. 7, no. 13, pp. 52-66, 2011.
- [54] Bengio, Y.; Courville, A.; and Vincent, P, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, p. 1798–1828, 2013.
- [55] Collobert, R. and Weston, J., "A unified architecture for natural language processing: deep neural networks with multitask learning," *In Proc. 25th ICML*, pp. 160-167, 2008.
- [56] Hinton, G.E. and Salakhutdinov, R.R., "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504-507, 2006.

- [57] Hinton, G. E., and Sejnowski, T. J, "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Cambridge, MA:MIT Press 1, 282–317, 1986.
- [58] Deng, L. and Yu, D., "Deep learning method and application," *Foundations and Trends in Signal Processing*, vol. 7, no. 3-4, pp. 1-199, 2004.
- [59] Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H., "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems*, vol. 19, 2006.
- [60] Friedman J.H., "An overview of predictive learning and functional approximation," in *in: V. Cherkassky, J.H. Friedman (Eds.), From Statistics to Neural Networks, Theory and Pattern Recognition Applications*, NATO ASI Series F, vol. 136, Springer, 1994, p. 1–61..
- [61] Hinton, G.E, "Deep belief networks," *Scholarpedia 4 (5)*, vol. 4, no. 5, p. 5947, 2009.
- [62] Schmidhuber, J. , "Deep learning in neural networks: an overview," *Neural Networks*, p. 61: 85–117, 2015.
- [63] Lv Y., Duan Y., Kang W., Li Z., and Wang F.Y., "Traffic flow prediction with big data: a deep learning approach," *Transaction of Intelligent Transportation System*, vol. 16, no. 2, pp. 865-873, 2015.
- [64] Ma, X. Yu, H., Wang, Y and Wang, Y., "Large-scale transportation network congestion evolution prediction using deep learning theory," *PLoS ONE*, vol. 10(3): e0119044, 2015.
- [65] Bengio, Y., "Learning Deep Architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1-127, 2009.
- [66] Hinton, G.E, Osindero, S., and Teh, Y. W., "A fast learning algorithm for deep belief nets," *Neural Computing*, vol. 18, p. 1527–1554, 2006.
- [67] O'Connor, P., Neil, P. Liu, S.H., Delbruck, P. and Pfeiffer, M., "Real-time classification and sensor

- fusion with a spiking deep belief network," *Institute of Neuroinformatics, University of Zurich and ETH Zurich, Zurich, Switzerland*, 2013.
- [68] Teh, Y.W. and Hinton, G.E., "Rate-coded restricted boltzmann machines for face recognition," *In: Advances in Neural Information Processing Systems*, pp. 908-914, 2001.
- [69] Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., and Elvezia, C., "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *Guide to Dynamical Recurrent Neural Networks*. eds S. C. Kremer and J. F. Kolen (New York, NY: IEEE Press), 237–244, 2001.
- [70] Blum, A.L. and Rivest, R.L., "Training a 3-node neural network is NP-complete," *Neural Networks*, vol. 5, no. 1, pp. 117-127, 1992.
- [71] Judd, J.S., "Neural network modeling and connectionism," *Neural Network Design and the Complexity of Learning*, MIT Press, 1990.
- [72] Huang, W., Hong, H., Li, M., Hu, W., Song, G., and Xie, K., "Deep architecture for traffic flow prediction," Vols. Part II, LNAI 8347, pp. 165-176, 2013.
- [73] Breiman, L., "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.
- [74] Liaw, A. and Wiener M., "Classification and Regression by randomForest," *R News*, vol. 2, no. 3, pp. 18-22, December 2002.
- [75] Cutler, A., Cutler, D.R., and Stevens, J.R., "Random Forests," in *Ensemble Machine Learning*, 2012, pp. 157-175.
- [76] Smith, B.L. and Demetsky, M.J., "Short-term traffic flow prediction models - A comparison of neural networks and non parametric regression approaches," in *IEEE*, 1994.
- [77] Zhang, H., Ritchie, S.G. and Z, Lo., "Macroscopic modeling of freeway traffic using an artificial neural network," *Transportation Research Record*, 1993.
- [78] Transports, Mobilite durable et Electrificaation des transport, "Quebec511," 2017. [Online].

- Available: <http://www.quebec511.info/en/cameras/montreal/index.aspx>. [Accessed 2 12 2017].
- [79] MATLAB and Statistics Toolbox Release 2012b, The MathWorks, Inc., Natick, Massachusetts, United States., [Online].
- [80] Reynolds, D., "Gaussian Mixture Models," MIT Lincoln Laboratory, 244 Wood St., Lexington, MA 02140, USA, nd.
- [81] MathWorks, "Tracking Cars Using Foreground Detection," 2017. [Online]. Available: <https://www.mathworks.com/help/vision/examples/tracking-cars-using-foreground-detection.html>. [Accessed 14 12 2017].
- [82] Lu, J, Chen, S., Wang, Wei, and Zuylen, H. v, "A hybrid model of partial least squares and neural network for incident detection," *Expert Systems with Applications*, vol. 39, pp. 4775-4784, 2012.
- [83] Bach, F. and Moulines, E., "Non-asymptotic analysis of stochastic approximation algorithms for machine learning," *Advances in Neural Information Processing Systems (NIPS)*, pp. 1-9, 2011.
- [84] Kiwiel, K. C., "Convergence and efficiency of subgradient methods for quasiconvex minimization.," *Mathematical Programming (Series A)*, vol. 90, no. 1, pp. 1-25, 2001.
- [85] Tensorflow , "Convolutional Neural Networks," 2017. [Online]. Available: https://www.tensorflow.org/tutorials/deep_cnn. [Accessed 14 12 2017].
- [86] SparkML, "Classification and regression," 2017. [Online]. Available: <https://spark.apache.org/docs/latest/ml-classification-regression.html>. [Accessed 11 11 2017].
- [87] "About Tensorflow- An open-source software library for Machine Intelligence," 2017. [Online]. Available: <https://www.tensorflow.org/>. [Accessed 2017].
- [88] "TensorFlow Wide & Deep Learning Tutorial," 2017. [Online]. Available: https://www.tensorflow.org/tutorials/wide_and_deep. [Accessed 17 12 2017].
- [89] Adetiloye, T., "Using Tensorflow to Predict Traffic Congestion Types," 2018. [Online]. Available:

<https://github.com/taiwotman/TensorflowPredictCongestionTypes>.

- [90] Spark MLlib, "MLlib is Apache Spark's scalable machine learning library.," [Online]. Available: <https://spark.apache.org/mllib/>. [Accessed 31 02 2018].
- [91] Frank, E., Hall M. A., and Witten I.H., "The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", " Morgan Kaufmann, Fourth Edition, 2016.
- [92] ZeroR, "weka.classifiers.rules," n.d.. [Online]. Available: <http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/ZeroR.html>. [Accessed 3 2018].
- [93] RandomizableFilteredClassifier, "weka.classifiers.meta," n.d. [Online]. Available: <http://weka.sourceforge.net/doc.dev/weka/classifiers/meta/RandomizableFilteredClassifier.html>. [Accessed 3 2018].
- [94] RandomTree, "Weka classifiers trees," n.d. [Online]. [Accessed 14 3 2018].
- [95] Lu, H-P., Sun, Z-y., Qu, W-c, "Big Data-Driven Based Real-Time Traffic Flow State Identification and Prediction," *Discrete Dynamics in Nature and Society*, Vols. Volume 2015, Article ID 284906, pp. 1-11, 2015.
- [96] Villars, R. L., Olofson, C. W., Eastwood, M., "Big Data: What It Is and Why You Should Care," IDC, 2011.
- [97] Vlahogianni, E.I, Park, B.B., and van Lint, J.W.C., "Big data in transportation and traffic engineering," *Transportation Research Part C: Emerging Technologies*, vol. 58, no. Part B, pp. 1-161, 2015.
- [98] Stopher, P.R. and Greaves, S.P., "Household travel surveys: where are we going?," *Transport. Res. Part A: Policy Pract.*, vol. 41 , no. 5, p. 367–381, 2007.
- [99] Wang, X. and Li, Z., "Traffic and transportation smart with cloud computing on big data,"

- International Journal of Computer Science and Applications*,, vol. 13, no. 1, pp. 1-16, 2016.
- [100] Philander, K. and Zhong, Y., "Twitter sentiment analysis: Capturing sentiment from integrated resort tweets," *International Journal of Hospitality Management*, vol. 55, pp. 16-24, 2016.
- [101] Korkontzelos, I, Nikfarjam, A., Shardlow, M., Sarker, A., Ananiadou, S., Gonzalez, G.H, "Analysis of the effect of sentiment analysis on extracting adverse drug reactions from tweets and forum posts," *Journal of Biomedical Informatics*, vol. 62, pp. 148-158, 2016.
- [102] Burscher, B., Vliegthart, R. and de Vreese C.H., "Frames beyond Words: Applying Cluster and Sentiment Analysis to News Coverage of the Nuclear Power Issue," *Social Science Computer Review*, vol. 34, no. 5, pp. 530-545, 2016.
- [103] Abidin, A.F., Kolberg, M. and Hussain,A., "Integrating Twitter Traffic Information with Kalman Filter Models for Public Transportation Vehicle Arrival Time Prediction," in *Big-Data Analytics and Cloud Computing*, M. Trovati, R. Hill, A. Anjum, S. Y. Zhu and L. Liu, Eds., 2015, pp. 67-82.
- [104] Pak A. and Paroubek, P., "Twitter as a corpus for sentiment analysis and opinion mining," in *Proceedings of the Seventh conference of International language Resources and Evaluation(LREC'10)*, 2010.
- [105] Go. A., Huang, L. and Bhayani, R. , "Twitter sentiament analysis," in *Stanford University*, Stanford California, USA, CS224N - Final Year Project, 2009.
- [106] Wang, J., Gu, Q., and Wang,G., "Potentila power and problems in sentiment mining of social media," *International Journal of Strategic Decision Science*, vol. 4, no. 2, pp. 16-26, 2013.
- [107] Kumar,A. and Sebastian, T.M., "Sentiment analysis on Twitter," *International Journal of Computer Science Issues*, vol. 9, no. 4:3, pp. 372-378, 2012.
- [108] He, J., Shen, W., Divakaruni, P., Wynter, L. and Lawrence, R., "Improving Traffic Prediction with Tweet Semantics," in *Proceedings of the Twenty-Third International Joint Conference on Artificial*

Intelligence, Beijing, China, 2013.

- [109] Grosenick, S., "Real-Time Traffic Prediction Improvement through Semantic Mining of Social Networks," Unpublished Masters Thesis. University of Washington, Washington, 2012.
- [110] Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35-45, 1960.
- [111] Elsafoury, F.A., "Monitoring urban traffic status using twitter messages," pp. 1-46, 2013.
- [112] Azam, N., Abulaish, M., and Haldar, N. A-H, "Twitter data mining for events classification and analysis," in *Second International Conference on Soft Computing and Machine Intelligence*, 2015.
- [113] Broder, A. Z. and Glassman, S. C, Manasse, M. S.; Zweig, G, "Syntactic clustering of the web," *Computer Networks and ISDN Systems*, vol. 29, no. 8, p. 1157–1166, 1997.
- [114] van Dongen S., "Graph clustering by flow simulation," University of Utrecht, Utrecht, Netherlands, 2000.
- [115] McHugh, D., "Traffic Prediction and Analysis using a Big Data and Visualisation Approach," Blanchardstown, Ireland, 2014.
- [116] Tejaswin, P., Kumar, R., and Gupta, S., "Tweeting Traffic: Analyzing Twitter for generating real-time city traffic insights and predictions," in *CODS-IKDD '15* , Bangalore, India, 2015.
- [117] Rais, K, "Twitter analysis," 2014. [Online]. Available: <http://www.slideshare.net/ajayohri/twitter-analysis-by-kaify-rais>. [Accessed 28 January 2017].
- [118] Cran. R, "The Comprehensive R Archive Network," [Online]. Available: <https://cran.r-project.org/web/packages/>. [Accessed 03 04 2017].
- [119] Eckert K, "Simplified Phrase Search Algorithm," 2008.
- [120] Manning, C.D., Raghavan, P. and Schütze, H., "Introduction to information retrieval," Cambridge University Press, 2008.

- [121] Samworth, R. J, "Optimal weighted nearest neighbour classifiers," *Annals of Statistics*, vol. 40 , no. 5, p. 2733–2763, 2012.
- [122] Hu, M. and Liu, B., "Mining opinion features in customer review," in *Proceedings of the 19th National Conference on Artificial Intelligence, AAAI'04*, 2004.
- [123] Kang, H., Yoo, S.J. and Han, D., "Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis," *Expert Systems with Applications*, vol. 39, pp. 6000-6010, 2012.
- [124] Spärck Jones, K., "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11-21, 1972.
- [125] "RStudio," [Online]. Available: <https://www.rstudio.com/products/rstudio/>. [Accessed 23 April 2018].
- [126] Kurt, H., "R Foundation. "R FAQ". The Comprehensive R Archive Network. 2.13 What is the R Foundation?," 2015.
- [127] Lexalytics, "Text Analytics Software SaaS and On-premise," [Online]. Available: <https://www.lexalytics.com/>. [Accessed 20 May 2018].
- [128] Genetec blufaxcloud traffic engine, [Online]. Available: <http://genetec1.blufaxcloud.com/engine/getLinks.hdo>. [Accessed 02 2018].
- [129] Anand R.A, Vanajaskshi, L. and Subramanian, S., "Traffic density estimation under heterogenous traffic conditions using data fusion," *2011 IEEE Intelligent Symposium(IV)*, pp. 31-36, 2011.
- [130] Dailey, D.J., Harn, P., and Lin, P.J., "ITS data fusion. ITS Research Program, Final Research Report," 1996.
- [131] Bachmann, C., "Multi-sensor data fusion for traffic speed and travel time estimation," Toronto, 2011.
- [132] Bin W., Jian,W., Jianping, W., and Baigen, C., "Study on Adaptive GPS INS Integrated Navigation

- System.," in *IEEE Proc on Intelligent Transportation Systems*, 2003.
- [133] Chu, L. Oh, J. and Recker, W., "Adaptive Kalman Filter based freeway travel time estimation," in *Transportation Research Board 8th Annual Meeting*, Washington DC, 2005.
- [134] El Faouzi, N-E. and Klein, L.A., "Data fusion for ITS: techniques and research needs," *Transportation Research Procedia*, vol. 15, p. 495–512, 2016.
- [135] Wang, C., Zhu, Q., Shan, Z., Xia, Y., Liu, Y., "Fusing heterogeneous traffic data by kalman Filters and gaussian mixture models," in *2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*, Qingdao, China, 2014.
- [136] Angela-Aida K. Runyoro , and Jesuk, K., "Real-Time Road Traffic Management Using Floating Car Data," *International Journal of Fuzzy Logic and Intelligent Systems.*, vol. 13, no. 4, pp. 269-277, 2013.
- [137] Kim, S-S. and Kang, Y-B., "Congested avoidance algorithm using extended kalman filter," in *International Conference on Convergence Information Technology*, 2007.
- [138] Peng D., Zuo X, Wu J. Wang C., and Zhang T., "A Kalman Filter based iInformation fusion method," in *2nd International Conference on Power Electronics and Intelligent Transportation System*, 2009.
- [139] Lwin, H.T. and Naing, T.T., "Estimation of road traffic congestion using GPS data," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 12, pp. 1-5, 2015.
- [140] Necula, E., "Analyzing Traffic Patterns on Street Segments Based on GPS Data Using R," *Transportation Research Procedia*, vol. 10, pp. 276-285, 2015.
- [141] Necula, E., "Dynamic Traffic Flow Prediction Based on GPS Data," in *IEEE 26th International Conference on Tools with Artificial Intelligence*, Limassol, 2014.

- [142] Kaklij,S.P, "Mining GPS data for traffic congestion detection and prediction," *International Journal of Science and Research*, vol. 4, no. 9, pp. 876-880, 2015.
- [143] Ito, T., and Hiramoto, T., "A general simulator approach to ETC toll traffic congestion," *Journal of Intelligent Manufacturing*, vol. 17, no. 5, p. 597–607, 2006.
- [144] Kim,S. and Suh, W., "Modeling traffic congestion using simulation software," in *International Conference on Information Science and Applications (ICISA)*, 2014.
- [145] Metkari, M., Budhkar, A and Maurya, A.K., "Development of simulation model for heterogeneous traffic with no lane discipline," 2013.
- [146] He, S., "Analysis method of traffic congestion degree based on spatio-temporal simulation," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 4, 2012.
- [147] Nellore, K., Hancke, G.P., and Reindl, L.M., "A Survey on Urban Traffic Management System Using Wireless Sensor Networks," vol. 16, no. 2, 2016.
- [148] Aslam, J., Lim, S., and Rus,D., "Congestion-aware Traffic Routing System Using Sensor Data," in *2012 15th International IEEE Conference on Intelligent Transportation Systems Anchorage*, Alaska, USA, 2012.
- [149] Mai and Hranac, "Twitter Interactions as a Data Source for Transportation Incidents," *92nd Annual Meeting Transportation Research Board*.
- [150] Chen Po-Ta, Chen, F., and Qian, Z., "Road traffic congestion monitoring in social media with hinge-loss Markov random fields," *2014 IEEE International Conference on Data Mining*, pp. 80 - 89, 2014.
- [151] Hofleitner, A., Herring, R., Abbeel, P., and Bayen, A., "Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1-15, 2012.

- [152] Wang,X., Peng,L., Chi, T., Li, M., Yao, X. and Shao, J., "A hidden markov model for urban-scale traffic estimation using floating car data," *PLOS ONE*, 2015.
- [153] Elmenreich, W., "Sensor Fusion in Time-Triggered System," PhD Thesis, Vienna, Austria: Vienna , 2002.
- [154] Klein, L A., "Sensor and data fusion: A tool for information assessment and decision making," SPIE Press, 2004, p. 51.
- [155] Steinberg, A.N. and Bowman C.L., "Rethinking the JDL Data Fusion Levels".
- [156] Blasch, E., Plano, S, "Level 5: User Refinement to aid the Fusion Process," Proceedings of the SPIE, Vol. 5099., 2003.
- [157] Castanedo F., "A review of data fusion techniques," *The scientific world journal*, pp. 1-19, 2013.
- [158] JDL, Data Fusion Lexicon, "Technical Panel for C3, F.E. White, San Diego, California, USA,," 1991.
- [159] Anand, A., Ramadurai , G. and Vanajakshi, L., "Data Fusion-Based Traffic Density Estimation and Prediction," *Journal of Intelligent Transportation Systems*, vol. 18, no. 4, pp. 367-378, 2014.
- [160] Zhanquan, S. Mu, G., Wei, L., Jinqiao F. and Jiaying, H., "Multisource traffic data fusion with entropy based method," in *International Conference on Artificial Intelligence and Computational Intelligence*, 2009.
- [161] Antoniou, C., Ben-Akiva, M. and Koutsopoulos, H.N., "Kalman Filter Applications for Traffic Management," in *Kalman Filter, InTech*, 2010.
- [162] Crowley, J. L. and Demazeau, Y. , "Principles and Techniques for Sensor Data Fusion Signal Processing," *Signal Processing*, vol. 32, no. 1–2, p. 5–27, 1993.
- [163] Kuwahara, M. and Tanaka, S., "Urban Transport Data Fusion and Advanced Traffic Management for Sustainable Mobility," in *Spatial Data Infrastructure for Urban Regeneration*, vol. 5 , Institute

- of Industrial Science, The University of Tokyo, 4-6-1, Komaba,, 2008, pp. 75-102.
- [164] Ben-Akiva, M., Bierlaire, M., Koutsopoulos, H. N., and Mishalani, R., "Real-time simulation of traffic demand-supply interactions within DynaMIT," in *Transportation and network analysis: current trends*, vol. 63, M. a. M. P. Gendreau, Ed., Springer US, 2002, p. 19–36.
- [165] Durrant-Whyte, H.F. and Stevens, M., "Data fusion in decentralized sensing networks," in *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Canada, 2001.
- [166] Yang, J-S., "Travel time prediction using the GPS test vehicle and Kalman Filtering techniques," in *American Control Conference*, Portland, OR, USA, 2005. .
- [167] Guo, J., Xia, J and Smith, B.L., "Kalman filter approach to speed estimation using single loop detector measurements under congested conditions," *Journal of Transportation Engineering*, vol. 135, no. 12, pp. 927-934, 2009.
- [168] Pamula, T. and Krol, A., "The traffic flow prediction using bayesian and neural networks," *Intelligent Transportation Systems- Problems and Perspectives, Studies in Systems, Decision and Control*, vol. 32, pp. 105-126, 2016.
- [169] Zheng, W., Lee, D-H., M.ASCE, Shi, Q., "Short-term freeway traffic flow prediction: bayesian combined neural network approach," *Transportation Engineering*, pp. 114-121, 2006.
- [170] Awan, M.S.K. and Awais, M.M., "Predicting weather condition using fuzzy rule based system," *Applied soft computing*, vol. 11(2011), pp. 56-63, 2011.

Appendix A.1

Sample of the vehicle count data

Time	v(t-15)	v(t)	vhist(t)	vhist(t+15)	v(t+15) (DBN predict)	v(t+15) (RF Predict)	v(t+15) (NN-predict)
10:05	17	19	13	17	12	12	14
10:05	12	19	10	19	8	13	6
10:05	19	8	12	8	11	11	11
10:05	9	13	13	8	11	11	13
10:10	14	19	11	17	18	16	18
10:10	14	18	10	15	18	15	21
10:10	17	13	17	11	18	13	18
10:10	18	17	9	16	13	13	13
10:15	9	13	11	13	10	10	11
10:15	11	9	19	12	17	17	17
10:15	12	11	15	11	10	10	10
10:15	17	13	13	8	11	11	11
10:20	14	15	23	14	22	22	23
10:20	12	22	22	11	23	21	23
10:20	8	9	16	25	15	15	15
10:20	20	13	18	16	8	8	8
10:05	17	16	13	14	19	19	21
10:05	10	11	17	10	11	11	11
10:05	19	18	14	13	17	17	17
10:05	17	14	8	17	17	18	17
10:10	19	14	17	12	17	17	15
10:10	13	16	12	15	8	8	8
10:10	13	16	13	19	16	16	16
10:10	15	13	11	11	18	18	18
10:15	9	12	8	17	9	9	9
10:15	16	19	12	19	19	19	19
10:15	9	8	15	16	19	19	16
10:15	13	25	16	18	9	9	10
10:20	11	9	13	24	14	14	14
10:20	18	23	22	18	14	14	14
10:20	10	19	24	18	20	20	19
10:20	17	8	16	20	15	15	15
10:05	14	17	36	35	34	32	36

10:05	15	15	19	14	10	10	10
10:05	8	18	13	8	10	8	10
10:05	19	15	14	17	10	9	10
10:10	11	16	16	13	10	10	10
10:10	17	19	11	17	17	17	18
10:10	12	17	13	14	15	13	15
10:10	10	19	10	16	13	13	13
10:15	9	18	10	11	15	18	16
10:15	15	11	9	16	13	13	13
10:15	16	11	17	18	19	19	19
10:15	19	13	16	19	13	13	13
10:20	16	13	13	23	16	16	18
10:20	19	13	12	17	8	8	8
10:20	9	13	23	9	8	6	9
10:20	14	21	18	12	10	10	10
10:05	9	13	14	11	17	19	15
10:05	17	19	10	14	18	17	20
10:05	12	15	18	15	8	8	8
10:05	17	8	8	9	12	12	11
10:10	12	15	10	19	11	11	10
10:10	8	19	15	17	9	9	9
10:10	9	10	17	11	13	13	13
10:10	10	19	11	12	19	19	19
10:15	13	9	19	14	13	13	15
10:15	10	11	12	16	8	8	8
10:15	19	9	11	19	9	9	9
10:15	10	10	8	8	8	8	8
10:20	25	19	19	19	12	12	12
10:20	11	9	17	20	9	9	9
10:20	25	19	22	19	21	21	21
10:20	15	21	24	15	19	19	19
10:05	19	9	10	11	9	9	9
10:05	8	17	8	14	19	19	20
10:05	19	11	16	18	10	10	10
10:05	16	17	9	18	10	10	12
10:10	12	8	12	15	15	15	15
10:10	17	11	9	14	9	9	9
10:10	19	18	9	18	14	14	14
10:10	14	16	18	19	13	13	13
10:15	16	10	14	16	14	14	10

10:15	11	10	12	17	11	11	9
10:15	14	12	8	14	9	9	7
10:15	13	17	15	15	13	13	14
10:20	12	10	25	20	20	20	18
10:20	18	24	21	10	19	19	18
10:20	9	11	16	13	12	12	13
10:20	21	11	15	23	25	23	27
10:05	9	15	16	18	11	11	11
10:05	14	14	11	15	16	16	16
10:05	14	15	14	10	11	11	11
10:05	14	16	10	13	16	16	16
10:10	18	17	10	13	17	17	17
10:10	11	19	16	16	19	19	19
10:10	17	12	9	18	17	17	17
10:10	11	20	13	10	19	19	19
10:15	10	25	8	19	10	10	10
10:15	19	18	14	15	8	10	8
10:15	14	8	12	9	12	12	12
10:15	12	25	10	12	9	9	9
10:20	19	18	12	18	13	13	13
10:20	12	20	24	25	14	14	12
10:20	20	25	22	14	20	20	18
10:20	12	8	25	11	9	9	9
10:05	8	14	14	13	19	19	21
10:05	17	12	12	18	8	8	8
10:05	17	10	8	17	12	12	12
10:05	9	9	15	15	12	14	12
10:10	19	16	16	8	11	12	11
10:10	12	15	14	18	12	12	12
10:10	9	18	10	13	12	12	10
10:10	10	19	14	13	9	12	9
10:15	16	17	11	8	15	16	13
10:15	9	10	10	16	8	8	10
10:15	10	18	18	13	16	16	16
10:15	16	16	15	21	11	13	13
10:20	24	21	9	9	21	21	23

Appendix A.2

Source code (Spark) –NN for traffic congestion prediction

```
import org.apache.log4j.{Level, Logger}
import org.apache.spark.ml.classification.{MultilayerPerceptronClassifier}
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{VectorAssembler}
import org.apache.spark.sql.DataFrame

/**
 * Created by taiwoadetiloye on 2017-12-15.
 */
object montreal_traffic_NeuralNetwork extends App {

  Logger.getLogger("org").setLevel(Level.OFF)

  val dir = "./src/main/resources/"

  val path = dir + "Trafficdatamontreal.csv"

  val spark = org.apache.spark.sql.SparkSession.builder
    .master("local")
    .appName("Spark CSV Reader")
    .getOrCreate;

  import spark.implicits._

  val df = spark.read
    .format("csv")
    .option("header", "true") //reading the headers
    .option("mode", "DROPMALFORMED")
    .load(path)

  case class MontrealTraffic(Day:Double, Length: Double, Time: String,
vtminus15: Double, vt: Double, vhistt: Double, vhisttplus15:
Double,vtplus15predict: Double)

  val data = df.map { a =>
MontrealTraffic(a(0).toString.toDouble,a(1).toString.toDouble, a(2).toString,
a(3).toString.toDouble, a(4).toString.toDouble,
a(5).toString.toDouble,a(6).toString.toDouble, a(7).toString.toDouble) }
    .toDF("Day","Length","Time", "vtminus15", "vt", "vhistt", "vhisttplus15",
"vtplus15predict")

  trait DataProcessor {
    def mainDataProcessor():DataFrame
  }

  class trainDataProcessor(data:DataFrame ) extends DataProcessor {
```

```

    val features = Array("Day", "Length", "vtminus15",
"vt", "vhistt", "vhisttplus15")

    def mainDataProcessor =
    {
        val assembler = new
VectorAssembler().setInputCols(features).setOutputCol("featureVectors")

        assembler.transform(data)
    }

    val dataProcessed = new trainDataProcessor(data)

    val dataModel = dataProcessed.mainDataProcessor.select("featureVectors",
"vtplus15predict")

    dataModel.show(10)

    // Split the data into train and test
    val splits = dataModel.randomSplit(Array(0.7, 0.3), seed = 1234L)
    val train = splits(0)
    val test = splits(1)

train.printSchema()

    val count = data.select("vtplus15predict").distinct().count()
    print("count: " + count)
    val layers = Array[Int](6, 7, 6, 35) //vary layers accordingly for NN -
single layer to multilayer

    // create the trainer and set its parameters
    val mlp = new MultilayerPerceptronClassifier()
        .setLayers(layers)
        .setFeaturesCol("featureVectors")
        .setLabelCol("vtplus15predict")
        .setMaxIter(1500)

    // train the model
    val model = mlp .fit(train)

    // compute accuracy on the test set
    val predictions = model.transform(test)
    val predictionAndLabels = predictions.select("prediction",
"vtplus15predict")

    // Select (prediction, true label) and compute test error.
    val evaluator = new MulticlassClassificationEvaluator()
        .setLabelCol("vtplus15predict")
        .setPredictionCol("prediction")

```

```

val evaluator1 = evaluator.setMetricName("accuracy")
val evaluator2 = evaluator.setMetricName("weightedPrecision")
val evaluator3 = evaluator.setMetricName("weightedRecall")
val evaluator4 = evaluator.setMetricName("f1")

val accuracy = evaluator1.evaluate(predictions)
val precision = evaluator2.evaluate(predictions)
val recall = evaluator3.evaluate(predictions)
val f1 = evaluator4.evaluate(predictions)

println("Accuracy = " + accuracy)
println("Precision = " + precision)
println("Recall = " + recall)
println("F1 = " + f1)
println(s"Test Error = ${1 - accuracy}")

val multiClassEvaluator = new MulticlassClassificationEvaluator()
val auroc =
multiClassEvaluator.setLabelCol("vtplus15predict").setPredictionCol("prediction")
multiClassEvaluator.evaluate(predictions)
println(s"Area under ROC = $auroc")

spark.stop()

```

Appendix A.3

Source code (Spark) –RF for traffic congestion prediction

```
import org.apache.log4j.{Level, Logger}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.classification.{MultilayerPerceptronClassifier,
RandomForestClassificationModel, RandomForestClassifier}
import org.apache.spark.ml.evaluation.{BinaryClassificationEvaluator,
MulticlassClassificationEvaluator}
import org.apache.spark.ml.feature.{RFormula, SQLTransformer, StringIndexer,
VectorAssembler}
import org.apache.spark.sql.DataFrame

/**
 * Created by taiwoadetiloye on 2017-12-08.
 */
object montreal_traffic_RandomForest extends App {

  Logger.getLogger("org").setLevel(Level.OFF)

  val dir = "./src/main/resources/"

  val path = dir + "Trafficdatamontreal.csv"

  val spark = org.apache.spark.sql.SparkSession.builder
    .master("local")
    .appName("Spark CSV Reader")
    .getOrCreate;

  import spark.implicits._

  val df = spark.read
    .format("csv")
    .option("header", "true") //reading the headers
    .option("mode", "DROPMALFORMED")
    .load(path).cache()

  case class MontrealTraffic(Day:Double, Length: Double, Time: String,
vtminus15: Double, vt: Double, vhistt: Double, vhisttplus15:
Double,vtplus15predict: Double)

  val data = df.map { a =>
MontrealTraffic(a(0).toString.toDouble,a(1).toString.toDouble, a(2).toString,
a(3).toString.toDouble, a(4).toString.toDouble,
a(5).toString.toDouble,a(6).toString.toDouble, a(7).toString.toDouble) }
    .toDF("Day","Length","Time", "vtminus15", "vt", "vhistt", "vhisttplus15",
"vtplus15predict")
```

```

//data.printSchema()
//data.show(10)

/*
 * This function helps to select Indexed columns obtained from the
stringIndex function using SQLTransformer
 */

trait DataProcessor {
  def mainDataProcessor():DataFrame
}

/* RFormula produces a vector column of features and a double or string
column of label.
Like when formulas are used in R for linear regression, string input
columns will be one-hot encoded,
and numeric columns will be cast to doubles. If the label column is of
type string,
it will be first transformed to double with StringIndexer.
If the label column does not exist in the DataFrame, If the label column
does not exist in the DataFrame,

 */

class trainDataProcessor(data:DataFrame ) extends DataProcessor {

  val features = Array("Day", "Length", "vtminus15",
"vt", "vhistt", "vhisttplus15")

  def mainDataProcessor =
  {
    val assembler = new
VectorAssembler().setInputCols(features).setOutputCol("featureVectors")

    assembler.transform(data)
  }
}

val dataProcessed = new trainDataProcessor(data)

val dataModel = dataProcessed.mainDataProcessor.select("featureVectors",
"vtplus15predict")

// Split the data into train and test
val splits = dataModel.randomSplit(Array(0.7, 0.3), seed = 1234L)
val train = splits(0)
val test = splits(1)

// create the trainer and set its parameters
val rf = new RandomForestClassifier()
  .setFeaturesCol("featureVectors")
  .setLabelCol("vtplus15predict")
  .setNumTrees(10)
//   .setSeed(1344L)

```

```

//      .setMaxDepth(8).setMaxBins(100)

val pipeline = new Pipeline()
    .setStages(Array( rf))

// train the model
val model = pipeline.fit(train)

// compute accuracy on the test set
val predictions = model.transform(test)
val predictionAndLabels = predictions.select("prediction",
"vtplus15predict").show(10)

// Select (prediction, true label) and compute test error.
// Select (prediction, true label) and compute test error.
val evaluator = new MulticlassClassificationEvaluator()
    .setLabelCol("vtplus15predict")
    .setPredictionCol("prediction")

val evaluator1 = evaluator.setMetricName("accuracy")
val evaluator2 = evaluator.setMetricName("weightedPrecision")
val evaluator3 = evaluator.setMetricName("weightedRecall")
val evaluator4 = evaluator.setMetricName("f1")

val accuracy = evaluator1.evaluate(predictions)
val precision = evaluator2.evaluate(predictions)
val recall = evaluator3.evaluate(predictions)
val f1 = evaluator4.evaluate(predictions)

println("Accuracy = " + accuracy)
println("Precision = " + precision)
println("Recall = " + recall)
println("F1 = " + f1)
println(s"Test Error = ${1 - accuracy}")

val rfModel = model.stages(0).asInstanceOf[RandomForestClassificationModel]
//println("Learned classification forest model:\n" + rfModel.toDebugString)

val multiClassEvaluator = new MulticlassClassificationEvaluator()
val auROC =
multiClassEvaluator.setLabelCol("vtplus15predict").setPredictionCol("predicti
on").evaluate(predictions)
println(s"Area under ROC = $auROC")

spark.stop()

```

Appendix B.1

Agent-based modeling of traffic congestion




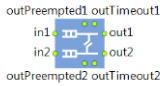

We present an agent-based model for congested traffic flow. AnyLogic² is used to develop the discrete event simulation model. The simulation experiments showed some main cases of using real-time analysis of vehicle agents cruising on a single lane traffic. The following three scenarios have been modeled separately in order to test the functionalities of the model: the onset of rush hour, the presence of traffic signalization and the passage of an emergency vehicle. Also, we present the steps of evaluating our model with regards to the vehicle-agents. These include the visualization as well as statistical analysis of the traffic flow and time delay. To create the microscopic traffic environment, the geolocation based on the latitude, longitude and scale are obtained using OpenStreetMap. Our main objective is to improve the quality of monitoring traffic situations with a view to reducing the traffic jams in the real-world and understanding the heterogeneous behaviors of traffic flow congestion. To simulate the congested traffic situations using AnyLogic, we made relevant selections from its process modeling library and then employed the selected process blocks to build the traffic agents structure as described in the following sections.

i. Process modeling library

The process modeling library supports discrete-event framework for various real-work systems that are agent specific like vehicle mobility in traffic streams. Its internal mechanisms cater for generating, disposing of, delaying, queueing and moving agents etc., to set the parameters and to perform the relevant statistical analysis. Table A.1 illustrates the important traffic agent buildings blocks.

² AnyLogic is a multimethod simulation modeling tool developed by The AnyLogic Company. It supports agent-based, discrete event, and system dynamics simulation methodologies.

Table A.1. Traffic agent block from AnyLogic Process modeling library. Adapted from AnyLogic

Agent block	Diagrammatic representation	Description
source		This is usually the starting point of a process model which generates agents from an output. Criteria can be set to define when and how many agents should be generated as well as setting parameters for the arrival rate, schedule of quantities etc.
sink		It disposes agents at the end point in a process model. The agents must be properly exited via the sinks while satisfying these conditions: <ul style="list-style-type: none"> ▪ The agent must have been unregistered from a network if it was in a network ▪ The agent should not possess any resource units or network resource units If the agent contains other agents, they all should satisfy the same disposal conditions ▪ If any of these conditions is violated, sink raises an error.
delay		Delays slow down an agent for a period of time. The delay of a traffic agent could be done dynamically or stochastically and can also depend on other factors. Introducing delay can be useful towards understanding the congestion pattern on a particular route.
match		It synchronizes two streams of agents such that once a new agent arrives at either of the input ports it is checked for a match against all agents in the queue for the other stream. The default match is to find if the match can be found so that both agents exit the Match object at the same time. Other criteria can be to have the queues fully customized based on timeout, priorities, pre-emption, etc. The agents that have not yet been matched are stored in two queues (one for each stream).
moveTo		This block allows the agent to move to a new location. The moving of the agent in traffic can be in a map system with specification of the latitude and longitude for the start and end points with agents moving along the routes.

ii. Traffic agents structure

To create the traffic agents structure, we made use of the selected blocks in a way that intuitively connects them to each other and defines the properties of the agent assigned to the structure. Manual input parameter settings are used. We define three agent-based managers

(please see Appendix B.2) to represent the main structure of the congested traffic model as follows:

- *Transit manager*

The transit manager is designed to connect the ‘source’ block to the ‘sink’ block via the ‘move-To’ block. This manager is fundamental to the movement of mobile agents in traffic. It is suitable for fast moving vehicle agents like an ambulance. In addition, criteria such as the arrival rate, trip time and quantity of agents arriving at the destination can be set.

- *Delay manager*

In modeling the delay manager, we introduce the ‘delay’ block between the ‘source’ and ‘moveTo’ block in the transit manager. We set the time delay for the ‘delay’ block parameter in order to slow down the speed of the vehicle agents. This creates the effect of medium to high congestion on the traffic streams in the AnyLogic simulation environment.

- *Match manager*

The match manager allows two streams of vehicle agents to have common features with queues implementation based on timeout, and priorities. Hence, it can be that once a new agent arrives at either of the input ports it is checked for a match against all agents in the queue for the other stream. This enables compatible agents to exit the match block at the same time as applicable to obeying the stops. This can be likened to common vehicle behavior at crossroads in the real-world.

iii. Single-lane system

In a single-lane system consisting of the driver-vehicles agents, we can assume that the mobile agents are basically vehicles: commercial buses and private cars; and they conform to the car-following within safe distance pattern. Their historic review into the car-following model provide key insights into the widely known Gazis-Herman-Rothery(GHR) model. The fundamental concepts, with its various contributions and developments over the past years, are important in our modelling and simulating of traffic congestion vis-à-vis the heterogeneous driver' behaviors and vehicle trajectory.

However, in order not to digress from our main objective of traffic congestion simulation based on the heterogeneous driver' behaviors without emphasis on the vehicle trajectory, we provide brief discussion of the car following GHR model.

The GHR model formulation is defined as:

$$a_n(t) = C_v^m(t) \frac{\Delta v(t-T)}{\Delta x^1(t-T)} \quad (1)$$

Where

a_n - the acceleration of vehicle n implemented at time t by a driver and is proportional to v the speed of the n^{th} vehicle

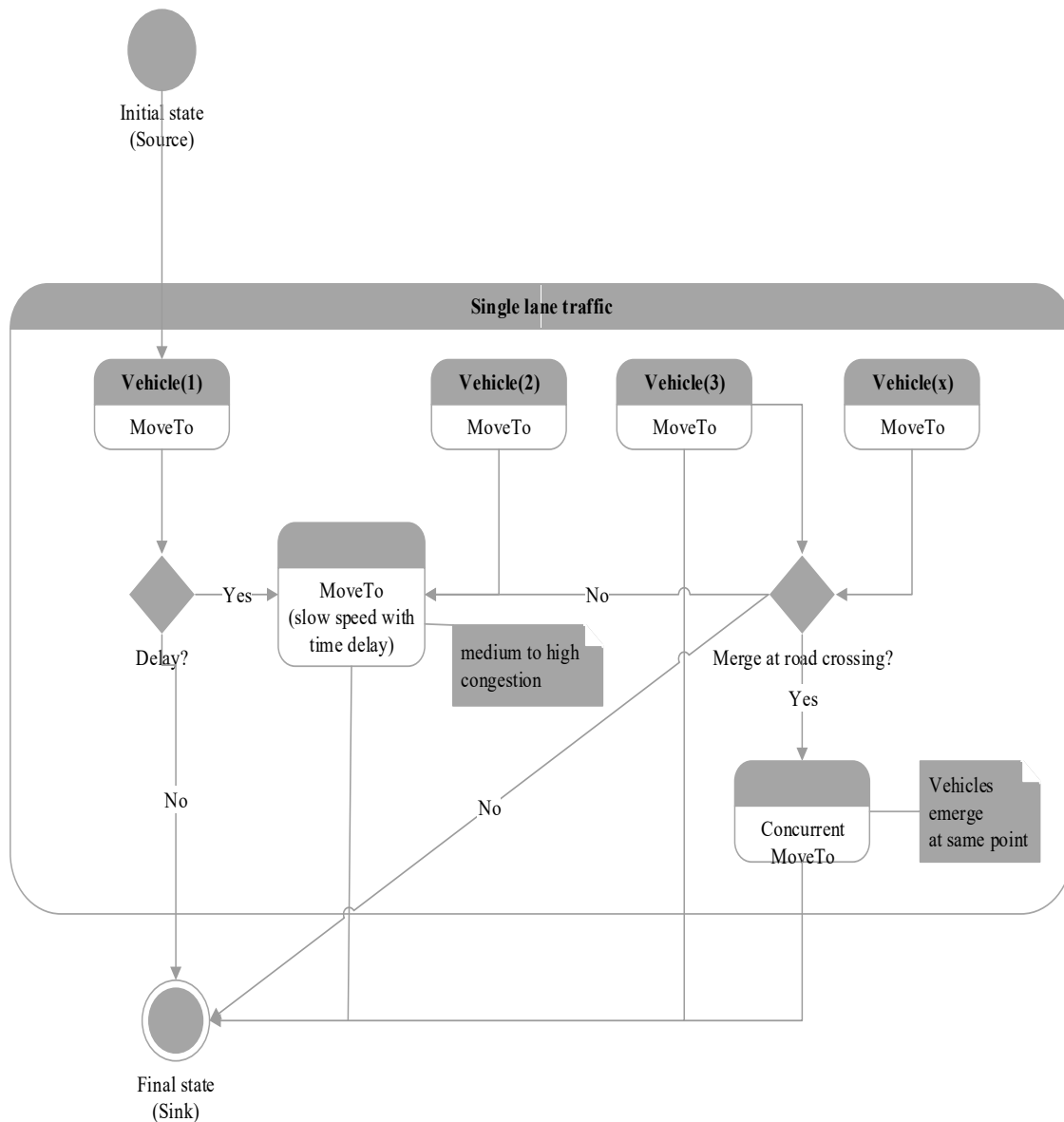
Δx and Δv - relative spacing and speeds, respectively between the n^{th} and $n - 1$ vehicle (that is the vehicle immediately in front), assessed at an earlier time $t - T$, where T is the driver reaction time

m , l and c are constants to be determined

To simulate our congested traffic streams in addition to the cognitive instance of the entities behaviors and interactions with respect to constraint measures, the parameters can have

the initial vehicle speed at the origin as 10m/s; and the highway speed within the interval [40, 80] m/s with expected compliance of the intelligent autonomous vehicle agents with the aforementioned GHR car-following model. As illustrated in the traffic state diagram in Figure A.1, heterogeneous behavior is undertaken by the vehicle agents as they move from source to sink nodes. They are able to make some logical decisions due to signal control at road junctions, obstruction or fast moving emergency vehicles. While, the vehicle agents constitute the main entity, others such as the traffic signal light, road networks and intersection can be segmented into secondary entities.

Also, we used reproducible random agent generation with first-in-first-out, FIFO, for the traffic events during simulation. To minimize memory usage the iteration is configured to stop in 100seconds of CPU runtime while having the maximum memory at 256Mb. It is noteworthy that in AnyLogic, cartesian map can be developed using latitude and longitude degree coordinates that can be scaled to desirable view. In this model, the scale is 1:5000. Additionally, the respective routes for the driver-vehicle agents were configured using the source to sink nodes. Other pertinent configuration details include pulling the map resource from OpenStreetMap (OSM) remote server via request message by AnyLogic routing server utilizing fastest routing method (See Figure A.2). Our simulation examples can be found online at <https://goo.gl/ZGYG10>



Input:
 Delay time(secs)
 Vehicle agents
 Average traffic flow(N_0 of vehicle agents /secs)

Output
 Congestion type:
 low, medium or high congestion

Figure A.1: Traffic state diagram

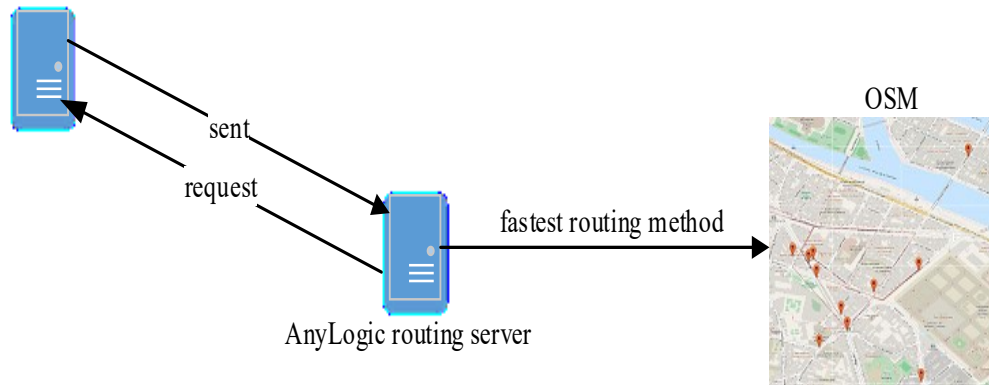


Figure A.2: Cartesian map configured based on OSM servers, and Anylogic routing servers employing fastest routing method and integrated with source and sink nodes

In order to determine behaviors of simulated vehicle sequence under low or congestion-free, and medium to high congested traffic flow, we statistically monitor the number of vehicles for insight into the travel actions of vehicle agents that may constitute traffic jam and how this actions can be related to the time delay and slow speed during traffic flow on single lanes traffic system. In the following test cases, we report the functionalities of our agent-based model:

1. *the start of rush hour when vehicles are moving in or out of the city center*
2. *when the effect of traffic signal controls bring about junction delays at road intersections*
3. *when an emergency vehicle-agent like ambulance or police cars have to navigate traffic causing others to make way in a maneuvering or pause state for the fast moving vehicles.*

In each of the three cases, the simultaneous movement of vehicles is modeled. This is a simplified model of a real-life situation with many more routes available to vehicles in the same city area. We focused mainly on three lanes: lane1, lane2 and lane3; respectively. It is assumed that each lane can have fast-moving agent vehicles operating for emergency response; and that

the vehicles start moving along lane2 and lane3 at the same time, and after a certain delay along the lane1. In this model, we take the unit of traffic flow to be the number of vehicles on a traffic lane per seconds. The time delay is the time difference between actual travel time and free-flow travel time (unit in seconds). Hence, the model tracks the traffic flow and time delay for each of the three lanes. Note that our measurements are in simulation settings. We consider the following cases:

- *Case 1: onset of rush hour*

Figure A.3 depicts the traffic flow at the onset of a rush hour. The onset of a rush hour is modeled by tracking the build-up of traffic on the lanes in order to determine the traffic trend and the level of congestion: free, low, medium or high. The amount of initial time delay of lane1 is approximately 30 seconds while there are no initial time delay on lane2 and lane3. We performed the simulation for a period of 60 secs to tally with start of rush hour.

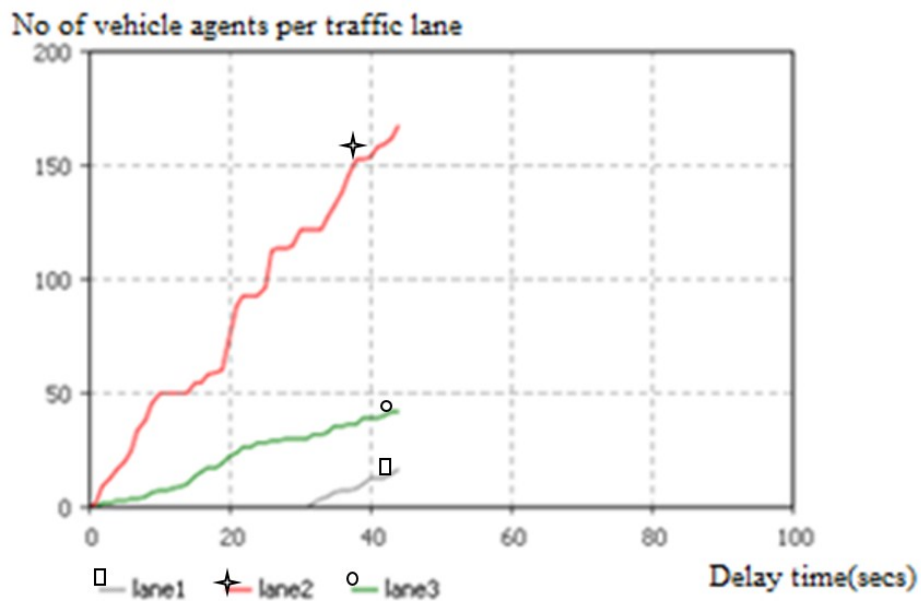


Figure A.3: Measure of traffic flow (at the onset of rush hour)

- *Case 2: effect of traffic signal controls*

In reality, traffic controls always exist in cities and can serve to ease congestion especially in road intersections. The effect of traffic signal controls is modeled separately from the other specific effects. Figure A.4 depicts the traffic flow with the presence of traffic signal controls.

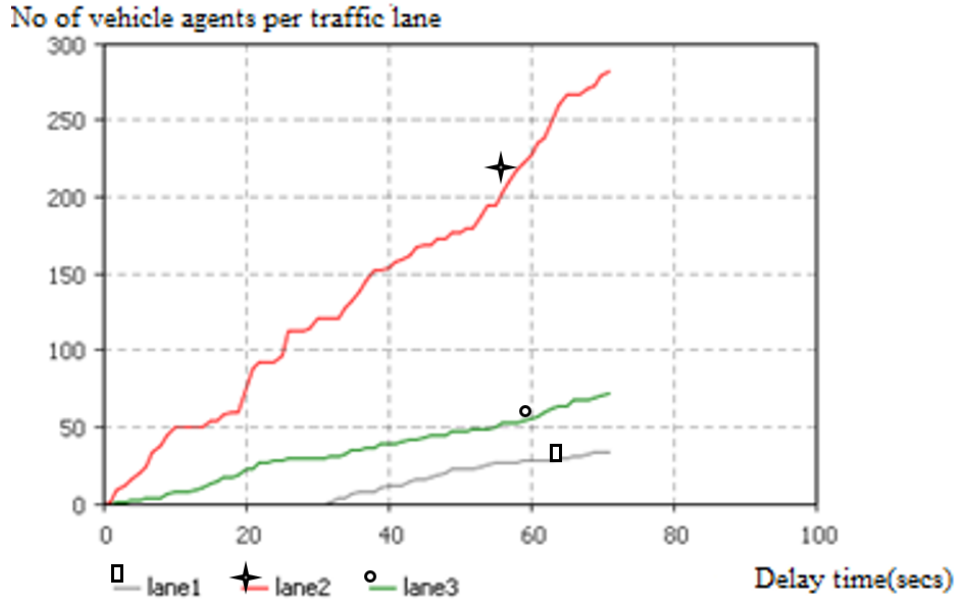


Figure A.4: Measure of traffic flow with the presence of traffic signal controls.

Traffic signal agent is assigned to each intersection points while discretizing the queue length in the past time frame. We observe that the vehicle agents are in a steep increase of speed with less delay under traffic light signal. Newton-Raphson method is used to iterate over the state space of the approaching agents towards the intersecting links for initial guess of priority queue, y_{k+1} :

$$g(y_{k+1}) := y_{k+1} - y_k - hf(x_{k+1}, y_{k+1}) = 0 \quad (3.3)$$

where f is a function of the amount of traffic flow with respect to the time delay, x_{k+1} represents the queue length, and h is the step-size.

- *Case 3: presence of emergency vehicle-agents*

In emergency traffic congestion, a vehicle such as an ambulance, a fire truck or a police vehicle travels in an emergency mode with high speed and maneuvering to avoid a collision as it moves to its destination. Other vehicles that find themselves along the path of the emergency vehicle and ahead of it will attempt to get out of the way and stop in the single-lane traffic. This considers other plausible situations of rush hours considering that emergency agent vehicle can ignore traffic signal control. Of course, it can be assumed that the emergency vehicle agents can travel in more than one traffic lanes in order to reach their destination in the fastest time. The delay is expected to increase with the number of vehicle agents on the roadway. For the same lane and the same level of traffic flow, the emergency vehicles are likely to cause the greatest delay in traffic.

In Figure A.5, it can be observed that the line plot of lane2 is the least smooth, due to the most pronounced instances of average delay times corresponding to the number of vehicle agents per traffic lane in the presence of emergency vehicle-agents.

Modeling traffic flows is key to understanding real world traffic jams and making improvement based on the analysis of heterogeneous traffic congestion. Our traffic simulation setup using AnyLogic cover three main cases of using real-time analysis of vehicle agents moving on single lane traffic during rush hour, under the effect of signal controls at intersections and in the presence of fast moving and emergency vehicle agents. We measured the heterogeneous traffic congestion on three single lane traffic systems. The congestion scenarios

are monitored with regards to the congestion type, delay time, number of vehicle agents and the average traffic flow. We observe that fast moving emergency vehicles are most likely to create congestion in a busy single-lane traffic due to their urgent cruise operations with high maneuvering and acceleration.

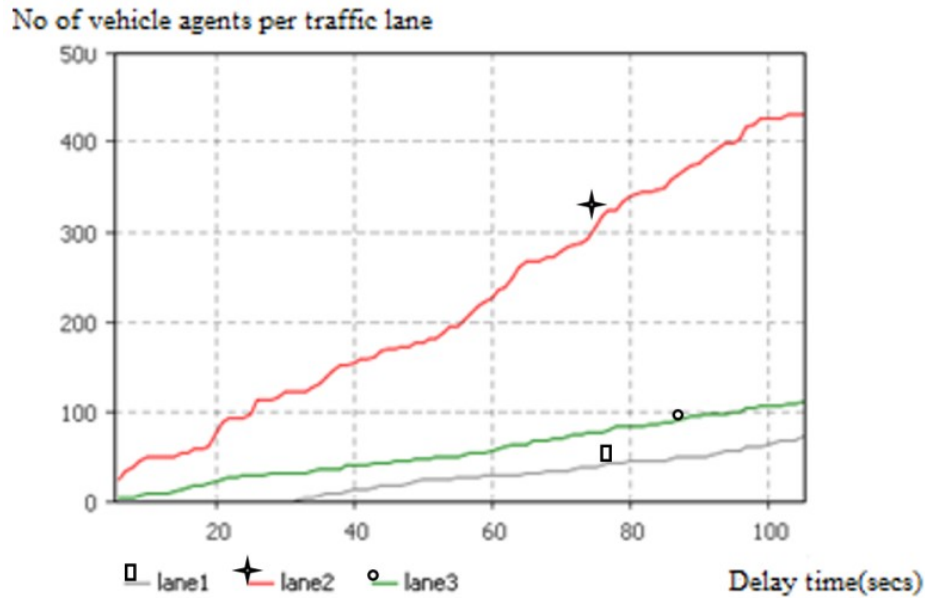
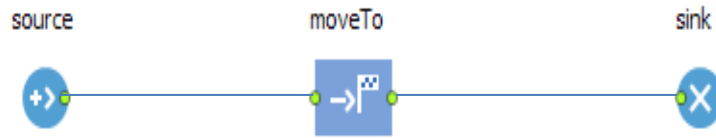


Figure A.5: Measure of traffic flow (with emergency vehicle agents in congested traffic)

In future work, integration of the developed technique with machine learning should be investigated. The model should be extended to multi-lane traffic. Also, it should be integrated with a system of sensors providing near real-time data. Such a model when validated, would assist drivers navigate traffic congestion and in the self-driving vehicles to choose optimal routes.

Appendix B.2

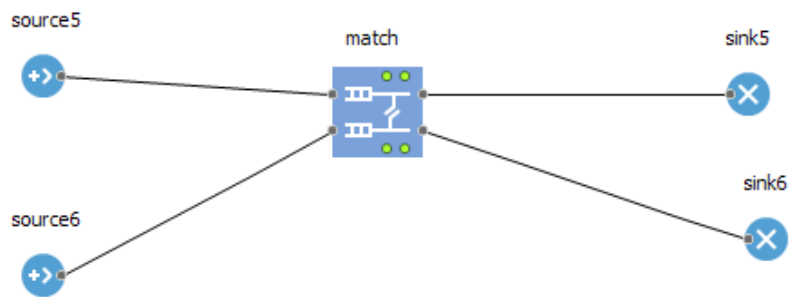
Diagram of traffic state manager



Transit manager



Delay manager



Match manager