

A framework for smart traffic management using heterogeneous data sources

Item Type	Thesis or dissertation
Authors	Jones, Angelica Salas
Publisher	University of Wolverhampton
Rights	Attribution-NonCommercial-NoDerivatives 4.0 International
Download date	2026-06-16 21:16:40
License	http://creativecommons.org/licenses/by-nc-nd/4.0/
Link to Item	http://hdl.handle.net/2436/623343

**A FRAMEWORK FOR SMART TRAFFIC MANAGEMENT USING
HETEROGENEOUS DATA SOURCES**

Angelica Milagros Salas Jones

BEng, MSc, PGCert

A thesis submitted in partial fulfilment of the requirements of the University of
Wolverhampton for the degree of Doctor of Philosophy (PhD)

March 2020

This work or any part thereof has not previously been presented in any form to the University or to any other body whether for the purposes of assessment, publication or for any other purpose (unless otherwise indicated). Save for any express acknowledgements, references and/or bibliographies cited in the work, I can confirm that the intellectual content of the work is the result of my own efforts and of no other person.

The right of Angelica Milagros Salas Jones to be identified as author of this work is asserted in accordance with ss. 77 and 78 of the Copyright, Design and Patents Act 1988. At this date, the author owns copyright.

Signature.....

Date.....

Abstract

Traffic congestion constitutes a social, economic and environmental issue to modern cities as it can negatively impact travel times, fuel consumption and carbon emissions. Traffic forecasting and incident detection systems are fundamental areas of Intelligent Transportation Systems (ITS) that have been widely researched in the last decade. These systems provide real time information about traffic congestion and other unexpected incidents that can support traffic management agencies to activate strategies and notify users accordingly. However, existing techniques suffer from high false alarm rate and incorrect traffic measurements. In recent years, there has been an increasing interest in integrating different types of data sources to achieve higher precision in traffic forecasting and incident detection techniques. In fact, a considerable amount of literature has grown around the influence of integrating data from heterogeneous data sources into existing traffic management systems.

This thesis presents a Smart Traffic Management framework for future cities. The proposed framework fusions different data sources and technologies to improve traffic prediction and incident detection systems. It is composed of two components: social media and simulator component. The social media component consists of a text classification algorithm to identify traffic related tweets. These traffic messages are then geolocated using Natural Language Processing (NLP) techniques. Finally, with the purpose of further analysing user emotions within the tweet, stress and relaxation strength detection is performed. The proposed text classification algorithm outperformed similar studies in the literature and demonstrated to be more accurate than other machine learning algorithms in the same dataset. Results from the stress and relaxation analysis detected a significant amount of stress in 40% of the tweets, while the other portion did not show any emotions associated with them. This information can potentially be used for policy making in transportation, to understand the users' perception of the transportation network. The simulator component proposes an optimisation procedure for determining missing roundabouts and urban roads flow distribution using constrained optimisation. Existing imputation methodologies have been developed on straight section of highways and their applicability for more complex networks have not been validated. This task presented a solution for the unavailability of roadway sensors in specific parts of the network and was able to successfully predict the missing values with very low percentage error. The proposed imputation methodology can serve as an aid for existing traffic forecasting and incident detection methodologies, as well as for the development of more realistic simulation networks.

Table of Contents

Abstract.....	i
Table of Contents.....	ii
List of tables.....	vii
List of figures.....	ix
List of abbreviations.....	xii
Dedication.....	xiv
Acknowledgments.....	xv
List of publications.....	xvi
Chapter 1: Introduction.....	1
1.1 Background and motivation.....	1
1.2 Research questions.....	5
1.3 Aim and objectives.....	5
1.4 Research methodology.....	7
1.4.1 Phase 1: Problem identification and objectives of a solution.....	7
1.4.2 Phase 2: Design and development.....	8
1.4.3 Phase 3: Demonstration.....	8
1.4.4 Phase 4: Evaluation.....	8
1.4.5 Phase 5: Communication.....	9
1.5 Research gaps and contribution to knowledge.....	9
1.6 Thesis outline.....	10
Chapter 2: Overview of Traffic Management Systems.....	13
2.1 Introduction.....	13
2.2 Traffic variables, sensors and algorithms.....	14
2.2.1 Traffic variables.....	14

2.2.2	Sensor Technologies	17
2.2.3	Algorithms	24
2.3	Survey of Traffic Management Systems	28
2.3.1	Incident Detection Systems	28
2.3.2	Traffic forecasting systems	31
2.3.3	Traffic flow modelling	34
2.4	Imputation methodologies for missing traffic data.....	37
2.5	Transportation for smart cities	40
2.5.1	The smart city concept.....	40
2.5.2	Smart mobility applications.....	42
2.5.3	Challenges and future perspectives	47
2.6	Research gaps	50
2.7	Summary.....	51
Chapter 3:	Social media for transportation	53
3.1	Introduction.....	53
3.2	Twitter as a sensor	54
3.2.1	Characterisation of Twitter	55
3.2.2	Accessing Twitter data	57
3.2.3	Twitter event detection methodologies	60
3.2.4	Detecting sentiment from Twitter streams	66
3.3	Twitter based transportation research	68
3.4	Challenges of using Twitter data for traffic event detection.....	72
3.5	Research gaps	75
3.6	Summary.....	75
Chapter 4:	Smart traffic management framework	77
4.1	Introduction.....	77

4.2	Research methodology	77
4.2.1	Design science research	77
4.2.2	Systems design methodologies	81
4.2.3	The study research design.....	84
4.3	Smart traffic management framework	85
4.3.1	Social media component.....	88
4.3.2	Simulator component	91
4.4	Testing and implementing the components.....	94
4.5	Summary.....	96
Chapter 5: Social media processing system		98
5.1	Introduction.....	98
5.2	System architecture	99
5.2.1	Twitter data acquisition	100
5.2.2	Pre-processing	103
5.2.3	Classification.....	105
5.2.4	Geolocation	109
5.2.5	Sentiment and stress analysis	111
5.3	Description of the datasets	113
5.4	Results.....	114
5.4.1	Evaluation metrics.....	114
5.4.2	Classification task.....	115
5.4.3	Geolocation	118
5.4.4	Sentiment and stress analysis	120
5.5	Discussion	122
5.6	Summary.....	126
Chapter 6: Real-time social mining for TMS applications.....		128

6.1	Introduction.....	128
6.2	Real-time processing system.....	128
6.3	Validation	138
6.4	Summary.....	142
Chapter 7: Optimisation-based traffic data imputation		144
7.1	Introduction.....	144
7.2	Model description.....	145
7.3	Preparing the network.....	147
7.3.1	Placing detectors.....	147
7.3.2	Importing traffic data and running simulations.....	149
7.4	Optimisation-based traffic data imputation	151
7.4.1	Optimisation task.....	154
7.4.2	Validation task	156
7.4.3	Experiments	157
7.5	Discussion	169
7.6	Summary.....	171
Chapter 8: Data imputation for real-time TMS applications.....		173
8.1	Introduction.....	173
8.2	Real-time imputation of missing flow in complex networks.....	173
8.3	Modelling the impact of incidents	181
8.4	Summary.....	184
Chapter 9: Conclusions and contributions to knowledge		185
9.1	Introduction.....	185
9.2	Conclusions	185
9.3	Main original contributions.....	190
9.4	Extended original contributions.....	192

9.5 Recommendations for future research.....	196
References.....	198
Appendices	217
Appendix 5-A.....	217
Appendix 5-B.....	219
Appendix 5-C.....	220
Appendix 6-A.....	224
Appendix 7-A.....	226
Appendix 7-B.....	228
Appendix 7-C.....	231
Appendix 8-A.....	235

List of tables

Table 1-1: Research gap and contribution to knowledge of the research.....	9
Table 2-1: Advantages and disadvantages of roadway-based sensors.....	21
Table 3-1: Streaming API categories.....	58
Table 3-2: Different filtering parameters of the Streaming API	59
Table 3-3: Twitter API keyword tracking examples	59
Table 3-4: Twitter API location tracking example.....	59
Table 3-5: Summary of traffic incident detection techniques using Twitter data.....	72
Table 4-1 Design Science Research guidelines.....	78
Table 4-2: Strengths and weaknesses of simulation packages	93
Table 5-1: Main attributes of a tweet used for the social media component	103
Table 5-2: Example of Geolocation stage	111
Table 5-3: Sentiment and stress analysis using SentiStrength and TensiStrength ..	112
Table 5-4: Example of labelled tweets	113
Table 5-5: Confusion matrix of a binary classification problem.....	115
Table 5-6: Evaluation metrics.....	115
Table 5-7: Classifiers accuracy vs Word n-gram features.....	116
Table 5-8: Classifiers evaluation metrics on the test dataset.....	117
Table 5-9: Training and prediction time (sec)	118
Table 5-10: NER model accuracy	119
Table 5-11: Performance of Geolocation engines.....	120
Table 5-12: Frequency of positive sentiment results	121
Table 5-13: Frequency of negative sentiment results	121

Table 5-14: Measures of central tendency for SentiStrength.....	121
Table 5-15: Frequency of relaxation results.....	122
Table 5-16: Frequency of Stress results	122
Table 5-17: Measures of central tendency for TensiStrength	122
Table 5-18: Comparing results with other studies in the literature	123
Table 6-1: Confusion matrix of the classification task.....	131
Table 6-2: Performance measure from real-time implementation.....	131
Table 6-3: Interpretation of Kappa	133
Table 6-4: Frequency table for positive emotions	135
Table 6-5: Frequency table for negative emotions	135
Table 6-6: Measures of central tendency for TensiStrength	135
Table 6-7: Validation of traffic events detected from Twitter	140
Table 6-8: Users' perception of the transport network	141
Table 7-1: Features of the modelled network	145
Table 7-2: Initial values for optimisation	163
Table 7-3: Results from optimisation routine	164
Table 7-4: MAPE of simulated results.....	164
Table 7-5: Initial values for optimisation	168
Table 7-6: Results from optimisation routine	168
Table 7-7: MAPE of simulated results.....	169
Table 8-1: Results from the validation of the imputation methodology.....	179
Table 8-2: Comparison of periods with low flow and normal flow distribution.....	180
Table 8-3: M6 performance indicators (percentage).....	183
Table 8-4: M40 performance indicators (percentage).....	183

List of figures

Figure 1-1: Top ten European capitals with worst traffic jams	1
Figure 1-2: Causes of Highway congestion.....	2
Figure 1-3: Design Science Research methodology process.....	7
Figure 1-4: Thesis layout	12
Figure 2-1: TMS activities	14
Figure 2-2 Example of roadway-based sensors.....	18
Figure 2-3 Example of Probe Vehicle data service.....	23
Figure 2-4: Smart City wheel.....	42
Figure 3-1: Global social media applications in 2018, ranked by number of users (millions)	53
Figure 3-2: Example of a Twitter timeline	56
Figure 3-3: Structure of a tweet in JSON format.....	60
Figure 3-4: Generic event detection methodology from Twitter streams.....	61
Figure 4-1: Level of maturity of the different DSR outputs	80
Figure 4-2: DSR Knowledge contribution framework	81
Figure 4-3: Systems Design Lifecycle	83
Figure 4-4: Existing TMS phases.....	86
Figure 4-5: Smart traffic management framework	88
Figure 4-6: Social media data processing system.....	89
Figure 5-1: System architecture of the social media component	100
Figure 5-2: Map view of the location bounding box.....	102
Figure 5-3: Tweet pre-processing example.....	105

Figure 5-4: Creating a text classification model	106
Figure 5-5: Comparison of Positive and Negative sentiment	126
Figure 5-6: Comparison of Stress (negative) and Relaxation (positive).....	126
Figure 6-1: Revised system architecture for social media component.....	129
Figure 6-2: Interpretation of the study's kappa score.....	134
Figure 6-3: Top 100-word cloud inspection	136
Figure 6-4: Stress perception on specific motorways (percentage).....	136
Figure 6-5: Geolocation results linked to stress analysis	137
Figure 7-1: City of Birmingham vs Modelled network	145
Figure 7-2: Major intersections in the model	146
Figure 7-3: Examples of roundabouts in the model.....	147
Figure 7-4: Example of a detector and its corresponding description.....	148
Figure 7-5: Flow distribution example	150
Figure 7-6: Proposed optimisation-based imputation methodology	153
Figure 7-7: Location of the experiments.....	157
Figure 7-8: Optimisation-based traffic data imputation detailed flow chart	160
Figure 7-9: M42 J5 location in AIMSUN and Google maps	161
Figure 7-10: Turn proportion on intersections within the M42 J5	162
Figure 7-11: Variable distribution for M42 Junction 5	162
Figure 7-12: Location in the AIMSUN model vs Google Maps.....	164
Figure 7-13: Comparison of variable distribution inside intersections on the M42 J5 and M42 J4	165
Figure 7-14: Turn proportion on intersection within the M42 J4.....	166
Figure 7-15: Variable distribution on the M42 J4.....	166

Figure 8-1: Simulated network vs Google Maps view	174
Figure 8-2: Variable distribution in both roundabouts.....	175
Figure 8-3: Relationship between inputs and exits.....	177
Figure 8-4: Incidents location on the modelled network.....	182

List of abbreviations

AI	Artificial Intelligence
AID	Automatic Incident Detection
ANN	Artificial Neural Network
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
ASCII	American Standard Code for Information Interchange
BN	Bayesian Networks
CAV	Connected Automated Vehicles
CPNN	Constructive Probabilistic Neural Network
CRF	Conditional Random Field
DSR	Design Science Research
EU	European Union
FCM	Fuzzy C-means
FFNN	Feed Forward Neural Networks
FHWA	Federal Highway Administration
FN	False Negative
FP	False Positive
GDP	Gross Domestic Product
GPS	Global Positioning System
HTML	HyperText Markup Language
ICT	Information and Communications Technology
IoT	Internet of Things
ITS	Intelligent Transportation Systems
JSON	JavaScript Object Notation
KARIMA	Kohonen Autoregressive Integrated Moving Average
KNN	K-nearest Neighbours
MAPE	Mean Absolute Percentage Error
MLP	Multilayer Perceptron
NB	Naïve Bayes

NER	Named Entity Recognition
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
NN	Neural Network
PNN	Probabilistic Neural Network
PPCA	Probabilistic Principal Component Analysis
PSO	Particle Swarm Optimisation
RRC	Ridge Regression Classifier
RSS	Residual Sum of Squares
RT	Retweet
SARIMA	Seasonal Autoregressive Integrated Moving Average
SDLC	System Design Lifecycle
SLSQP	Sequential Least Squares Programming
SMS	Short Message Service
SSADM	Structured Systems Analysis and Design Method
SVM	Support Vector Machine
SVR	Support Vector Regression
Tfidf	Term-frequency times inverse document-frequency
TfL	Transport for London
TMS	Traffic Management System
TN	True Negative
TP	True Positive
UN	United Nations
URL	Uniform Resource Locator

Dedication

This dissertation is dedicated to my mother, Emelinda Jones. Her sacrifice, unconditional support and constant love have sustained me throughout my life.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my Director of Studies, Dr Panagiotis Georgakis, for the inspirational guidance, support and engagement provided throughout the period of this research. Without his uttermost flexibility and understanding, this PhD thesis would have not been accomplished. I am also very grateful to my PhD supervisor, Dr Suresh Renukappa, for his encouragements and guidance towards the completion of this thesis.

I would like to thank my fellow PhD-colleagues, for the stimulating discussions, late working times, and every moment we spent together in the last four years. Special thanks to my good friends Pablo Perez and Victor Lajas, who gave me strength, support and unconditional friendship when I needed it the most.

I would like to acknowledge the financial support provided by the government of the Dominican Republic, all through the duration of this research.

Last but not least, I would like to thank my family and loved ones, words cannot express how thankful I am for their warm love and spiritual support throughout this journey.

List of publications

Salas, A., Georgakis, P. and Petalas, Y. (2017) Incident detection using data from social media. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 751-755). IEEE.

Salas, A., Georgakis, P., Nwagboso, C., Ammari, A. and Petalas, I. (2017) Traffic event detection framework using social media. In *2017 IEEE International Conference on Smart Grid and Smart Cities (ICSGSC)* (pp. 303-307). IEEE.

Jones, A.S., Georgakis, P., Petalas, Y. and Renukappa, S. (2018) Real-time traffic event detection using Twitter data. *Infrastructure Asset Management*, 5 (3), pp.77-84.

Under review

Salas, A., Georgakis, P., Petalas, Y. and Renukappa, S. (2019). An imputation methodology for missing traffic flow data in complex networks. *Transportation research*

Salas, A., Georgakis, P., and Renukappa, S. (2019) Social media for policy making in transportation. *Transportation research*

Chapter 1: Introduction

1.1 Background and motivation

Traffic congestion constitutes a social, economic, and environmental issue to modern cities as it can negatively impact travel times, safety, fuel consumption and carbon emissions. In many European cities, commuters find themselves in long tailbacks for hours due to high levels of car ownership, such that roads are operating over their capacity at peak times of the day. In 2017, commuters in Moscow and London spent an average of 91 and 74 hours respectively in peak traffic delays (See Figure 1-1) (Cookson and Pishue, 2017). In fact, drivers in the UK lost an average of 178 hours due to congestion, costing the country £7.9 billion, with London alone contributing to £4.9 billion (Reed and Kidd, 2019).

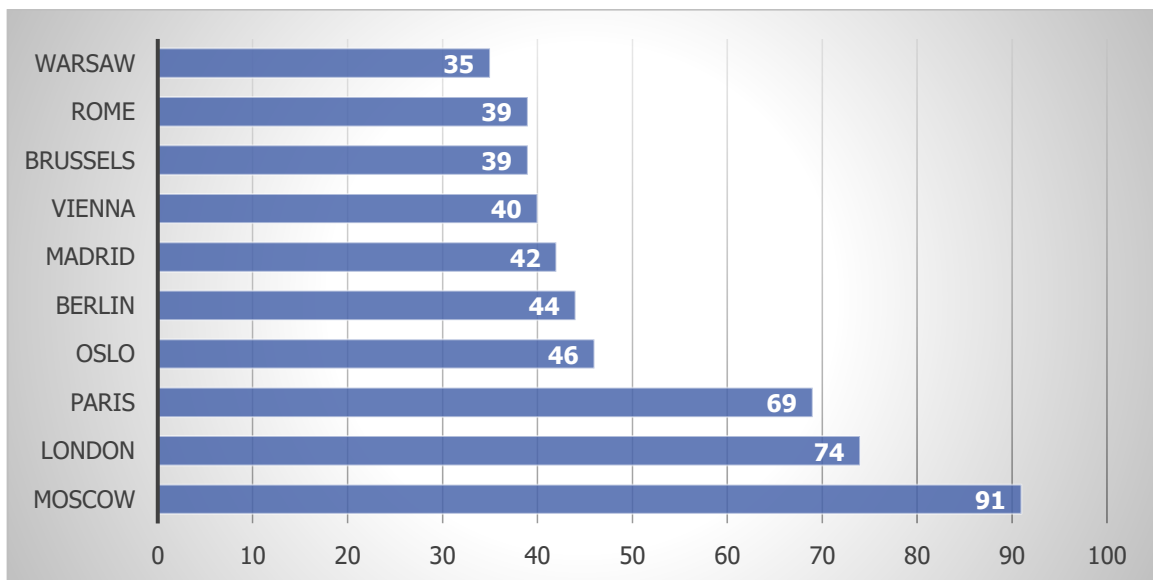
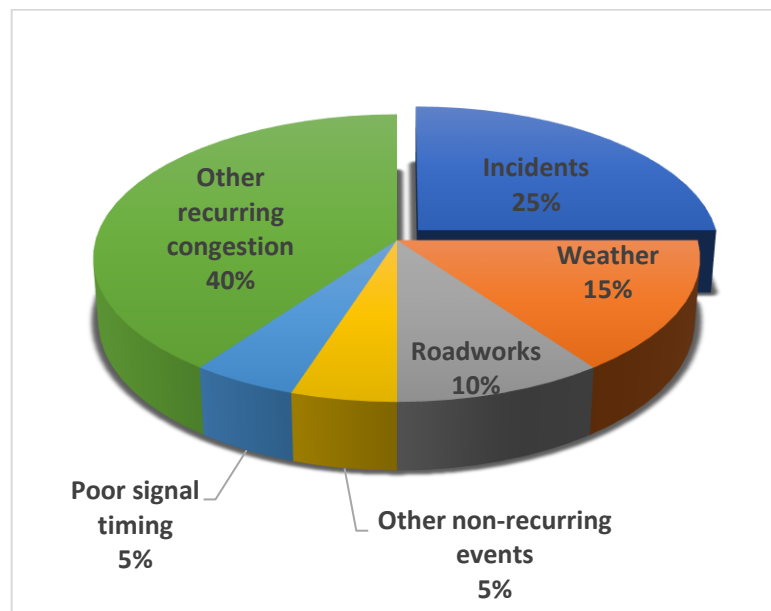


Figure 1-1: Top ten European capitals with worst traffic jams

Increases in traffic lead to more potential conflicts between drivers, and therefore, a heightened likelihood of traffic incidents. Traffic incidents can be defined as non-

recurrent events such as accidents, roadworks, disabled vehicles, inclement weather, social activities, debris, spilled loads, and other unexpected events that disrupt normal traffic. Incidents are one of the major contributors to traffic congestion, accounting for more than half of all traffic congestion as shown in Figure 1-2 (Helman, 2004; Al Rillie and Byard, 2006; FHWA, 2018;). During an incident, the normal capacity of the network is restricted, leading to bottlenecks and delays (Fonseca *et al.*, 2009). In fact, research has indicated that highway efficiency can be decreased by up to 79%, depending on the severity of the accident (Dogru and Subasi, 2018) .



Adapted from Helman (2004)

Figure 1-2: Causes of Highway congestion

Although the most associated issue with traffic congestion is traveller delay, the most serious problem is how congestion caused by incidents may affect the safety and mobility of all travellers. For each additional minute that an incident remains on the road, the likelihood of a secondary crash grows by 2.8 percent (Khattak, Wang and Zhang, 2012). The Federal Highways Administration (FHWA) defined secondary crashes as unplanned incidents (e.g. crashes, engine stalls, overheating, running out

of fuel) that occur because of an original incident (FHWA, 2004). Secondary crashes are often more severe than the original incident, and lead to increased risks of additional crashes and reduction on highway capacity (Xu *et al.*, 2016). In the United States, secondary crashes account for approximately 18 percent of all fatalities on highways, and around 20 percent of all traffic incidents (Yang, Bartin and Ozbay, 2013). Traffic Management Systems (TMS) can help reducing these negative effects by managing the road network, allocating resources, deploying strategies and notifying the users accordingly. Traffic forecasting and incident detection systems are fundamental areas of TMS that have been widely researched in the last decade. These systems provide real time information about traffic congestion and other unexpected incidents that can support traffic management agencies. Depending on the sources of data utilised, two types of TMS have been proposed. The first type exploits data collected from sensors placed in the transport network such as loop detectors, surveillance cameras and infrared sensors. Automatic Incident Detection (AID) algorithms detect traffic incidents by finding anomalies in the traffic data measured by these sensors. However, these systems have limitations with regards of cost, operational performance and area coverage (Dia and Thomas, 2011). The second type collects traffic data from mobile sensors, such as vehicles equipped with Global Positioning Systems (GPS) and cellular geolocation systems (Parkany and Xie, 2005). These sensors can overcome the main limitations of roadway-based sensors, namely, high installation costs and limited coverage of the transport network (D'Andrea and Marcelloni, 2017). Compared to road-based sensors, these types of sensors have the advantage of covering wider areas of the transportation network with lower associated

costs. Nevertheless, existing algorithms need a large volume of mobile sensors, which will require a wide among of users willing to share their location.

While there is a considerable amount of literature around traffic prediction and automatic incident detection systems, current technologies still lack of traffic parameters measurement accuracy and real-time report of traffic events (Dia and Thomas, 2011). In fact, several surveys have found that in many traffic management centres, AID systems are not in use due to high false alarm rate and low accuracy (Parkany and Xie, 2005). In addition, most of the methodologies proposed have been tested on highways, but their usability has not been confirmed for arterial roads. Finally, there is a limitation in resources to support solutions with sophisticated equipment that could guarantee higher accuracy.

A modern TMS should collect data from heterogeneous data sources and provide real-time simulation and visualisation of the transport network. Traffic simulation tools can help to visualise the impact of incidents into the expected traffic conditions, thus better decisions can be taken into consideration in case of an incident. In contrast, Djahel *et al.* (2015) argue that a TMS for future smart cities should incorporate social media data, in addition to traditional data sources. Social media data can enrich the real-time perception of traffic conditions in modern cities. For instance, some users turn to Twitter to report traffic incidents, to describe the traffic situation they are currently in, or to receive traffic information from official traffic management accounts. This information can potentially help to reveal the real causes of the sudden increase in congestion level and help identifying the issues that are directly affecting the transport network (De Souza *et al.*, 2017). Moreover, it could also aid traffic management

agencies to have a better understanding of the citizens' perception of the road network. Several researchers have developed different methodologies for exploiting Twitter data for traffic event detection. Although they obtained promising results in regards of accuracy, some of these studies worked on historical data rather than in real-time. In addition, it still constitutes a challenge to identify the location of the incident, as users do not tend to share their location, and it is hard to identify location from abbreviations. Finally, there is a general lack of research about incorporating user emotions for policy making.

1.2 Research questions

From the above discussion, the fundamental research questions which need answering in order to bridge the knowledge gaps are:

- To what extent Twitter data and simulations of the transport network can support TMS on a real time-basis?
- Can data from real-time streams of information be successfully used for the extraction of traffic events?
- Is the information extracted from social media useful in the context of opinion mining for transportation?
- Can a realistic traffic simulation model be built with limited availability of traffic data?

1.3 Aim and objectives

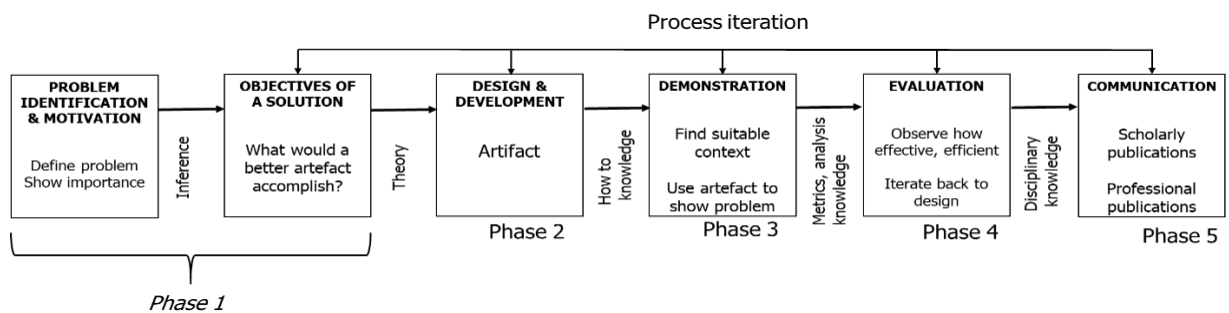
In order to fulfil the research questions stated above, this PhD thesis envisages to develop a smart traffic management framework using heterogeneous data sources. The proposed framework aims to integrate social media data and real-time simulation

outputs as part of existing traffic management systems. This is to be achieved through the utilisation of machine learning algorithms, Natural Language Processing (NLP) techniques, and traffic simulation tools. The objectives needed to achieve this aim are as follows:

1. To review existing developments on traffic prediction and incident detection systems, as well as the current state-of-the-art into the use of social media sources for real-world event detection.
2. To design a conceptual framework that allows the integration of heterogenous data from systems, devices and traffic simulation tools as to offer a platform for the realisation of traffic management services.
3. To develop a social media component that uses state-of-the-art NLP and machine learning techniques to process real-time Twitter data.
4. To develop a simulator component that provides large-scale real-time traffic simulations of the transport network, and a novel technique to impute missing traffic flow data.
5. To develop the proposed framework by the integration of the social media and simulator components.
6. To validate the effectiveness of the framework through its real-time deployment in the region of the West Midlands.

1.4 Research methodology

The main purpose behind the work in this thesis is to develop a smart traffic management framework. To this end, the methodology selected for this thesis follows the Design Science Research (DSR) methodology introduced by Hevner *et al.* (2004). DSR comprises the rigorous process of designing artifacts to solve observed problems, to make research contributions, to evaluate the designs, and to communicate the results to appropriate audiences (Hevner *et al.*, 2004). Figure 1-3 presents the different phases of the DSRM process adapted to the research design of this thesis. The following subsections describe more in depth each phase and the thesis chapter related to it.



Adapted from Peffers *et al.* (2007)

Figure 1-3: Design Science Research methodology process

1.4.1 Phase 1: Problem identification and objectives of a solution

The first stage of the methodology encompasses identifying the problem and justifying the value of a solution. A literature review was carried out to acquire knowledge about the problem and current solutions. Chapter 2 overviews traffic measurements, traditional sensor technologies, algorithms and existing traffic management systems methodologies. Chapter 3 explores the use of Twitter for real-world event detection, as well as the challenges involved with exploiting Twitter data. From these chapters, the research gap along with the aim and objectives of this work were identified.

1.4.2 Phase 2: Design and development

This phase constitutes the design and development of the artifact. The complete design of the proposed framework can be found in Chapter 4. The different natural language processing techniques, machine learning algorithms, geolocation methodologies and sentiment analysis procedures considered for the methodology are introduced in this chapter. This chapter also presents the selection of the traffic simulation tool that will be part of the simulator component of the framework.

1.4.3 Phase 3: Demonstration

This phase involves demonstrating the use of the framework to solve one or more instances of the problem. In chapter 5, different experiments are carried out to test the performance of the techniques proposed in chapter 4. The effectiveness of the proposed framework to identify and classify traffic tweets, as well as a comparison with other approaches in the literature can also be found in this chapter 5. Finally, Chapter 7 presents the simulation platform, along with the different techniques used for importing traffic data and running simulations. It also presents the optimisation procedure for the imputation of missing traffic data in complex networks.

1.4.4 Phase 4: Evaluation

This phase comprehends measuring how well the frameworks supports a solution to the problem. In this thesis, the effectiveness of the framework was evaluated with an experimental implementation of a real-time traffic event detection using Twitter data. Chapter 6 presents the results and validation from the real-time implementation of the social media processing system. In chapter 8, results from the real-time operation of the proposed imputation methodology are presented.

1.4.5 Phase 5: Communication

This step consists of communicating the artifact, its design and effectiveness. The main findings of the work presented in this thesis have been published in scholarly publications. This thesis resulted in two conference papers and one journal paper. The publications can be found in the list of publications section.

1.5 Research gaps and contribution to knowledge

Table 1-1 illustrates the alignment of the research gaps and the contribution to knowledge of this research, based on literature review findings from Chapter 2 and Chapter 3.

Table 1-1: Research gap and contribution to knowledge of the research

Research gap	Contribution to knowledge
Smart traffic management systems do not fully exploit the combination of simulation and modelling tools and data from heterogeneous sources.	A smart traffic management framework that exploits data from heterogeneous data sources to improve both traffic prediction and incident detection techniques, and provide real-time simulations of the road network.
The performance of existing traffic management systems is negatively influenced by missing traffic data due to sensor malfunction.	A simulator component powered by an imputation methodology for missing traffic flow data in complex networks using constrained optimisation.
Existing imputation methodologies for missing traffic data work well on highways and have not been validated in complex networks.	
Twitter data has proven to be beneficial for many real-world purposes but hasn't been widely assessed in the context of transportation applications.	A social media component to identify traffic events and issues on real-time. It uses state-of-the-art techniques for processing, classifying and geolocating the tweets on real-time.
There are still many challenges associated with mining Twitter data, related to text mining and geolocation techniques.	
Opinion mining using social media data is a promising field that can help in obtaining feedback from drivers about optimal routes and the state of the transport network.	A sentiment and stress analysis task within the social media component with the purpose of identifying the users' perception of the transport network.

1.6 Thesis outline

Figure 1-4 illustrates an in-depth design flow layout of the thesis mapped with the DSRM stages followed by this research. The content of the thesis are detailed in the following chapters:

- Chapter 2 provides a review of the literature on the most relevant traffic parameters, sensor technologies and algorithms used for TMS. It then presents a survey of TMS related to incident detection, traffic forecasting and traffic flow modelling. The different challenges faced by these systems are highlighted in this section, along with a review of the imputation methodologies for missing traffic data found in the literature. The chapter finalises by explaining the concept of a smart city and its relevance to the transportation domain.
- Chapter 3 reviews the potential of social media data for real-world problems. It starts by surveying existing research on the identification of events using Twitter data. It then goes on to the twitter-based transportation research focused on incident detection and traffic forecasting. It also emphasises the potential of using user-generated data for policy making, and the possible impact on the transportation domain. Challenges of using Twitter data for event detection are also explained in this chapter.
- Chapter 4 introduces the smart traffic management proposed in this research. The chapter starts describing the research methodology followed, as well as the outcome of this research. In the following sections, the conceptual framework, its features and the proposed components are described more in depth. The

chapter also introduces the study area and the selected simulation tool for the implementation and operation of the different components.

- Chapter 5 presents the detailed architecture of the social media component. In this chapter historical Twitter data from a three months period is used to evaluate the proposed social media component. Five different machine learning algorithms are compared with the purpose of finding the most accurate classifier. During this chapter, the geolocation, sentiment and stress analysis techniques are further described and tested. Results from the implementation are compared with similar studies in the literature.
- Chapter 6 evaluates the proposed social media component over a real-time implementation. A real-time processing pipeline is created using the most accurate techniques detected in chapter 5. Traffic event data identified in Twitter were validated using events data from news websites. Results and recommendations for future work are presented at the last part of the chapter.
- Chapter 7 introduces the proposed imputation methodology for missing traffic flow data using constrained optimisation. In this chapter, the simulated network and the different processes implemented to set up the model are presented. The different experiments and results obtained from the case study evaluation of the proposed methodology are also shown in this chapter.
- Chapter 8 presents the real-time operation of the imputation methodology on a complex network. Using the different techniques develop to set up the network, a processing pipeline is created. The imputation methodology is evaluated on a 24-hour period in 15-minute intervals. A discussion of the results is provided in the final part of the chapter.

- Chapter 9 reports the conclusions and contributions to the body of knowledge made under this research in the area of smart traffic management. Recommendations for future work are also highlighted in the final part of this chapter.

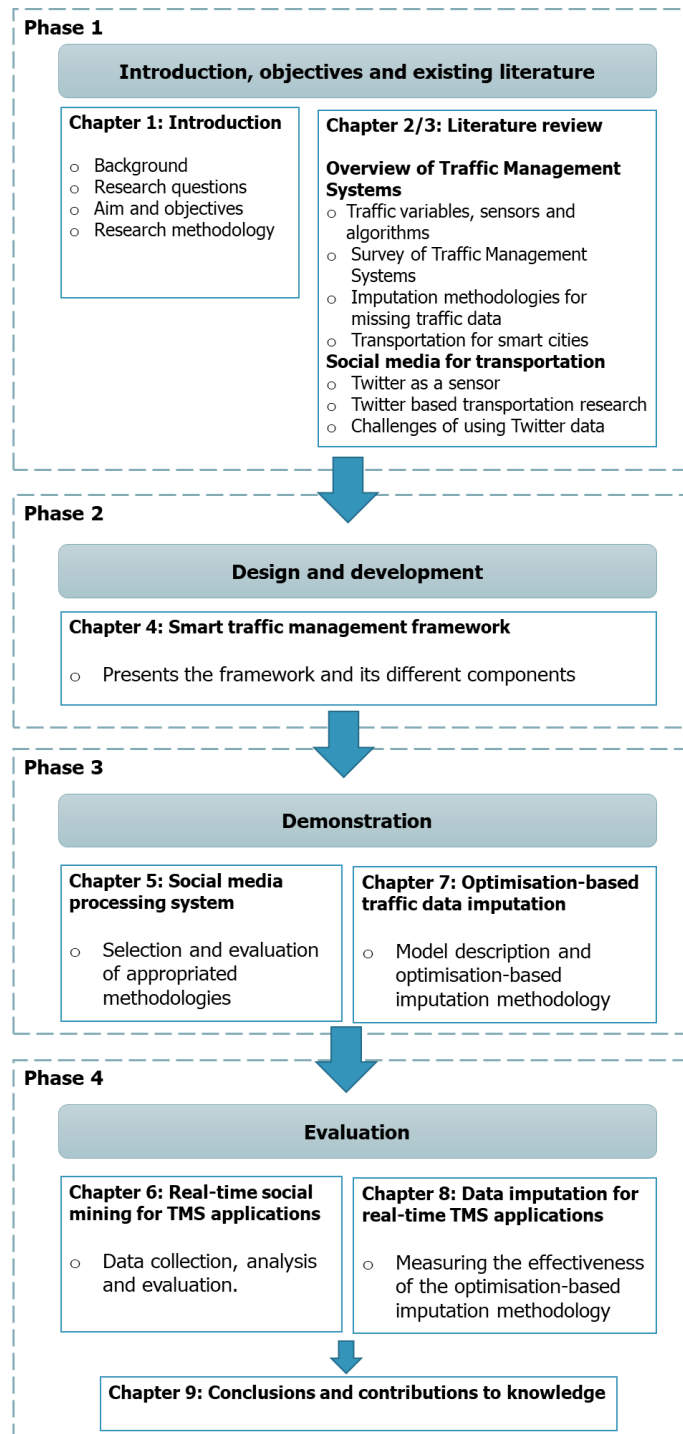


Figure 1-4: Thesis layout

Chapter 2: Overview of Traffic Management Systems

2.1 Introduction

Traffic management systems (TMS) play an important role in the design and deployment of Intelligent Transportation Systems (ITS). These systems provide real time information about traffic congestion and other unexpected incidents that can support traffic control centres to activate strategies and notify users accordingly. A typical TMS involves of a set of complementary stages, as shown in Figure 2-1. Data gathering involves the acquisition of traffic parameters (e.g. speed, flow, density) collected from sensors placed on the transport network. Subsequently, these data are aggregated during the Data aggregation stage, with the purpose of extracting useful traffic information. The next stage, Data exploitation, uses the processed data to compute traffic forecasts, incident detection, and other traffic statistics. Lastly, the Service Delivery stage delivers the information to the users through a variety of control devices. This chapter starts by providing an overview of the most relevant traffic parameters, sensor technologies and algorithms employed in existing TMS, in section 2.2. Section 2.3 reviews existing research with regards of TMS, including incident detection systems, traffic forecasting and traffic flow modelling. Then, in Section 2.4 different imputation methodologies for missing traffic data are reviewed and discussed. A future perspective of TMS and its role in smart cities is provided in Section 2.5.

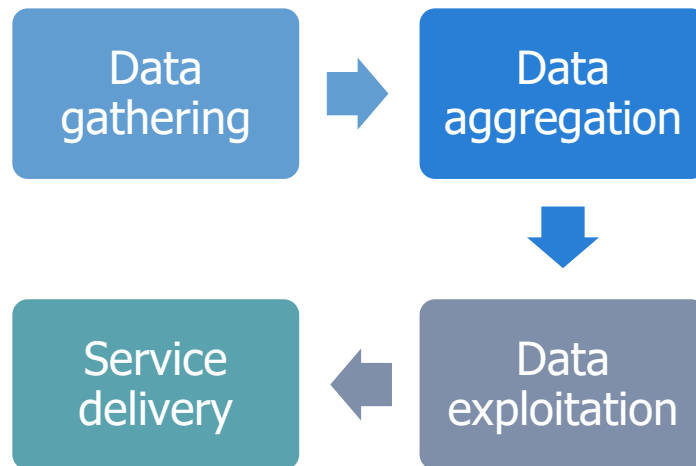


Figure 2-1: TMS activities

2.2 Traffic variables, sensors and algorithms

Modern traffic management centres rely in sensor technologies such as detectors, cameras and probe vehicles, to record data related to the state of the transport network. These technologies employ different traffic variables to describe the conditions of the network, e.g. speed, traffic flow and occupancy. This information arrives periodically to traffic management centres, where is processed by algorithms in order to identify anomalies or forecast the future state of the network. This section provides a general description of the most relevant traffic variables, sensor technologies and algorithms adopted in TMS.

2.2.1 Traffic variables

Traffic variables are the means used for describing the state of the network in a specific time interval. These measurements provide the basis for evaluating the performance of the transport network. The fundamentals of traffic streams measurements and definitions are based on the work of Wardrop (1952), Lighthill and Whitham (1955) and Edie (1963). Three basic variables -traffic flow, speed and density- have been

denoted the most important characteristics of traffic (Gerlough and Huber, 1976), and continue to form the underpinnings of traffic analysis (Mannering, Washburn and Kilareski, 2009). This section presents a brief description of these three traffic variables.

2.2.1.1 Traffic flow

Traffic flow, or volume, is defined as the number of vehicles passing a specific point on the highway over any period of time. It is represented by the following formula:

$$q = \frac{n}{t}$$

where

q = traffic flow in vehicle per unit time,

n = number of vehicles passing a fixed highway point, and

t = duration of time interval.

When the term volume is used, it is understood that the corresponding value is in units of vehicle per hour (veh/h). While the flow is generally measured over the course of an hour, it can be recorded daily, hourly or sub-hourly. Mannering, Washburn and Kilareski (2009) argue that in practice, flow is usually based on the peak 15-minute interval within the hour of interest.

2.2.1.2 Traffic density

Traffic density is defined as the number of vehicles on a given length of the highway at some specified time. It is represented by the following formula:

$$k = \frac{n}{l}$$

where

k = traffic density in vehicle per unit distance

n = number of vehicles passing a length of the roadway

l = length of the roadway

It is usually expressed in vehicle per kilometre per lane (veh/km/lane), where the total traffic density would be the sum of the lane traffic densities. Because traffic density provides a measure of the distance between successive vehicles, it reflects mobility based on the psychological comfort of the drivers. As a result, it has been considered as the most important among the three main parameters, as it gives an indication of traffic flow quality (Quek, Pasquier and Lim, 2009).

2.2.1.3 Mean speed

Speed is commonly defined as the rate of movement in distance per unit of time, usually expressed in kilometres per hour (km/h). However, in practice, speed is defined in two ways: time mean speed and space mean speed. The first, time-mean speed, is the arithmetic mean of the vehicle speeds observed at some fixed point on the highway.

It is expressed as:

$$\bar{u}_t = \frac{\sum_{i=1}^n u_i}{n}$$

where

\bar{u}_t = time-mean speed in unit distance per unit time

u_i = spot speed of the i th vehicle

n = number of measured vehicle spot speeds

The second one, space-mean speed, tends to be more useful for traffic analysis purposes. It is defined as the time necessary for a vehicle to travel a specific part of the road, and is expressed as:

$$\bar{u}_s = \frac{l}{\bar{t}}$$

where

\bar{u}_s = space-mean speed in unit distance per unit time

l = length of the roadway

\bar{t} = average vehicle travel time

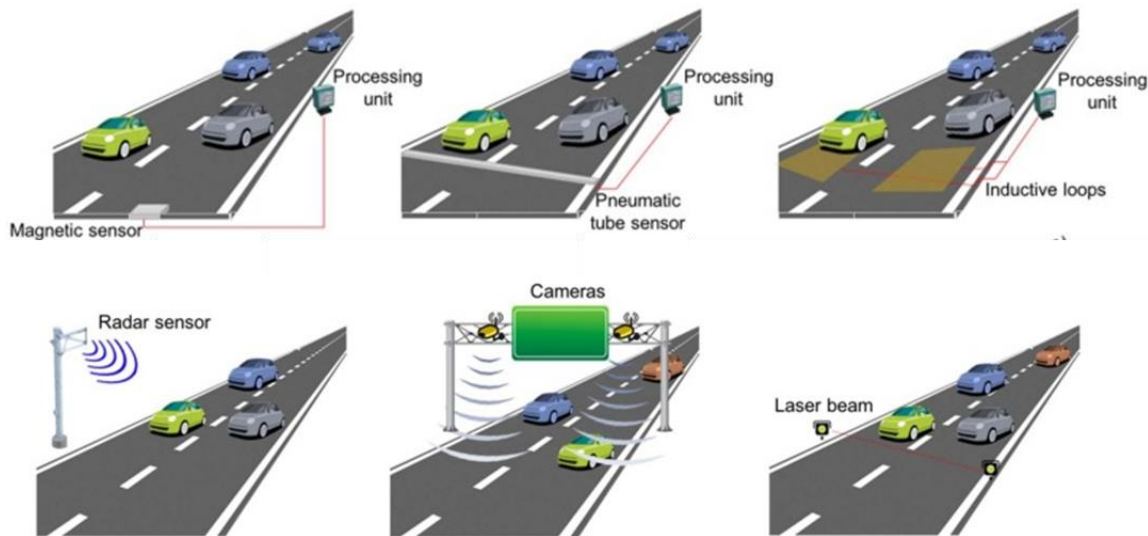
2.2.2 Sensor Technologies

A wide variety of technologies are used for detecting and providing real-time traffic information for TMS. These technologies have been divided in two categories: Roadway Based Sensors and Probe Based Sensors (D'Andrea and Marcelloni, 2017; Parkany and Xie, 2005; Kinoshita, Takasu and Adachi, 2015; Asakura *et al.*, 2017). In the following sub-sections, the most widely used technologies within these categories will be briefly discussed.

2.2.2.1 Roadway based sensors

Roadway based sensors are amongst the most commonly used detector systems. These sensors are fixed on the pavement, installed roadside or mounted over the road (Parkany and Xie, 2005). They obtain traffic information from the point that they are located based on the vehicles passing the detection zone. Traditional roadway-based sensors such as inductive loop detectors and microwave sensors have been employed for decades. In light of recent technology advancements, new sensors have been

implemented powered by cost effectiveness and flexibility (Zhu *et al.*, 2018). Examples of those sensors are magnetometers, infrared sensors and video image processors. Each of these technologies possess different characteristics regarding the principles of operation, accuracy of measures and network coverage (Lopes *et al.*, 2010). Figure 2-2 presents examples of the operation of some of these sensor technologies.



Source: Guerrero-Ibañez, Zeadally and Contreras-Castillo (2018)

Figure 2-2 Example of roadway-based sensors

First introduced in the 1960s, inductive loop detectors are one of the most employed sensors in TMS (Middleton, Gopalakrishna and Raman, 2003; Mathew, 2014). It consists of insulated loop wire that is installed into or under the pavement. When a vehicle drives over the detection zone (area enclosed by the loop), the inductance of the loop decreases. This frequency change is detected by the electronic unit and its interpreted as vehicle detection. However, these technologies suffer from certain disadvantages. For instance, installation and maintenance of these sensors can be problematic as it requires the disruption of the traffic. They are also easily damaged by heavy vehicles, as well as expensive to maintain.

Magnetic sensors, or magnetometers, work on the principle that the presence of a metal object (e.g. vehicle) distorts Earth's magnetic field. A magnetometer detects changes in the vertical and horizontal component of Earth's magnetic field. When operated in the pulse output mode, magnetometers can sense the passage of a vehicle yielding count data. Magnetometers on presence output mode, measures vehicle presence by providing a continuous output if a vehicle occupies the detection zone (Dalla Chiara, Deflorio and Pinna, 2010). They are usually installed on bridge decks and in heavily reinforced pavements, where steel affected the performance of inductive loop detectors (Nelson, 2000). For a similar price, magnetic sensors are easier to install and maintain compare to Inductive loop detectors. They can sustain greater stress and break less often. But they are limited to specific applications as they can only provide passage data (Mimbela and Klein, 2007).

Microwave sensors are divided into two categories: Frequency Modulated Continuous Wave (FMCW) and Doppler Microwave detectors. Doppler microwave detectors use the Doppler principle to calculate the vehicle speed from a continuous wave that transmits electromagnetic energy at a constant frequency. However, these types of sensor can only detect moving vehicles, thus they are not capable of detecting vehicle presence. The second type, FMCW detectors, transmit electromagnetic energy in frequency bands between 2.5 to 24GHz. FMCW sensors can provide count, speed, lane occupancy, vehicle length and vehicle presence. Microwave sensors are a cost-effective alternative to Inductive Loop Detectors and Magnetic sensors due to their low cost, small size and low power consumption (Parkany and Xie, 2005).

Infrared sensors detect energy generated by vehicles, road surfaces and other objects into the sensor aperture. They are divided into two categories: Passive Infrared (PIR) and Active Infrared (AIR). PIR sensors transmit no energy on their own, but detect the energy based on emission or reflection of infrared radiation. They are used to collect flow, vehicle presence and occupancy. In the AIR sensors, a detection zone is illuminated with low power infrared energy transmitted by light emitting diodes operating in the infrared region. These sensors collect data on flow, speed, vehicle presence, classification and density (Guerrero-Ibañez, Zeadally and Contreras-Castillo, 2018). The operation of these type of sensors are vulnerable to weather conditions such as fog, rain, snow and precipitation (Mimbela and Klein, 2007).

Video Image Processors (VIP) typically consist of a set of video cameras, a microprocessor compute for processing the images, and an algorithm-based software to interpret the images and convert them into traffic data (Minami *et al.*, 2008). Video cameras placed on the network collect images from successive frames to determine the changes in traffic. VIPs can provide vehicle presence, speed, volume, lane occupancy and classification. They can also track a specific vehicle or a group of vehicles through the field view of the cameras, with the purpose of providing origin-destination information. The main disadvantage of VIP systems is that they are susceptible to inclement weather: fog, rain, snow and precipitation (Antoniou, Balakrishna and Koutsopoulos, 2011)

Table 2-1 summarises the advantages and disadvantages of the aforementioned sensor technologies based on installation, maintenance, and performance in different types of weather (Klein *et al.*, 2006; Leduc, 2008). The general conclusion is that most

of these sensors (microwave, infrared and CCTV) are compact and are not roadway invasive, which makes their installation and maintenance relatively easy. However, these sensors are also more vulnerable to inclement weather conditions such as fog, heavy rain and snow. In addition, they can be easily spotted by drivers, resulting in reductions in speed and changing their driving behaviour (Guerrero-Ibañez, Zeadally and Contreras-Castillo, 2018). On the contrary, more invasive sensors such as loop detectors and magnetometers, are insensitive to changes in the weather conditions with a relatively good overall performance. Due to their technology maturity, several solutions have been implemented to address the known issues of intrusive sensors with regards of traffic disruption during installation and maintenance. One solution has been the introduction of wireless battery-powered sensors nodes that are installed over the pavement (Zhou *et al.*, 2016).

Table 2-1: Advantages and disadvantages of roadway-based sensors

Technology	Advantages	Disadvantages
Inductive loop detectors	<ul style="list-style-type: none"> • Low per-unit cost • Relatively good performance • Provides basic traffic parameters • Large experience base • Insensitive to inclement weather 	<ul style="list-style-type: none"> • Installation requires pavement cut • Installation and maintenance require pavement cut • Easily damaged by heavy vehicles and road repairs • Susceptible to stress of traffic and temperature • Multiple loops require to monitor a location
Magnetometers	<ul style="list-style-type: none"> • Less vulnerable than loops to stresses of traffic • Insensitive to inclement weather 	<ul style="list-style-type: none"> • Installation requires pavement cut • Installation and maintenance require pavement cut • Limited application • Medium cost
Microwave sensors	<ul style="list-style-type: none"> • Installation and repair do not disrupt traffic • Multiple lane operation • Insensitive to weather 	<ul style="list-style-type: none"> • Relatively low precision • Doppler sensors cannot detect stopped vehicles

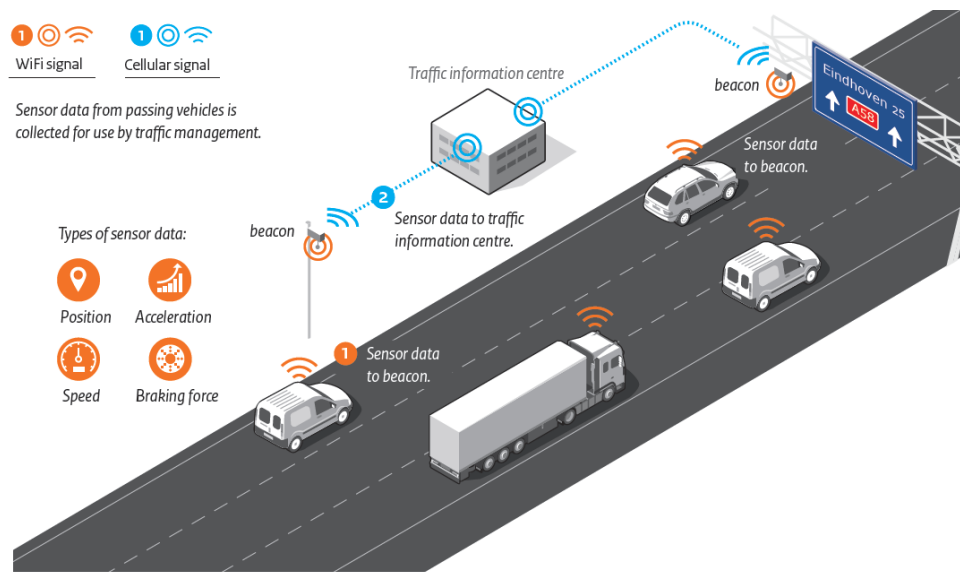
Infrared sensors	<ul style="list-style-type: none"> • Installation and repair do not disrupt traffic • Compact size 	<ul style="list-style-type: none"> • Operations may be affected in fog, heavy rain and snow • Some models not recommended for presence detection
Video image processors	<ul style="list-style-type: none"> • No traffic disruption for installation and maintenance • Monitors multiple lanes • Provides vehicle tracking • Live images of traffic 	<ul style="list-style-type: none"> • Live images require expensive equipment • Susceptible to inclement weather • Different algorithms for day and night use

2.2.2.2 Probe based sensors

Probe-based refer to sensors mounted within the vehicle and have the capability of sending real-time positioning data through wireless communications. These types of sensors move along the transport network and transmit individual vehicle's movement, such as position, velocity and timestamps. With the growing number of GPS trackers installed both in vehicles and mobile devices, it is becoming more popular to collect data from these sensors as they can reflect better the traffic conditions. Compared to roadway-based sensors, these sensors can cover a wider area of the transport network with less investment in cost and maintenance. However, existing algorithms need a large volume of traffic data which will require a proportionally large number of users who are willing to offer GPS data in real time (Siripanpornchana, Panichpapiboon and Chaovalit, 2016).

Global positioning systems (GPS) are designed to track the location of a specific vehicle at a particular point in time. GPS is a global navigation positioning system that transmit geolocation and time stamps to a GPS receiver. With GPS information from moving vehicles, it is possible to match the vehicle position, along with the speed and time parameters by using geographic information systems and map matching techniques.

An example these technologies are those identifiable vehicles that share mobility data with roadside services (Lopes *et al.*, 2010), as shown in Figure 2-3. While a wide variety of authors have successfully used GPS data for traffic forecasting and incident detection, there are some challenges involved with matching GPS data to specific roads. In addition, GPS technologies struggle to transmit through most objects or specific locations such as bridges and tunnels.



Source: Rijkswaterstaat (2015)

Figure 2-3 Example of Probe Vehicle data service

Traffic management applications using cellular geolocation technology are currently being researched by a wide variety of authors. Similar to in-vehicle GPS systems, this sensor technology is based on the widespread use of mobile devices equipped with a GPS receiver. Google Maps¹ is a good example of using cellular geolocation systems to analyse the traffic and road conditions. When users enable location services, their phones send anonymous data to the Google servers which analyse the number of cars and their speeds on a road at any given time. The main advantage of this sensor

¹ <https://www.google.com/maps>

technology is that as it makes use of an existing infrastructure, it does not require alteration of the base station or any installation. In addition, contrary to in-vehicle systems, cellular positioning systems have been tested to work better in complex environments. However, there is an issue involved with privacy as users would need to give permission for the access of their location, or have their geolocation systems turn on. Popular mapping applications, similar to Google Maps, are limited by low sampling rates making it difficult to understand the traffic situation.

2.2.3 Algorithms

Different algorithms have been employed for traffic management applications such as traffic flow forecasting and incident detection. With the purpose of creating an efficient TMS, it is necessary to provide a continuous flow of information about the way traffic conditions evolve over time. Due to its importance, a wide number of researchers have studied and implemented a significant number of methods for the prediction of traffic flow and automatic identification of accidents. The following sub-sections briefly discuss the algorithm approaches often found in the literature.

2.2.3.1 Time series models

A time series is a set of observations sequentially recorded over a period of time. Time series algorithms use these observations to build a model describing the underlying relationship. This model is then implemented to forecast future time series values. First introduced in the 1976 by Box and Jenkins (1976), Auto-regressive integrated moving average (ARIMA) is one of the most popular time series models employed in the literature. The Autoregressive (AR) component predicts the variable of interest based on a regressed combination of past values, while the Moving Average (MA) component

indicates a linear combination of past forecasts errors. The “integrated” part computes the difference between consecutive raw observations, with the purpose of making the time series stationary. These models employ the Box–Jenkins method finds the most appropriate model (AR and/or MA) based on assessing stationarity, identifying seasonality, and estimating the components of the ARIMA model. The major limitation of ARIMA models is that a linear correlation structure is assumed in the time series values. For this reason, the results of using these types of models for complex problems are not always satisfactory. Throughout the years, the ARIMA model has been the foundation for several variants, such as KARIMA (Van Der Voort, Dougherty and Watson, 1996), VARMA (Kamarianakis and Prastacos, 2003) and STARIMA (Kamarianakis and Prastacos, 2003).

2.2.3.2 Traffic simulation models

Traffic simulation is the mathematical modelling of real-world transport systems used to better understand, plan and design transportation networks. They are powerful tools for analysing very complex transport problems, assessing the impact of infrastructure options, and demonstrating how the system is likely to perform in the future. Based on the level of detail that the simulation seeks to achieve, traffic simulation models can be classified into three categories: microscopic, macroscopic or mesoscopic.

Microscopic simulations describe the behaviour of every vehicle and their interaction with other elements in a Multi-Agent System (Ehlert and Rothkrantz, 2001). It allows the user to assign specific characteristics to each vehicle such as type (e.g. truck, car), driving style, route and destination. One of the most common microscopic models is the Cellular automata, where roads are divided into cells which can be empty or

occupied by a vehicle (Maerivoet and De Moor, 2005). Cellular Automata model have proved to be very efficient in capturing the different phenomena that occur in traffic flows (Barlovic *et al.*, 1999; Chowdhury, Santen and Schadschneider, 2000).

Macroscopic simulations, rather than describing the behaviour of specific elements, represent traffic flow at a low level of detail via aggregate traffic variables such as density, speed or traffic count. Contrary to microscopic models, macroscopic approaches are preferred for large scale systems with higher density due to their fast-computational demand.

Mesoscopic simulations are a combination of microscopic and macroscopic models. Mesoscopic models fill the gap between the individual behaviour of microscopic models and the aggregate level approach of macroscopic ones (Burghout, Koutsopoulos and Andreasson, 2005). The main application of these models is when the detail of a microscopic simulation is needed, but it is not feasible due to the large size of the network.

2.2.3.3 Artificial Intelligence Algorithms

Artificial intelligence refers to a set of techniques that emulate the human reasoning process in complex real-world problems. The most popular artificial intelligent techniques employed for traffic management systems include Neural Networks, Fuzzy Logic, K-nearest Neighbour and Support Vector Regression. With the purpose of improving the efficiency of these methods, several hybrid techniques have emerged from the fusion of these algorithms with other models.

First introduced in the 1950s by Rosenblatt (1958), Neural Networks are computing systems inspired by the biological neurons of the human brain. Neural network is not

an algorithm, but a set of many different machine learning techniques that are able to process complex data input into a desired output. They are composed of many interconnected units called 'neurons' that mimic the operation of a human brain. Each of these neurons have many input signals, but only one output that is generally sent as an input to other neurons, until they produce the final output (Hecht-Nielsen, 1988). One of the main characteristics of these systems is that they learn from processing many examples, without the need to be reprogrammed. However, these algorithms need substantial training due to their complex internal structure. The most common used neural network algorithm for traffic related purposes are the Feed Forward Neural Networks (FFNN) and Probabilistic Neural Networks (PNN).

The term Fuzzy Logic was introduced by Zadeh (1965), it consists of a problem-solving system that intends to model logical reasoning, in which true values are fuzzy subsets of the unit interval. Fuzzy subsets assign a degree of truth, these values range from 0 (totally false) to 1 (totally true). It provides a mechanism that captures and represents real-world uncertainty with its fuzzy data (Zadeh, 1996). Their main advantage is their ability to make decisions based on incomplete data, which can aid traffic management systems with missing sensor data.

K-nearest neighbour (KNN) is a simple regression algorithm introduced 1967 by Cover and Hart (1967). The fundamental assumption of KNN is that future states to be predicted are similar to a neighbourhood of the past. Once the neighbourhood has been recognised, the past cases are used to forecast the values of the dependant variable (Smith, Williams and Keith Oswald, 2002). Similar to other AI approaches, KNN's computational efficiency depends on the vastness of its training data. The

biggest disadvantage of this approach is the selection of the database of potential neighbours. Selecting a large database, while desirable, it can have significant implications on the computing time.

Support Vector Machine (SVM) was originally developed by Cortes and Vapnik (1995), to solve pattern recognition and classification problems. Support Vector Regression (SVR) is an extension of SVM that can handle non-linear regression estimation problems. In essence, the input data is mapped into a hyperplane in a high-dimensional space, where a linear function that formulates a non-linear relationship between the input and output data (Hong, 2011). While this method has proven to be very effective for complicated network traffic problems, poor forecasting accuracy has been experienced from lack of knowledge on the selection of the different tuning parameters.

2.3 Survey of Traffic Management Systems

This section reviews the different traffic management systems applications. For the purpose of providing a consistent review, existing applications have been classified into three categories: incident detection, traffic forecasting and traffic flow modelling systems.

2.3.1 Incident Detection Systems

The increasing number of vehicles causing traffic congestions, bottlenecks, and incidents, has prompted a growing interest in developing efficient Intelligent Transportation Systems (ITS). The purpose of ITS is to take advantage of advanced communication, information and electronics technology to solve transportation problems such as traffic congestion, safety and transport efficiency (Figueiredo *et al.*,

2001). One of the main challenges of ITS is to distinguish an incident from a traffic congestion (D'Andrea and Marcelloni, 2017). A traffic incident is an unexpected event that temporarily disrupts the normal traffic flow, such as broken-down vehicles, accidents, roadworks, severe weather or other unusual events. Traffic incidents are one of the major causes of traffic congestion. In fact, it has been indicated that for every minute a traffic incident remains uncleared, it takes around four minutes for traffic to go back to normal conditions (Saka, 2000). It is, therefore, not surprising that Automatic Incident Detection (AID) constitutes an important part of ITS that has attracted the interest of many researchers in the last decades. Effective and efficient AID systems can accelerate the response and intervention of traffic management agencies, and reduce the possible loss caused by incidents in terms of life, money and time (Wang *et al.*, 2013).

Algorithms techniques mentioned in the previous sections, have been successfully employed for automatic incident detection systems. ANN has been one of the most popular and efficient techniques in the literature. The application of ANN for incident detection was first introduced by Ritchie and Cheu (1993), where they proposed ANN for the automatic detection of lane blocking incidents on a freeway. But then, Wen *et al.* (2001) developed a Probabilistic Neural Network (PNN), where they considered incident detection as a pattern recognition problem. They evaluated the performance of their PNN by simulating a wide range of incidents with a variety of flow conditions and traffic periods. The main limitation of PNN is that due to the large neural network size, a large memory and computational time is required. Jin, Xin, Cheu and Srinivasan (2002) built a Constructive Probabilistic Neural Network (CPNN) that provided solutions

to the memory control and adaptation issues mentioned before. Results from this study showed that CPNN had better overall performance than conventional PNN techniques.

Recently, there has been a growing interest in fusing data from roadway-based sensors and probe vehicles. Yu *et al.* (2008) proposed a fusion of loop detector and probe vehicle data for traffic incident detection using a Back Propagation (BP) neural network. In their model, a Cumulative Sum (CUSUM) approach was used to detect incidents based on loop detector and probe vehicle data. The BP neural network combines the output from the two sources and determines if there is an accident by comparing the fusion output with the defined threshold. Dia and Thomas (2011) tested various neural network data fusion methodologies based on simulated loop detector and probe vehicle data for incident detection on arterial roads, resulting in a 90% incident detection rate and 0.5% false alarm rate. In the last few years, several authors have developed ANN models based on different methodologies and data sources (Ki *et al.*, 2018; Zhu *et al.*, 2018; Chakraborty *et al.*, 2018). The main drawback of ANN models is their complex internal structure and that they require a large set of historical data.

Fuzzy logic is another technique that has gained popularity in the field of AID (Lee, Sibok, Krammes and Yen, 1998; Ahmed and Hawas, 2013; Rossi *et al.*, 2015; Nikolaev *et al.*, 2016). For instance, Rossi *et al.* (2015) presented a fuzzy logic approach using fuzzy linguistics to describe the variation of traffic parameters. They used the traffic simulation tool PARAMICS to evaluate the performance of the system in terms of detection rate, false alarm rate and mean time to detection. Their implementation showed excellent results for high flow rate values (more than 3000 veh/h), and a performance decrease in instances with low flow rate. The most notable disadvantage

of fuzzy logic approaches is that it requires extensive calibration due to the number of parameters that need to be defined.

Some of the studies mentioned before, whilst they have obtained low false alarm rates, have only been tested using traffic simulation tools. On the other hand, other literature suggest that they do not meet the operational need of traffic management centres as they demand a much lower false alarm rate to minimise operator overload (Dia and Thomas, 2011). In fact, results from a survey in the US showed that more than 53% of TMS operators are not satisfied with their current AID system due to malfunctions, high false alarm rates, inconvenient installation/maintenance, and difficult calibration and implementation (Parkany and Xie, 2005). Furthermore, in Williams and Guin (2007), more than 70% of the operators considered as insufficient the current methods employed for automatic incident detection.

2.3.2 Traffic forecasting systems

Traffic flow forecasting constitutes a critical aspect of Intelligent Transport Systems (ITS). In order to create an efficient ITS, it is necessary to provide a continuous flow of information about the way traffic conditions, such as traffic flow, occupancy and speed, evolve over time (Lieu, 2000). Vlahogianni, Golias and Karlaftis (2004) defined short-term traffic forecasting as "*the process of estimating directly the anticipated traffic conditions at a future time, given continuous short-term feedback of traffic condition*". Due to its importance, a wide number of researchers have studied and implemented a significant number of methods for the prediction of traffic flow. Most of the existing traffic forecasting methodologies are statistically base methods. Ahmed and Cook (1979), Levin and Tsao (1980) and Nihan and Holmesland (1980) were

amongst the first to implement the ARIMA model for traffic prediction as an alternative approach based on the stochastic nature of traffic. In 1999, Lee, Sangsoo and Fambro (1999) employed a subset ARIMA, instead of using the original coefficient vectors determined by the Box-Jenkins method. Their results showed that the subset ARIMA model gave more accurate results than a full ARIMA model. Billings and Yang (2006) studied the application of ARIMA models for the prediction of travel time on urban roadways. They selected the ARIMA model due to the non-stationary property of the data and indicated the potential of using these types of model for travel time prediction.

The Seasonal Autoregressive Integrated Moving Average (SARIMA) is one of the most popular univariate variations used for traffic prediction. SARIMA models are useful for dealing with the seasonal characteristic of traffic flow. Williams *et al.* (2003) and Williams *et al.* (1998) presented the theoretical basis of forecasting traffic flow as a SARIMA process. Experimental results from both studies outperformed other forecasting techniques. In 1996, Van Der Voort, Dougherty and Watson (1996) introduced the Kohonen Autoregressive Integrated Moving Average (KARIMA) model. This method performs clustering between time series samples, which are aggregated using a Kohonen self-organising map.

Time series techniques tend to have low accuracy and decreased performance when the prediction horizon increases. In addition, due to their mathematical assumptions, they do not perform well under complex real-world environments. For these reasons, AI techniques are preferred when modelling complex datasets with missing data or non-linearities (Karlaftis and Vlahogianni, 2011). ANN is one of the most implemented approaches for traffic prediction problems. Dia (2001) developed an object-oriented

neural network model for predicting short-term traffic conditions. The models described in this study predicted speed data with an accuracy of 94% in a 5 minutes horizon. In contrast, Goves *et al.* (2016) were able to reliably predict traffic conditions 15 minutes into the future using a simplified ANN model. Kumar, Parida and Katiyar (2013) presented an ANN model that had consistent performance in intervals from 5 to 15 minutes, considering the speed of the different type of vehicles as different inputs variables.

While ANN has proven to be highly accurate and capable of modelling a non-linear environment, many hybrid techniques have been based on neural networks as a way of improving its complex internal structure. For instance, Jiang and Adeli (2005) improved neural network by creating a dynamic wavelet neural network model that is able of capturing traffic flow and pattern recognition with enhanced feature detection capability. Park (2002) presented a hybrid neuro-fuzzy application for short-term freeway traffic forecasting. This model consisted of a fuzzy C-means (FCM) component that classified traffic flow patterns into a pair of clusters, and a radial-basis-function (RBF) neural network that developed forecasting models associated with each cluster. In fact, fuzzy-neural approached developed by several authors in the literature (Yin, H. *et al.*, 2002; Srinivasan, Sanyal and Sharma, 2007; Stathopoulos, Dimitriou and Tsekeris, 2008; Tang *et al.*, 2017), have performed better than ANN with a lower computing time requirement.

The main disadvantage of the aforementioned techniques is the need for extensive data for training purposes. This can become a challenge as traffic data can be unavailable due to a number of reasons: malfunctions of hardware or software,

communication network problems, unavailability of hardware, and so on (Bae *et al.*, 2018). In addition, most of these techniques have been developed for freeways, not for arterial roads. Arterial roads are particularly difficult to forecast due to the low sensor coverage and different traffic behaviour.

2.3.3 Traffic flow modelling

Drew (1968) defined a simulation system as '*a dynamic representation of some part of the real world, achieved by building a computer model and moving it through time*'.

Microscopic traffic simulation is an effective and popular tool used to better understand, plan and design transportation networks. Microscopic simulations are a viable alternative for analysing a wide variety of transport problems that are not amenable to study by other means (Lianyu Chu *et al.*, 2003). It offers a more feasible and cost-effective approach to evaluate operational traffic problems such as incident management systems, traveller information and adaptive traffic management. A traffic simulation tool provides the environment where these different issues can be modelled and evaluated, in a controlled setting and without disrupting real-world networks. There are three commonly used traffic simulation tools: AIMSUN (TSS—transport simulation systems, 2008), PARAMICS (Quadstone, 2003), and VISSIM (PTV, 2009). These tools are based in different car-following and lane-changing theories.

AIMSUN (Advanced Interactive Microscopic Simulator for Urban and Non-Urban networks) developed by TSS, integrates traffic assignment, microscopic and mesoscopic simulations. AIMSUN is based on the car following model developed by Gipps, where vehicles are classified as free or constrained by the vehicle in front. It is used to develop and evaluate traffic management rules, traffic control systems, access

controls, location of tolls and can interact with real-time applications, such as vehicle guidance systems (Saidallah, El Fergougui and Elalaoui, 2016). AIMSUN is embedded in GETRAM, a simulation environment that offers a traffic network editor (TEDI), a module for detailed simulations, animated 2D and 3D output of the simulations, and an Application Programming Interface (API), which aims at interfacing to other simulation and assignment models. The model can communicate with external user-defined applications, through an additional library of functions.

PARAMICS (Parallel Microscopic Simulation) developed by Quadstones, is a microsimulation model based on the psycho-physical model developed by Fritzsche in 1994. It uses the lane changing and car following model that are based in the drivers' desire to achieve target headways and speeds. PARAMICS allows running predefined scenarios, which include the opportunity vary flow levels, simulate additional features and user defined algorithms through the API. Moreover, it supports the development of plug-ins to interface with third party software and real-world systems, and it provides a realistic visualisation of the network with 2D and 3D views.

VISSIM developed by PTV, is a multimodal simulator that allows to simulate more than one traffic type, such as motorised private transport, rail and road public transport, cycles, pedestrians and goods transport. It is based in Weidemann's model (1974), a car-following and lane changing model that considers a mixture of physical and psychological aspects of the driver behaviour. VISSIM offers flexibility in modelling geometry with any level of complexity, as well as 3D animations for different scenarios. VISSIM provides a Component Object Model (COM) programming interface to manipulate the different functions and parameters of the simulations. It allows users

to develop their own application that can modify simulation settings, vehicle behaviour, path flows, signal controlling and running simulations.

Many studies have evaluated the effectiveness of AIMSUN, PARAMICS and VISSIM, with regards of different applications. For instance, Panwai and Dia (2005) presented a comparative evaluation of the car-following models in AIMSUN, VISSIM and PARAMICS. They tested the performance of the algorithms in the same dataset, and reported that AIMSUN showed lower error values, with respect of VISSIM and PARAMICS. Xiao *et al.* (2005) presented a multicriteria analysis for selecting a microscopic simulator between AIMSUN and VISSIM, by using both qualitative and quantitative criteria. In their assessment, each criterion was assigned a score, based on the performance of the simulator. Results showed that both AIMSUN and VISSIM can incorporate most of the standard features used in traffic modelling. AIMSUN had a slightly better overall score, overtaking VISSIM on incident management and traffic control capabilities, such as lane blocking, capacity reduction, and adaptive control signals. In general, the literature suggests that there is no obvious difference between these three tools, and the selection of a simulator tool should depend on the specific requirements and purpose of the application (Zhang and Hounsell, 2010; Mahmud *et al.*, 2018).

The credibility of a simulation model lies on its ability to replicate real-world conditions. This can be achieved by incorporating real-world traffic data into the model, as well as other parameters such as speed, drivers' reaction time, visibility distance, and so on. Traffic flow data from the sensors associated to the modelled network are essential to complete a simulation model, as they require the traffic flow measurements and the

turn proportions to be specified as inputs for the simulation. However, as stated before, traffic data from different transport facilities are often missing or incomplete due to sensor malfunction or unavailability of hardware. Becoming this, one of the main challenges associated with creating a realistic simulation model.

2.4 Imputation methodologies for missing traffic data

As established in the previous section, missing traffic data severely limits the application of TMS. In traffic forecasting and incident detection systems, the prediction performance reduces sharply when data is missing for extended period of times (Smith, Scherer and Conklin, 2003). Consequently, it does not come as a surprise the importance of developing a methodology for correctly estimating missing traffic data. Imputation is the practice of replacing missing data with estimated values. Many researchers have developed different methodologies for the imputation of missing traffic data. These methods can be divided into three categories: prediction, interpolation, and statistical based methods (Tan *et al.*, 2013).

Prediction based methods identify a relationship between historical and future data to predict missing data points. These approaches directly used existing traffic prediction models such as ARIMA (Lee, Sangsoo and Fambro, 1999; Zhong, Sharma and Lingras, 2004), Bayesian networks (BNs) (Zhang, Sun and Yu, 2004; Ghosh, Basu and O'Mahony, 2007), SVR (Jin, Xuexiang, Zhang and Yao, 2007; Castro-Neto *et al.*, 2009) and FFNN (Dia, 2001; Vlahogianni, Karlaftis and Golias, 2005). However, two shortcomings have been identified of using these methods (Shang *et al.*, 2018). First, many prediction models cannot use information collected after the missed data, which

would reduce the imputation effectiveness. Second, prediction methods are less effective when a consecutive series of data are lost.

Interpolation based methods predict the missing data with a weighted average of known data that is neighbouring the missing data (Qu *et al.*, 2009). Neighbouring data can be classified in two types: temporal and pattern neighbouring. Temporal neighbouring works with data collected from the same detector at the same time period of neighbouring days. This approach assumes that daily traffic flows are highly similar during several consecutive days, and do not take into consideration daily flow variation to improve accuracy (Zhong, Sharma and Liu, 2005). Alternatively, pattern neighbouring uses historical data from the same detector but on a different day with similar flow pattern of the missing data (Yin, Murray-Tuite and Rakha, 2012). Pattern-similar models are usually more effective than temporal neighbouring, as they use both daily flow variation and similarity information of traffic data from different days. KNN (Liu, Sharma and Datla, 2008) and Local Least Squares (LLS) (Kim, Golub and Park, 2004) are two typical pattern-neighbourhood methods. However, these methods work under the assumption that neighbouring traffic flows are similar, and sometimes this is not the case. Moreover, the performance of these model decreases when the neighbouring traffic data are also missing.

Statistical based methods take advantage of the statistical characteristics of traffic flow by using the observed data to learn a scheme, and then infer the missing data in an iterated fashion. These methods first assume a probability distribution based on the observed data. Then, the values that best fit the assumed probability distribution are estimated through iterations. The imputation performance of statistical based methods

is usually higher than prediction and interpolation, since the assumed probability distribution captures the essentials of traffic flow variations (Li, Li and Li, 2014). Markov Chain Monte Carlo (MCMC) (Ni and Leonard II, 2004) and Probabilistic Principal Component Analysis (PPCA) (Qu *et al.*, 2009b) are two representations of the statistical methods. From the latter, several methods have emerged by blending it with other theories. For instance, Chiou *et al.* (2014) developed a methodology based on a functional principal component analysis (FPCA) that outperformed PPCA.

In the last few years, new methodologies for the imputation of missing traffic data have arisen. Tan *et al.* (2013) introduced a tensor-based method for the imputation of missing traffic data. They formulated traffic data as a tensor pattern, maintaining the multidimensional characteristics of traffic data. Their tensor-based method achieved better performance than probabilistic-based methods, even if the ratio of missing data was 90%. More recently, Asif *et al.* (2016) proposed several matrix and tensor-based methods to impute missing traffic flow values by identifying common traffic patterns in large road networks. Duan *et al.* (2016) implemented a deep learning model that obtained a better imputation performance than ARIMA model and BP neural network model. In 2017, Chen *et al.* (2017) presented a novel correlation-based method is used to first represent traffic data as a matrix, and then use an adaptive KNN to explore the relationship between the missing and complete data. Their approach outperformed probabilistic-based methodologies.

The methods described above work well for prediction of random errors, but sometimes data is missing for extended periods of time, and spatial-temporal information is not available. While many of these methods have been developed for

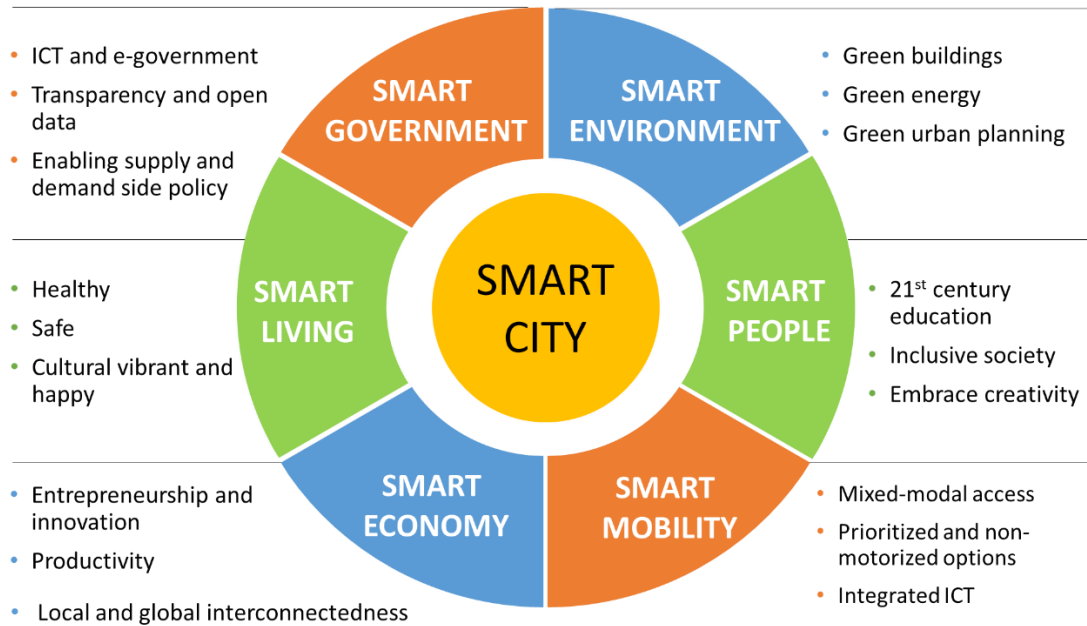
imputation of traffic data for straight section of highways (Tan *et al.*, 2013; Duan *et al.*, 2016; Chen *et al.*, 2017), their suitability for urban roads have not been validated. The continuity of traffic flow is better on a highway than in an urban road, as highways are relatively closed and there are no intersections to interrupt the operation of the traffic flow (Shang *et al.*, 2018). For example, Muralidharan and Horowitz (2009) explained that existing methodologies cannot be applied for on-ramp and off-ramps detectors, as it is not possible to guarantee a high correlation with neighbouring detectors. As a result, they proposed an imputation procedure to determine ramp flow data using the Link-Node Cell simulation model, by using an adaptive identification technique to minimise the error between simulated and measured values. Similarly, Shang *et al.* (2018) developed a hybrid method for missing traffic data on urban expressways and arterial roads. Their method combined Particle Swarm Optimisation (PSO) and SVR and obtained promising results in estimating missing traffic flow data.

2.5 Transportation for smart cities

2.5.1 The smart city concept

Since 2009, it has been reported that most of the world's population lives in cities with more than six devices per person connected to the internet (Evans, 2011). This implies that almost every object around us is connected to the internet via the Internet of Things (IoT). The smart city concept originates from the significant amount of interest towards incorporating IoT and information, and communications technologies (ICT) to improve the quality of human life with respect to social, economic and environmental challenges.

In order to understand this concept, it is imperative to recognise why it is important to make cities 'smarter'. Modern cities play an important role in social and economic aspects and have a massive impact on the environment. According to the United Nations Population Fund, the world's population is expected to increase from 7.7 billion (2019) to 9.7 billion in 2050 and could reach nearly 11 billion around 2100 (United Nations, 2019). In Europe, 74% of the population lives in urban areas and the number is expected to increase to 83.7% in 2050. As a result, urban cities are the major contributors to climate change, producing more than 60% of greenhouse gas emissions and consuming 78 percent of the world's energy (UN-Habitat, 2016). Smart cities applications bring together cities, businesses and citizens with the purpose of minimising their environmental impact, and creating more inclusive, sustainable and connected cities. While the term 'smart' builds upon technological applications, the concept of smart city is far from being limited to technological solutions. With the intent of clarifying what constitute a smart city, many researchers have separated this concept into different dimensions. The smart city wheel, developed by Cohen (2012), has become a widely accepted tool for identifying smart cities based on six dimensions (as shown in Figure 2-4): smart living, smart people, smart economy, smart mobility, smart government and smart environment.



Adapted from Cohen (2012)

Figure 2-4: Smart City wheel

The separation of the smart city concept in six dimensions suggest that cities should not be evaluated in a holistic way, but different aspects need to be taken into consideration. Albino, Berardi and Dangelico (2015) argue that these dimensions rely on traditional and neoclassical theories of urban growth and development: human and social capital, natural resources, regional competitiveness, participation of society members, transport and ICT economics. It goes beyond technological and environmental aspects, as it should promote the engagement of different stakeholders (especially citizens). Indeed, collaboration and innovation between the government, private sector, academia and the citizens is a key driver for success in these six dimensions.

2.5.2 Smart mobility applications

With over 60% of the world's population living in urban areas and sharing the same infrastructure, urban mobility has become one of the most pressing issues for modern

cities (United Nations, 2019). According to the European Commission (2019a) statistical pocketbook, urban mobility constitutes 40% of the CO₂ emissions associated to road transport and up to 70% of emissions from all transportation modes. Moreover, traffic congestion in the EU costs nearly €100 billion annually, approximately 1-2% of the EU's gross domestic product (GDP). Enhancing urban mobility, while at the same time minimising these undesirable impacts on the economy, society and the environment, is a main priority of major cities throughout the world.

According to Horizon 2020, the European research framework program (European Commission, 2013): "transport shall be smart, green and integrated". As illustrated in the smart city wheel (see Figure 2-4), one of the main goals of smart cities is to develop smart mobility solutions for sustainable transportation. The term 'smart mobility' originates from the integration of smart transport and smart user. This is because users should take advantage of smart devices ("things" in the IoT concept), in-vehicle devices, infrastructure sensor technologies, open source electronic platforms and social media data, to enable a real-time integrated mobility information service (Motta *et al.*, 2015). Smart mobility embraces all main transportation domains, with applications providing services for real-time traveller information and communication, traffic management systems, and multimodal mobility (e.g. shared mobility, low carbon initiatives).

In the last few years, researchers from both industry and academia have concentrated their efforts into exploiting advance communication technologies to improve the efficiency of TMS within the smart city concept. The growing use of mobile phones and vehicular wireless technologies have enabled real-time communication between drivers

and traffic management centres. This real-time information sharing aids traffic management agencies to identify changes to the traffic conditions, plan appropriate alternatives and alert road users accordingly. Informing drivers about the present traffic conditions can help them identify the fastest and most economic routes and avoid potential hazards. To this end, a wide amount of smart mobility initiatives have been deployed in the field of real time traveller information by exploiting data from heterogeneous data sources (e.g. crowd, social, open, sensor data). In Italy, Anastasi *et al.* (2013) presented the SMARTY project. The SMARTY project collected data from environmental sensors deployed in the urban area, and user social interaction through social network. All this information was processed to provide the users with real-time traffic condition, parking lot availability, suggestion of optimal routes, and pay for transport services (e.g. parking space, book tickets). Similarly, Farkas *et al.* (2015) proposed TrafficInfo, an application that collected users' position data, public transport vehicles tracking and annotation data entered by the user, and generated live updates of public transport operations. Users are informed about the live public transport situation, such as crowdedness information, vehicle position and deviation from the timetable. In the USA, the city of Louisville adopted a crowdsourcing approach to encourage their citizens to participate in a smart city application (Blayney, 2016). They achieved this by joining Waze's Connected Citizens Program, where Waze exchanged traffic data with the city of Louisville to identify congestion and improve the traffic situation.

In terms of road traffic management optimisation, the research community have proposed and implemented new mechanisms to mitigate traffic congestion.

Considering the impact of congested street intersections on the traffic efficiency, Chen and Cheng (2010) argue that a TMS on a Smart City should put more emphasis on regulating traffic signals on these critical intersections, in order to improve the overall traffic management performance. Many authors in the literature have explored traffic signal optimisation by employing state-of-the-art algorithms and simulation tools. For instance, Ezzat *et al.* (2014) proposed a genetic algorithm to optimises traffic signal timing, based on a non-linear function that uses queuing lengths and vehicular waiting times as performance measures. Similarly, Li *et al.* (2016) integrated a genetic algorithm into a microscopic traffic simulation environment to determine traffic signal settings that could minimise the drivers' average travel time. In Li *et al.* (2017), a framework for optimising traffic signal settings based on a multi-agent system for large-scale transportation systems. They developed an intelligent real-time traffic management mechanism that combined online and offline optimisation technique to configure traffic signal setting for random and uncertain traffic flows.

Shared mobility is another field that has caught the attention of the research community in the last few years. Shared mobility involves the use of a vehicle, bicycle, or other low-speed mode that enable users to gain short-term access to transport modes on a 'as-needed' basis, usually serving as a first-or last-mile connection to other modes, such as public transportation (Shaheen and Chan, 2016). Example of these new mobility paradigms are ridesharing (e.g. carpooling, vanpooling), bike sharing, car sharing and on-demand ride services (e.g. Uber, Lyft). The documented benefits associated with shared mobility are reduced vehicle miles travelled, reduced personal vehicle ownership, reduced greenhouse emissions and increased economic activity

near commercial areas. Mobility as a service (MaaS) is a frequently used term to describe the integration of different modes of transportation, shared mobility services and enables payments in a single interface. Although the MaaS concept is relatively new, there has been many successful MaaS initiatives implemented in Europe such as Whim², Smile³ and Moovel⁴.

The quality, efficiency and connectivity of the public transportation system plays a key role in motivating users to shift from private cars towards new mobility paradigms. Indeed, many cities have invested in making their public transportation system 'smarter'. Debnath *et al.* (2014) evaluated 26 cities according to the 'smartness' in their public transportation. The transport system of London was found to be the smartest of all. London has a wide range of smart technologies deployed in their public transport system: buses equipped with Automatic Vehicle Location Systems, intermodal electronic fare collection, driverless control of transit vehicles, bus lane enforcement system and a whole corridor traffic signal priority system for public transport buses. Additional to these, initiatives for shared transport services have been also deployed in London. BlueCity⁵ is an electric car sharing scheme offering point to point service. Users can drive around the city by booking an electric car through the BlueCity app and paying only for the amount of time used. This scheme has hundreds of drop-off locations and cars available around London. Similarly, Santander cycle is a bike sharing

² <https://whimapp.com/uk/>

³ http://smile-einfachmobil.at/index_mobile.html

⁴ <https://www.moovel.com/en>

⁵ <https://www.blue-city.co.uk/>

scheme promoted by Transport for London (TfL), where you can hire, ride and return a bike into any docking station within the city of London.

2.5.3 Challenges and future perspectives

Despite of the many sophisticated mechanisms employed to improve urban mobility, road transport still constitutes an environmental, social and economic issue to modern cities. In 2016, passenger cars accounted for 82.9% of road transport passengers in the EU, with other modes accounting for less than a tenth of all traffic (European Commission, 2019b). It is expected that road transport activity will continue to grow, with road passenger transport being projected to increase by 30% in 2050. In 2017, commuters in Moscow and London spent an average of 91 and 74 hours respectively in peak traffic delays (Cookson and Pishue, 2017). In fact, drivers in the UK lost an average of 178 hours due to congestion, costing the country £7.9 billion, with London alone contributing to £4.9 billion (Reed and Kidd, 2019). These increases in traffic lead to more potential conflicts between drivers, and therefore, a heightened likelihood of traffic incidents. The EU aimed to reduce traffic fatalities by 50% by 2020 compared to 2010, which would correspond to a rate of less than 3.1 fatalities per 100,000 habitants. However, in recent years, it has been concluded that the target will be missed. As of 2018, road traffic fatalities have decreased by 21% from 2010, with a rate of 4.9 road deaths per 100,000 habitants (European Commission, 2019c). Compared to their national average, cities are actually scoring much better in terms of traffic safety, with almost all recording lower fatality rates. Nevertheless, more concrete and swift actions are needed to make a big leap towards meeting future targets.

Shared mobility initiatives can potentially contribute in modifying the demand of road transport activities. However, the situation is starting to get more complex. Alonso Raposo *et al.* (2019) explains that instead of eliminating vehicles from the road, these new initiatives could actually attract people from other modes of transport. For instance, carpooling systems introduced at low prices can attract cyclists, pedestrians (Le Vine, Adamou and Polak, 2014; Hoadley, 2018) and users from public transport (Barrios, Hochberg and Yi, 2019). This can have a harmful effect in the level of services of other modes of transport, due to reduced income, possibly causing a shift to passenger cars. A similar situation has been discussed about other shared mobility initiatives such as car sharing and on-demand services. A study in the US found that approximately 55% of ride-hailing users (e.g. Uber, Lyft) would have either used public transportation, cycled or walked if the service had not been available (Clewlow and Mishra, 2017). This increases the demand of vehicle needed to satisfy the demand, and therefore, creates greater pressure on the transport network. Connected automated vehicles (CAVs), connected and autonomous vehicles and driver-less cars, have been studied as a solution for changing people's activities and travel behaviours. Automated shuttle bus services and ridesharing with automated vehicles can have a great impact on travel costs, which would potentially impact traffic demand and vehicle ownership. In fact, Wadud, MacKenzie and Leiby (2016) estimated that they can cut the cost of travel by 80%, with a significantly lower number of vehicles on the roads. Many studies in the literature have showcased the positive effects of CAVs with regards of reduction of accidents, improving transportation efficiency and lowering the number of cars owned (Bajpai, 2016; Talebpour and Mahmassani, 2016; Ye and Yamamoto,

2018). However, this technology is still on its early stages, and it is expected to be available on the mass market not earlier than 2025 (Elliott, Keen and Miao, 2019).

These potential increases in road transport provide a clear indication of the importance of employing advance congestion control mechanisms in modern cities. Unfortunately, existing traffic management systems are still unable to provide satisfactory and accurate monitoring of the transport network (Zanella *et al.*, 2014; Djahel *et al.*, 2015; De Souza *et al.*, 2017). These systems still suffer from a lack of traffic parameters accuracy and struggle to report real-time traffic events. Therefore, new implementations and mechanisms have been proposed as solutions towards modern TMS in a smart city environment (Masek *et al.*, 2016). A modern TMS should take advantage of emerging technologies to overcome the limitations mentioned above. According to Djahel *et al.* (2015), a TMS for future smart cities should meet the following requirements:

- Ensure higher accuracy in detecting traffic conditions and better efficiency in dealing with emergency situations.
- Provide real-time road traffic simulation and visualisation.
- Integrate data of existing systems and new technologies.
- Manage road networks of different size and characteristics.

One of the key principles of a TMS for a smart city is to take advantage of heterogeneous data sources to improve incident detection and traffic prediction tasks. To this end, new efforts should focus on integrating information from historical traffic data, real-time traffic data and social media feeds. De Souza *et al.* (2017) argues that incorporating data from publications in social media such as Facebook, Twitter,

Instagram and Foursquare, can directly influence the traffic efficiency. These data can be combined to traffic characteristics, and congested areas can be forecasted based on the social media check-ins based on the time and day of the week. Moreover, social media data can also be employed to identify issues that are directly affecting the transport network. In contrast, Djahel *et al.* (2015) recommended the integration of traffic simulation to infer the impact of random traffic events on the transport network, thus more informed decisions can be taken in case of a real incident. Similarly, Hadi, Sinha and Wang (2007) highlighted the importance of evaluating the different incident management alternatives in traffic simulation programs, as these tools can properly model the reductions on highways capacity due to incidents and lane changing behaviours of the drivers. For instance, Barceló *et al.* (2002) evaluated the impact of traffic incidents and roadworks by simulating the effectiveness of different traffic management strategies in urban and inter-urban areas. They demonstrated the efficiency of traffic simulation programs in finding the optimal solution as what are the best lanes to close in case of an accident.

2.6 Research gaps

The review of the literature presented in this chapter have identified the following gaps:

- Existing sensor technologies not always provide accurate traffic measurements and do not cover broad areas of the transport network. As a result, the performance of existing TMS is influenced by missing traffic data due to sensor malfunction.

- Existing imputation methodologies for missing traffic data work well on highways and have not been validated in arterial roads.
- Smart traffic management systems aim to use a combination of simulation and modelling tools and to exploit data from heterogeneous sources.

2.7 Summary

In this chapter the general components of a TMS are discussed. The performance of the different sensor technologies play an important role into the accuracy of existing TMS. These technologies have evolved over the years, from traditional sensor technologies (e.g. loop detectors) to more recent advancements (e.g. cellular geolocation systems). The traffic parameters collected by these sensors are then fed to sophisticated algorithms for the detection and prediction of the traffic conditions. However, literature has shown that existing sensor technologies still fail to provide accurate measurements and do not fully cover the transportation network. As a result, existing AID and traffic prediction systems suffer from high false alarm rate and have not been validated in arterial roads. Moreover, this issue also impacts the credibility of a simulation model as they highly depend on the validity of the traffic data. While there has been a considerable amount of research into the imputation of missing traffic data for highways, their validity on arterial roads is yet to be confirmed. Recently, the smart city paradigm has caught the attention of the research community. A future smart city should take advantage of smart devices, in-vehicle technologies and social media data to provide a real-time integrated mobility service. Hereafter, authors in the literature have suggested the incorporation of social media data and real-time simulations of the

transport network as a solution to the current issues faced by TMS. Thus, the next chapter is focused on the potential of social media data for transportation.

Chapter 3: Social media for transportation

3.1 Introduction

Since their introduction in 1997, social media sites have attracted millions of users, many of them using them on a daily basis. Social media sites are computer-mediated communication technologies that allow people to connect, interact, produce and share content on real-time (Lewis, 2009). Based on how users interact with them, social media sites have been categorised into social networking, microblogging, photo sharing and video sharing sites. Figure 3-1 ranks the most popular platforms in 2018, based on number of active users in millions.

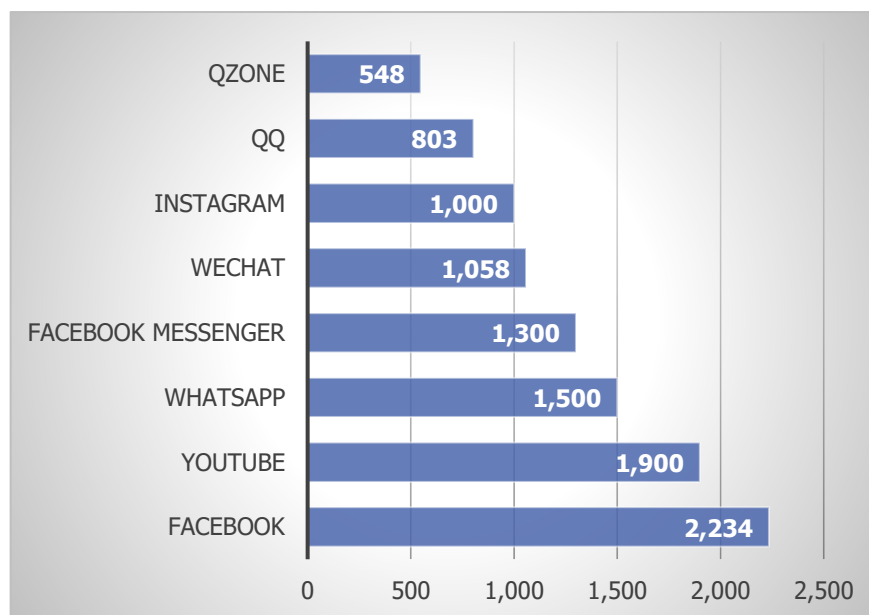


Figure 3-1: Global social media applications in 2018, ranked by number of users (millions)

Facebook is the leading social networking site worldwide with 2,234 million of active users. Facebook allows its users to share a wide range of information: photos, videos, status updates and news. However, it is Twitter that has been the focus of study in the last few years. Twitter is the most popular microblogging site with 335 million of

active users posting around 500 million tweets per day. Twitter enables users to post short messages (tweets) of no more than 280 characters. While some users consider the 280-character limit as a constraint, Atefeh and Khreich (2015) argue that this is precisely what sets Twitter apart as short information is easier to consume and faster to spread. People use Twitter as a way of expressing their views in different matters, or to comment about real-time events happening around them. As a result, several authors have studied the influence of Twitter at predicting real-world outcomes such as revenues, accidents, natural disasters and even traffic.

This chapter outlines the potential of Twitter for traffic incident detection. Section 3.2 starts with a brief introduction of the main features of Twitter, along with a review of the different methodologies employed for event detection from Twitter streams. This section also explores the potential of social media sources for opinion mining and decision making. Section 3.3 reviews existing literature for traffic incident detection using Twitter streams and the potential of such methodologies to improve existing TMS. Finally, section 3.4 presents the challenges associated with exploiting Twitter data for traffic incident related purposes.

3.2 Twitter as a sensor

Microblogging is a compact form of blogging that has become popular in the last decade. The main feature of a microblog is the limitation it sets to its users about posting short pieces of text. Compared to a traditional blog, information in microblogs is faster to process and easier to spread, due to its short nature. Twitter, one of the most popular microblogging services, was created in 2006 and by 2012 more than 100 million users were posting 340 million tweets a day. People use Twitter not only for

chatting and communicating, but also for discussing real-time events as they happen or shortly after. For this reason, many authors have exploited Twitter data for detecting and tracking events in real-time. This section starts with a brief introduction on Twitter properties that are relevant for this research (in section 3.2.1 and section 3.2.2), and it then discusses different methodologies for processing real-time Twitter streams for event detection (section 3.2.3). Finally, the potential of Twitter data for sentiment analysis is presented in section 3.2.4.

3.2.1 Characterisation of Twitter

Twitter is a popular microblogging site that allows its users to post and interact through short messages called 'tweets'. Originally, tweets were restricted to 140 characters, but as of November 2017 this limit was doubled to 280 characters for all languages except Chinese, Japanese and Korean (Rosen, 2017). Apart from the web interface, users can access Twitter through its mobile application or through Short Message Service (SMS). By default, all accounts are public which means that anyone can see the content of an account, even if they do not have a Twitter profile. Users have the option to follow others, in order to see their real-time updates in their 'timeline' (Twitter's main web interface), as shown in Figure 3-2. However, it is possible to set privacy settings that would only allow accepted users ('followers') to see updates. As a result, Twitter is comprised by three different categories of followers and followees: broadcasters (newspapers, celebrities, bloggers), acquaintances and spammers.

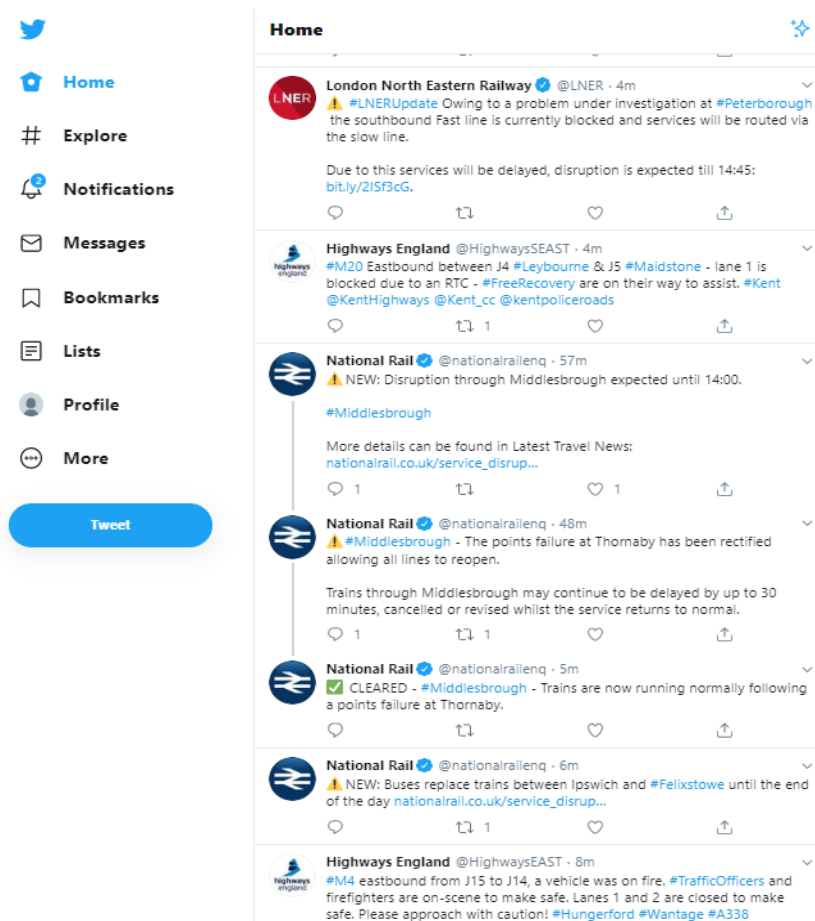


Figure 3-2: Example of a Twitter timeline

Due to its conversational characteristics, Twitter have been often referred as the 'SMS of the Internet' (Wang, 2016). The interaction amongst users is supported through a universal tweeting system that uses three basic symbols: mentions (@), retweets (RT) and hashtags (#). Users can mention or reply to others by using the character '@' followed by the account name they would like to mention. It is also possible to share the tweet of another user, by 'retweeting' the tweet with their followers. A 'RT' prefix is used in front of the original tweet, followed by the username of the person/entity that created the message. A hashtag (#) is the topic indicator of Twitter, and it allows users to initiate and participate in specific topics by adding a '#' before a word/phrase. This is designed with the purpose of grouping similar tweets to allow users to retrieve

a specific topic with just one click. Some examples of the interaction in Twitter can be perceived in Figure 3-2.

The above-mentioned aspects reveal the great degree of importance that Twitter content has. While a large majority of interactions are amongst acquaintances, it has been shown that another great proportion of users with similar interests are very likely to be connected (Java *et al.*, 2007). As a result, people tend to talk about their daily activities, discuss their views in different topics and seek real-time information. The latter, mainly because the hashtag feature facilitates topic tracking, making it easier for users to find posts/news of interest. In fact, according to Kwak *et al.* (2010), approximately 85% of topics on Twitter are headline news or persistent news in nature. In view of the interesting one-sided relationship, the topic tracking feature through hashtags, the retweet mechanism that allows fast spread of information, and on top of that easy and free to crawl data through the Twitter API (see section 3.2.2), Twitter has caught the early attention of the research community as a way of forecasting sales (Asur and Huberman, 2010), polls (Tumasjan *et al.*, 2010), events (Sakaki, Okazaki and Matsuo, 2010) and even diseases (Aramaki, Maskawa and Morita, 2011).

3.2.2 Accessing Twitter data

Twitter offers two public Application Programming Interfaces (API) to developers and researchers for collecting Twitter data: Search API and Streaming API. For non-paying developers, it is not possible to retrieve all Twitter data as different rate limits restrict the access to both APIs. The Search API⁶ returns a collection of relevant tweets

⁶ <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets>

matching a specific query. The query format may include location (latitude, longitude, radius), language and keywords. However, this tool is not meant to be a comprehensive source of tweets, as not all Tweets are made available via the search interface. Moreover, on its free-access version (standard), Twitter restricts the queries to a sampling of recent Tweets published in the past 7 days with a limit of 180 requests every 15-minute interval.

The Twitter Streaming API⁷ offers a real-time stream of the public tweets through an uninterrupted connection. With the standard version (statuses/filter), it is only possible to obtain up to 1% of the volume of tweets per streaming second (firehose). In addition, while it is possible to query by different parameters (e.g. keyword, location, language), the standard version only allows one filter rule on one allowed connection. The number of filters that can be tracked is also limited to 400 keywords, 5000 user ids and 25 location boxes for a single query. On the other hand, the PowerTrack is the enterprise edition of the streaming API, that allows to filter with more than one rule, and provides full access to the firehose. Table 3-1 contains information on the different filtering capabilities, rate-limits and rules of both the standard and enterprise edition of the Streaming API.

Table 3-1: Streaming API categories

API	Category	Number of filters	Rules	Limit
Statuses/Filter	Standard	<ul style="list-style-type: none"> - 400 keywords - 5000 user ids - 25 location boxes 	One filter rule on one allowed connection	1% of the firehose
PowerTrack	Enterprise	Up to 250,000 filters,	Thousands of rules on a single connection	Full firehose

⁷ <https://developer.twitter.com/en/docs/tweets/filter-realtime/overview>

The most relevant filtering parameters allowed by the Streaming API are shown in Table 3-2. For keyword tracking, Twitter uses a fuzzy matching mechanism instead of exact matching. For instance, when using 'Twitter' as a keyword, the engine will match it with tweets that contain the term in lower case and upper case, as well as those with quotation marks, hashtags, mentions and URLs, as shown in

Table 3-3. When tracking by location, it is necessary to provide a list of longitude and latitude pairs for specifying a set of bounding boxes to filter the tweets. Contrary to the Search API, the Streaming API will only include geolocated Tweets falling within the bounding box, without taking into consideration the user's profile location. Table 3-4 presents some bounding box examples for location tracking within the Streaming API.

Table 3-2: Different filtering parameters of the Streaming API

Name	Description
Follow	A comma separated list of user IDs, indicating the users to return statuses for in the stream.
Tracks	Keywords to track. Phrases of keywords are specified by a comma-separated list.
Locations	Specifies a set of bounding boxes to track.
Delimited	Specifies whether messages should be length-delimited.
Language	Returns tweets in a specific language

Table 3-3: Twitter API keyword tracking examples

Parameter value	Will match	Will not match
Twitter	TWITTERtwitter "Twitter" twitter. #twitter @twitter http://twitter.com	TwitterTracker#newtwitter
Twitter's	I like Twitter's new design	Someday I'd like to visit @Twitter's office
Twitter api, twitter streaming	The Twitter API is awesomeThe twitter streaming service is fast Twitter has a streaming API	I'm new to Twitter

Table 3-4: Twitter API location tracking example

Parameter value	Track tweets from
-122.75, 36.8, -121.75, 37.8	San Francisco
-74, 40, -73, 41	New York City
-122.75, 36.8, -121.75, 37.8, -74, 40, -73, 41	San Francisco OR New York City

Tweets from the Twitter APIs are delivered in JSON⁸ format. JSON (JavaScript Object Notation) is a lightweight-data format that implements human-readable text consisting on key-value pairs, with named attributes and associated values. The JSON document contains different attributes of the tweet such as author, text, coordinates, timestamp, a unique ID and sometimes even geo metadata shared by the user. It can also contain the retweet status of the tweet. As a result, in addition to the text content itself, a tweet can have over 150 attributes associated with it. Figure 3-3 illustrates the structure of a typical tweet in JSON format.

```
{
  "created_at": "Thu Apr 06 15:24:15 +0000 2017",
  "id_str": "850006245121695744",
  "text": "1\ Today we\u2019re sharing our vision for the future of the Twitter API platform!\nhttps://t.co/XweGngmx1P",
  "user": {
    "id": 2244994945,
    "name": "Twitter Dev",
    "screen_name": "TwitterDev",
    "location": "Internet",
    "url": "https://dev.twitter.com/",
    "description": "Your official source for Twitter Platform news, updates & events. Need technical help?
    \n\nVisit https://twittercommunity.com/ \u2328\u201c\u201c #TapIntoTwitter"
  },
  "place": {
  },
  "entities": {
    "hashtags": [
    ],
    "urls": [
      {
        "url": "https://t.co/XweGngmx1P",
        "unwound": {
          "url": "https://cards.twitter.com/cards/18ce53wgo4h/3xo1c",
          "title": "Building the Future of the Twitter API Platform"
        }
      }
    ],
    "user_mentions": [
    ]
  }
}
```

Source: Twitter Developer

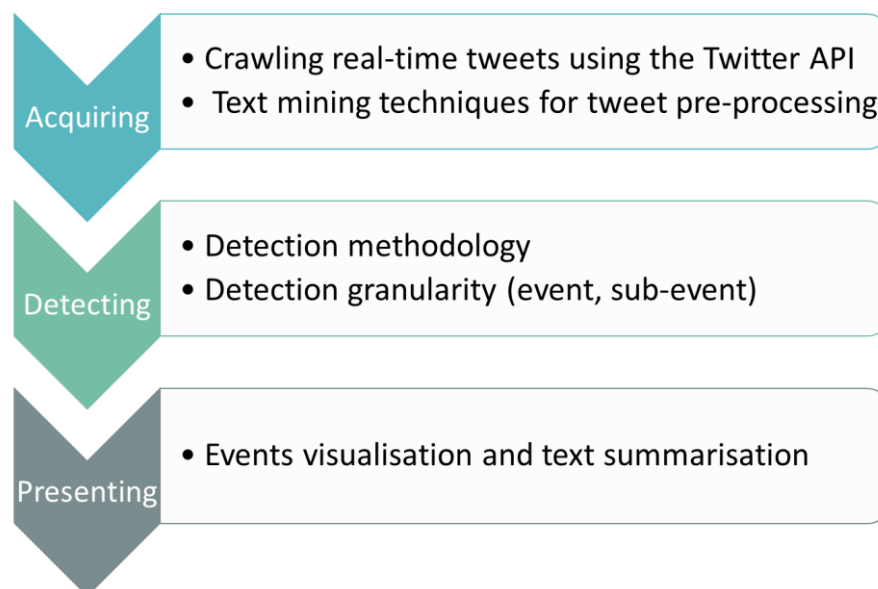
Figure 3-3: Structure of a tweet in JSON format

3.2.3 Twitter event detection methodologies

Most people witnessing an event can disseminate details of its occurrence in real-time using microblogs (Atefeh and Khreich, 2015). People turn to Twitter to tweet about

⁸<https://www.json.org/>

specific events as they happen, or shortly after. This makes Twitter data a valuable source of information regarding incidents that differ significantly by type, location, and time. However, contrarily to traditional websites (e.g. news websites and blogs), the raw Twitter stream provides large amounts of unstructured and informal text. For this reason, research attempts in the literature are aimed at developing methodologies for extracting, processing and detecting useful information from a massive volume of Twitter streams. Figure 3-4 illustrates a generic workflow of the stages observed in the different event detection methodologies in the literature.



Adapted from Wang (2016)

Figure 3-4: Generic event detection methodology from Twitter streams

While event detection methodologies in the literature employed different text mining techniques and algorithms for processing Twitter data, Wang (2016) concluded that the production of event information from tweets is achieved by three main stages: **acquiring** real-time data, analysing the data for **detecting** events, and synthesising the detected results for **presenting** event summarisation to users (as seen in Figure 3-4). The following sub-sections provide a detailed review of the different event detection methodology stages associated with relevant systems found in the literature.

3.2.3.1 Acquiring: Retrieving and pre-processing tweets

With more than 500 million tweets per day (Twitter, 2016), it is difficult to detect useful information from noise (e.g., non-traffic related, false reporting, spam). In fact, Analytics (2009) concluded that 40% of the tweets could be considered as 'pointless babble'. As a result, the acquisition stage entails identifying and crawling only those tweets that are relevant to the events of interest. As mentioned in Section 3.2.2, Twitter provides two different APIs for developers and researchers to access twitter streams: the search API and streaming API. The latter, being the most suitable one for supporting real-time applications.

Some studies in the literature crawled real-time tweets from the streaming API without using any keyword or location criteria (Petrović, Osborne and Lavrenko, 2010; Abel *et al.*, 2012; Krstajic *et al.*, 2012). These studies included in their dataset almost every type of tweets, such as breaking news, advertisements and users' daily timeline activity. These datasets tend to be more vulnerable to noise and reaching Twitter API limits. There are other researchers with more specific requirements, where their retrieval is achieved by using some specific criteria (e.g. keywords, location). This is the most common way of acquiring tweets for event detection, although it could lead to losing some meaningful information. The most common approach is keyword and hashtag filtering, and it has been widely adapted by authors in the literature (Sakaki, Okazaki and Matsuo, 2010; Starbird and Palen, 2012; Nichols, Mahmud and Drews, 2012). For instance, Sakaki, Okazaki and Matsuo (2010) retrieved emergency related tweets by using the terms 'earthquake', 'shake' and 'typhoon' as filters. Nichols, Mahmud and Drews (2012) collected sports related tweets using 'worldcup' and 'wc2010'. On the

other hand, location filtering is another popular property employed for tweet crawling by various authors in the literature (Abdelhaq, Sengstock and Gertz, 2013; Xie *et al.*, 2016). It tends to be used mainly when the research concentrates in local events, rather than world news.

Due to the short and informal nature of a tweet, the acquired data needs cleaning before proceeding to the next stage. The pre-processing step consists of employing text processing techniques to represent the data in a manner that could be efficiently analysed by the detection stage. The most common pre-processing techniques implemented in the literature are:

- **Tokenisation:** Tokenisation is usually the first pre-processing step, and it consists of splitting the text into a set of tokens (e.g. words, syllables, phrases). Transforming a tweet into a set of tokens is usually done by using spaces between words as a separator. In addition to this, authors in the literature also take this step to remove punctuation, non-text characters and symbols.
- **Stop word elimination:** There are some words that are too frequent to be considered a useful feature to characterise a text, as they do not provide any semantic significance to the texts or phrases they appear in. For instance, the verb 'be' or the article 'the' can be found in almost all documents, without giving any significance to them. These words are called stop words and are usually removed from the dataset. Some authors also remove abbreviations often used in short informal messages. For instance, Alsaedi (2017) also removed common abbreviations found in tweets, such as 'omg', 'lol' and 'lmao'.

- **Stemming:** Stemming is a pre-processing step consisting of reducing each work to its stem or root form. It usually implies removing suffixes and prefixes attached to a word. Several stemming processes have been employed as a pre-processing step in event detection technologies in the literature (Sakaki, Okazaki and Matsuo, 2010; Agarwal, P. *et al.*, 2012; Lampos, 2012; Alsaedi, 2017), most of them using suffixing stripping such as the Porter Stemmer methodology (Porter, 1980; Porter, 2001).

3.2.3.2 Detecting events from twitter data

After crawling and pre-processing tweets, the next stage involves detecting meaningful information from them. Developing an accurate Twitter event detection algorithm is perhaps one of the most explored topics of research in the last decade. Depending on the systems specifications, some authors would also develop methodologies for identifying the location of the event. These methodologies could be based on machine learning algorithms or natural language processing (NLP) techniques. Sakaki, Okazaki and Matsuo (2010) were amongst the first to propose a methodology to detect events using Twitter. They were able to detect earthquakes with a 96% probability by using a Support Vector Machine (SVM) for classification, and a Kalman filtering for location estimation. Li *et al.* (2012) employed a Passive-Aggressive algorithm as a classifier, and NLP techniques to identify the location of the event from the keywords within the tweet. Osborne *et al.* (2014) implemented a supervised Bayesian model for identifying security-related events such as violent events, natural disasters, or emergency situations. Their model achieved 85.8% of accuracy in detecting such events.

Some researchers have explored other machine learning techniques that monitor the frequency of individual and multiple features in a set of tweets, in order to achieve better event detection results. For instance, Krstajic *et al.* (2012) detected potential events by monitoring the frequency of individual keywords and for those with unexpected high frequency values, it calculated additional scores that could help on describing the event. The most common employed features in the literature are co-occurrence of terms (Popescu and Pennacchiotti, 2010), frequency of terms (Weng and Lee, 2011; Asakura *et al.*, 2017) and tweet metadata (Psallidas *et al.*, 2013; Abhik and Toshniwal, 2013). Similarly, burst detection methodologies have been widely implemented in the literature. They derive from the idea that people tend to show a sudden interest in posting and forwarding tweets of breaking news and events. As a result, the frequency of a term being used will deviate from a normal value and the algorithm will detect an event from this burst of features (Wang, 2016). For example, Xie *et al.* (2016) proposed a framework for topic detection by monitoring the occurrence of each word pair and each word triple. The bursty topic inference would be triggered based on indicators of a potential surge of tweet popularity. More recently, Comito, Forestiero and Pizzuti (2019) proposed an algorithm that monitored a number of textual and temporal features to group similar tweets, continuously generated over time, into clusters. Results from the experimental implementation revealed that the method was capable of detecting events revealed by a sudden burst of attention on certain topics.

3.2.3.3 Presenting event summarisation

The last component often found in event detection methodologies corresponds to presenting the detected events. The overall objective of this stage is to present a visual summarisation of the different events detected by the system. Some researchers present the results from the detection algorithms without further processing, by using word clouds, maps and charts (Wanner *et al.*, 2014). However, due to the informal and short nature of tweets, these outputs could be hard to interpret. As a result, it is common practice to summarise the information and present the most important event information. This can be achieved by presenting a small group of key phrases or limited number of tweets. One of the most commonly employed mechanism for describing an event using key terms is based on the term frequency value. For instance, some researchers group the set of terms which frequency bursts appear correlated to describe the events (Xie *et al.*, 2016; Comito, Forestiero and Pizzuti, 2019). Other researchers in the literature find that a list of key terms may lose some of the context, thus they select the most descriptive and informative tweets as an output. This is achieved by calculating the most influential tweet using the sum of term weight or the normalised average of the term frequency score (Shen *et al.*, 2013). However, it has been concluded that term summarisation produces most promising performances (Wang, 2016).

3.2.4 Detecting sentiment from Twitter streams

Microblogging websites are a valuable source for opinion mining and sentiment analysis as users tend to share their opinions on different subjects (Pak and Paroubek, 2010). People use social media to comment and complain about events and problems of their

social life. For this reason, researchers have taken interest on Twitter due to the diversity of its topics and the wide amount of opiated data. Sentiment analysis is the area of NLP that studies opinions, sentiments, and emotions from the written language (Liu, 2012). In recent years, a lot of work has been done in the field of sentiment analysis from Twitter data. In the early stages, sentiment analysis applications were intended for binary classification only (e.g. positive and negative). Agarwal *et al.* (2011) were amongst the first to introduce a 3-way model for classifying sentiment into positive, negative and neutral classes. They experimented with models based on unigrams and tree kernels and concluded that sentiment analysis for Twitter data is not that different as for other genres. Initially introduced in 2010, Thelwall *et al.* proposed a sentiment strength detection algorithm called SentiStrength (Thelwall *et al.*, 2010; Thelwall, Buckley and Paltoglou, 2012; Thelwall, 2017). Contrary to existing sentiment analysis algorithms, SentiStrength detects the strength of the emotion expressed in short texts, based on psychology of emotion research which suggests sentiment as two separately measurable positive and negative components. It employs two scales as its main purpose is to detect sentiment, rather than expressing an overall polarity.

Detecting sentiment from Twitter streams can help understanding how people feel about certain situations (e.g. emergency), events (e.g. elections) and products (services, companies, movies). For example, Garcia Esparza, O'Mahony and Smyth (2012) studied the potential of leveraging social media data, particularly microblogs, for obtaining product recommendation based on users' opinion on certain subjects. They evaluated the relationship between users and products based on the terms used in their associated messages. Carpenter and Way (2012) presented a sentiment

analysis tracking algorithm using Twitter data, and implemented it on a web-based application that could track the sentiment as it changes over time. Results from this type of implementation are a valuable source of information to improve situational awareness during emergency events. In fact, some event detection methodologies include sentiment data in their systems to identify the nature of the event based on the emotions associated with it (Krstajic *et al.*, 2012; Osborne *et al.*, 2014).

Many authors have studied what is called 'participatory sensing'. Participatory sensing is the concept that support civic engagement, in which citizens can bring to light a civic bottleneck, hazard, personal-safety concern, cultural asset, or other data relevant to urban planning. For instance, Lin *et al.* (2006) presented a system to automatically detect attitude and moods for social media to provide support for decision makers. They developed statistical models to capture the perspectives expressed in short sentences and evaluated their model on political articles. Stylios *et al.* (2010) presented a method for capturing the public's opinion about governmental decisions from social media. Their objective was to be able to identify the public view on specific governmental decisions, thus it could be taken into consideration in subsequent regulations.

3.3 Twitter based transportation research

Intelligent Transport Systems (ITS) is another area where a considerable amount of literature has grown around the influence of user-based content into traffic information. Many users turn to Twitter to report traffic incidents, describe the traffic situation they are currently in, or to receive information from official Twitter traffic management accounts. This makes twitter data a real-time source of human travel information.

Using this data for traffic incident detection can overcome some of the previously stated issues faced by conventional sensor technologies. Firstly, there is no cost involved as Twitter grants free access to a subset of its data (See section 3.2.2). Secondly, as traditional sensors only detect changes in traffic measures, they are unable to identify the traffic event taking place, or to obtain feedback from the users. Moreover, sensor failures and communication errors are common problem in traffic management operations. A tweet usually contains a more detailed information about the traffic incident (e.g. accident, roadworks, social event), and it can provide an insight into the user's perception about the status of the transport network. Lastly, users can tweet from any location, covering broader areas of the network.

He et al. (2013) were one of the first to incorporate Twitter data into traffic forecasting. By comparing linear regression using traffic flow against the one that included social media, they were able to demonstrate the effectiveness of integrating Twitter data on traffic prediction. Another way of incorporating social media was by creating a system architecture based on social media information. To that extent, Wibisono *et al.* (2012) developed an intelligent system capable of obtaining traffic information data from the Twitter accounts of official channels and disseminating this information to the users in their mobile devices, using neural networks.

In the last decade, many researchers have explored the use of twitter data as a transportation sensor for traffic event detection. The detection of traffic incidents from Twitter streams follows the same structure of as event detection methodologies previously discussed (see section 3.2.3). For instance, Wanichayapong *et al.* (2011) implemented a unique traffic word dictionary-tokeniser of four categories: place, verb,

ban and preposition. After crawling, tokenising and filtering the tweets, it then geolocated the tweets using latitude and longitude from the place dictionary and Google geocoding. Their system was based on NLP techniques. Gutierrez *et al.* (2015) described an approach for integrating tweets from different traffic agencies in the UK, with the purpose of notifying drivers about the status of the network in real-time. Schulz, Ristoski and Paulheim (2013) presented a methodology for the identification of small scale incidents by combining text classification techniques with a machine learning algorithm. They identified car crashes with an accuracy of 89% using a SVM. D'Andrea *et al.* (2015) filtered tweets by traffic related keywords, and compared different machine learning algorithms for the classification task. The SVM outperformed others with a 95.75% precision. Gu, Qian and Chen (2016) used a methodology to detect traffic informing tweets using a Semi-Naïve-Bayes classifier, with an overall accuracy of 90.5%. They compared historical Twitter data with existing incidents reports and concluded that a small sample of tweets covered most of the incidents reported. Zhang *et al.* (2018) employed a deep learning technique to detect traffic related tweets, and achieved an accuracy of 85%. They found that Twitter data was able to identify nearly 66% of traffic incidents, with more than 80% of them being tied to abnormal traffic data.

Traffic-related tweets tend to be filled with emotions as users usually complain about the state of the network or express their stress and concern about a specific traffic incident. In fact, it has been concluded that driving is the most stressful mode of transport (Wener and Evans, 2011). By detecting the level of stress within a traffic related tweet, agencies can identify the factors causing non-recurrent congestions. It

is important to include this subjective data into traffic incident detection, as it can give a better understanding of the users' perception about the status of the transport network, not only as a whole but also for specific routes (Kokkinogenis et al., 2015). Thelwall (2017) proposed TensiStrength, an effective method to detect stress from short informal text. This study argues that although stress is indirectly taken into consideration in ITS systems (e.g. rush hours, stressful journeys), it is not possible to predict how accidents or random traffic jams would affect the travellers. Apart from this study, there is a general lack of research in incorporating user emotions in the transportation domain.

Table 3-5 contains a summary of the studies mentioned above for identifying traffic events using Twitter, along with the properties that a modern traffic event detection technique should satisfy. An approach is considered to be in real-time if it employed the Streaming API to collect tweets as they happened, instead of using the Search API. Mainly, because the Search API returns only relevant tweets rather than the real-time tweet feed. Geolocation is a very important step in traffic incident detection. While traditional event detection techniques not necessarily include this stage, it is crucial to know the exact location of the traffic event taking place (e.g. road, junction). Another factor considered important is if the system is user centred. This research considered an approach as user-centred if the dataset was based in tweets from road users, rather than newspapers or highways agencies. Finally, the inclusion of user sentiment is also compared in this table, as none of the existing research incorporate user emotions into their methodologies. It can be perceived that none of the most relevant work presented in the literature took into consideration all these properties. Indeed,

D'Andrea *et al.* achieved the highest accuracy in the literature (95.75%), but their methodology did not identify the location of the tweets or included sentiment analysis. The work presented on this thesis aims to satisfy all the properties presented in Table 3-5.

Table 3-5: Summary of traffic incident detection techniques using Twitter data

Authors	Real-time	Geolocation	User-centred	Sentiment analysis	Accuracy
Wanichayapong <i>et al.</i> (2011)		x	x		76.85%
Schulz, Ristoski and Paulheim (2013)		x	x		89%
Gutierrez <i>et al.</i> (2015)		x			95.5%
D'Andrea <i>et al.</i> (2015)	x		x		95.75%
Gu, Qian and Chen (2016)		x	x		90.5%
Zhang <i>et al.</i> (2018)	x		x		85%
This work	x	x	x	x	

3.4 Challenges of using Twitter data for traffic event detection

Authors in the literature have identified four main challenges when detecting traffic events from twitter streams (Gutierrez *et al.*, 2015; D'Andrea *et al.*, 2015; Gu, Qian and Chen, 2016; Zhang *et al.*, 2018):

- Unpredictable volume of data
- Veracity of the information detected
- Tweet geolocation
- Twitter API limits

The first challenge is related to the massive and unpredictable volume of data. It is very difficult to detect useful information from over 500 million of real-time streams of information. Twitter data usually contains high amounts of not useful or meaningless information (e.g., non-traffic related, false reporting, spam). Moreover, due to the short nature of a tweet, traditional text mining techniques do not work well on them, as they often contain emoticons, typos, grammatical errors and improper sentence structures (Gu, Qian and Chen, 2016). As a result, efficient and complex approaches are needed to handle the vast amount and uncertainty of Twitter data such as the NLP techniques described in section 3.2.3.1.

Another challenge refers to the veracity of the traffic events detected from Twitter streams. Much of the content found on Twitter could be referring to an event that is not necessarily happening at the current instant. Many studies have concluded that people are motivated to share real information by the desire to help others (Ghaisani, Handayani and Munajat, 2017), but not all travellers tweet when they are passing a specific incident for their own traffic safety. As a consequence, there is the possibility that the tweet has been posted when the event has already finished (Zhang *et al.*, 2018). It is also probable that some of these event-related tweets are false alarms.

Perhaps one of the biggest challenges found in the literature is the geolocation of tweets. Only a small portion of tweets are geotagged, and this location does not necessarily represent the actual location of the event. In addition, GPS errors could cause inaccuracy in mapping these events. Therefore, different authors have proposed text mining techniques, such as Named Entity Recognition (NER) (Wanichayapong *et al.*, 2011; Schulz, Ristoski and Paulheim, 2013; Gutierrez *et al.*, 2015), to identify the

location from the tweet text. However, when using short and informal messages, one concept could refer to more than one place or the description might not be enough to link the text to an actual location.

Lastly, there is the rate limit imposed by the Twitter APIs. As described in section 3.2.2, Twitter offers free access to its data through two different types of API: Search API and Streaming API. However, there is a limit on the number of tweets that can be obtained in real-time. On the Search API, applications can make 180 queries every 15 minutes. On the other hand, every Twitter account can obtain up to 1% of the volume of tweets per streaming second, using the Streaming API. Unlike the Streaming API, the Search API allows to filter by location and keyword, being this the main reason that has led researchers in the literature to use the Search API for their studies (Wanichayapong *et al.*, 2011; Schulz, Ristoski and Paulheim, 2013; Gutierrez *et al.*, 2015; Gu, Qian and Chen, 2016). However, the Streaming API has the advantage of receiving tweets as they are posted on real-time. These limitations, with regards to tweets sampling and keyword filtering, can pose a challenge in obtaining the already limited amount of traffic tweets available. Mainly, because when there is a peak in the volume of tweets per second, which can easily be caused by random events (e.g. elections, world cup), the sample of tweets obtained would be influenced by these events. This would result in obtaining a low amount of traffic tweets, and a significant amount of noise.

Due to the above-mentioned challenges, mainly with regards to geolocation and limits imposed by Twitter API's, it can be concluded that it is unlikely to cover all traffic events using Twitter (Schulz, Ristoski and Paulheim, 2013; Zhang, *et al.*, 2018). Rather than a replacement of the existing AID techniques, the information detected from Twitter

has the potential to supplement their operations, by improving their accuracy and covering areas of the transport networks not equipped with ITS detection technologies.

3.5 Research gaps

From the review presented in this chapter, the following research gaps have been identified:

- Twitter data has proven to be beneficial for many real-world purposes, but hasn't been widely assessed in the context of transportation applications.
- There are still many challenges associated with mining Twitter data, related to text mining and geolocation techniques.
- Opinion mining using social media data is a promising field that can help in obtaining feedback from drivers about optimal routes and the state of the transport network.

3.6 Summary

This chapter reviewed the potential of Twitter data in the transportation domain. Twitter is one of the most popular microblogging services with over 500 million tweets a day. People turn to Twitter to post about real-time events as they happen or shortly after. As a result, many authors have exploited the potential of Twitter data for traffic incident detection. Incident detection techniques from social media can overcome some of the challenges faced by traditional sensor technologies, in terms of cost, coverage and event description. However, there are still many challenges associated with mining Twitter data, with regards of text processing, API limits and geolocation techniques. As a result, traffic incident techniques based on Twitter data aim to enhance conventional methodologies, rather than substituting them. Sentiment

analysis from twitter streams is another field that has caught the interest of the research community. Twitter data is characterised by a high amount of opiated data that can potentially help traffic managers to have a better understanding of the users' perception of the transportation network. Existing incident detection methodologies from Twitter data fail to include this subjective information. Moreover, many of these methodologies have been tested using historical data, rather than real-time data. This research aims to include all the features that a modern Twitter-based traffic incident detection system should satisfy: real-time Twitter feed, geolocation of the incident and sentiment strength detection. The next chapter will present a Smart Traffic Management framework that proposes the incorporation of social media data and real-time simulations of the transport network to existing TMS.

Chapter 4: Smart traffic management framework

4.1 Introduction

In previous chapters, the different sensor technologies and methodologies employed in traffic management systems were discussed. Although many authors have successfully employed a wide variety of traffic prediction and automatic incident detection methodologies, accurate and reliable models for practical use are still not fully functioning. One of the aims of Smart Traffic management is to integrate different data sources and technologies to improve urban mobility. This chapter proposes a smart traffic management framework that integrates real-time simulation of the transport network and social media into existing TMS. The chapter starts by describing the research design methodology selected for this investigation (Section 4.2). Section 4.3 introduces the proposed smart traffic management framework and its main components. To conclude, section 4.4 outlines the selected study area to test the different components of the proposed framework.

4.2 Research methodology

4.2.1 Design science research

As outlined in section 1.4, the methodology selected for this investigation was Design Science Research (DSR). Hevner and Chatterjee (2010) defined design science as a pragmatic research paradigm that creates and evaluates innovative artifacts to solve real-world problems. Design science focuses on the artifact and its relevance on the problem context.

Hevner *et al.* (2004) described the research activities of design science through a clear set of guidelines to conduct and evaluate good design science research. Table 4-1 outlines the DSR guidelines associated to the corresponding phases of the research design of this thesis, as outlined in section 1.4 (Figure 1-3). The main principle behind these guidelines is that, essentially, DSR is a problem-solving process where knowledge and understanding of a problem can be acquired through the creation of an artifact. This artifact should be relevant to the specified problem and its accuracy should be rigorously evaluated. Moreover, this artifact should provide a solution of an unsolved problem, or a more effective way of solving a known problem. The process to create the artifact should incorporate existing theories and knowledge to find a solution to the problem. Finally, the results must be communicated to appropriate audiences.

Table 4-1 Design Science Research guidelines

Phase	Guideline	Description
1	Guideline 1: Problem relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
2	Guideline 2: Research rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
	Guideline 3: Design as an artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
3/4	Guideline 4: Design evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
5	Guideline 5: Research contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.

The output of DSR is manifested through the creation of an artifact, in the form of constructs, models, methods, or an instantiation. March and Smith (1995) defined constructs as a conceptual vocabulary to describe problems of a domain and to specify their solutions. Constructs may be formalised as entities, attributes, relationships or constraints. Models are a set of propositions that express relationship among constructs. Conceptual frameworks are an example of models, where constructs are used to define the structure of the model and its context. Methods are set of steps used to perform a task. This can be represented as algorithms, guidelines or techniques. The final output, instantiation, operationalises the proposed artifact. It is the implementation of the artifact in its context (e.g. products, systems, processes). Instantiation is usually denoted as a material artifact, while the others are referred to as abstract artifacts.

Purao (2002) introduced a framework to recognise the different outputs of DSR as research deliverables and classified them by level of abstraction and generalisation. This framework conceptualises the level of maturity of the different artifacts described before. Figure 4-1 presents an adaptation of the previously described framework. In this figure, the multiple outputs of DSR are differentiated by level of abstraction and in terms of knowledge maturity level, where outputs at higher levels demonstrate more contribution to knowledge. A DSR can produce outputs on one or more of these levels ranging from instantiations at level 1, to more general contributions at Level 2 in the form of nascent design theory (e.g. constructs, models), to well-developed design theories about the phenomena under study.

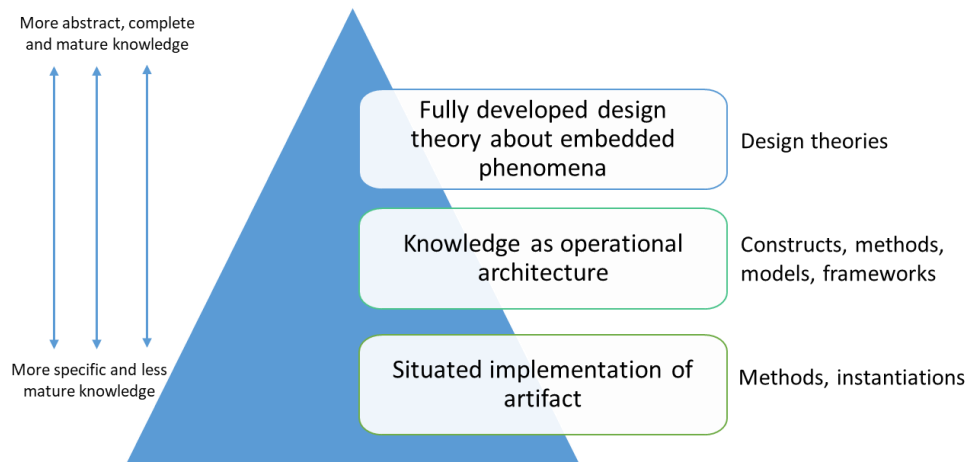


Figure 4-1: Level of maturity of the different DSR outputs

It is important to understand the possible types and levels of knowledge contribution of a DSR. These contributions are related to the problem and solution maturity available in the context of the DSR project. Gregor and Hevner (2013) developed a framework to classify the potential knowledge contributions for DSR in four categories: improvement, invention, adaptation and routine design. In their framework, each of these categories were mapped in a 2x2 matrix depending on their maturity in the solution and application domain (Figure 4-2). An invention entails new and interesting application of research where there is little understanding of the problem, and there are no effective solutions. Most of the research that falls into the invention category are in the form of instantiations. In an improvement, better solutions for known problems are created in the form of more efficient processes and technologies. While most of the previous work in DSR belongs to improvements, the main challenge of this contribution lies in demonstrating that the solution improves previous knowledge. Adaptation research is about extending design knowledge that already exists in new applications or disciplines. In this contribution, the researcher needs to prove that the application of the artifact in the new field is non-trivial and interesting. The last

contribution, routine design, occurs when existing knowledge for both the problem and the possible solutions are well understood. Routine design would rarely be considered as a contribution to knowledge, as it is mainly the application of known solutions to known problems.

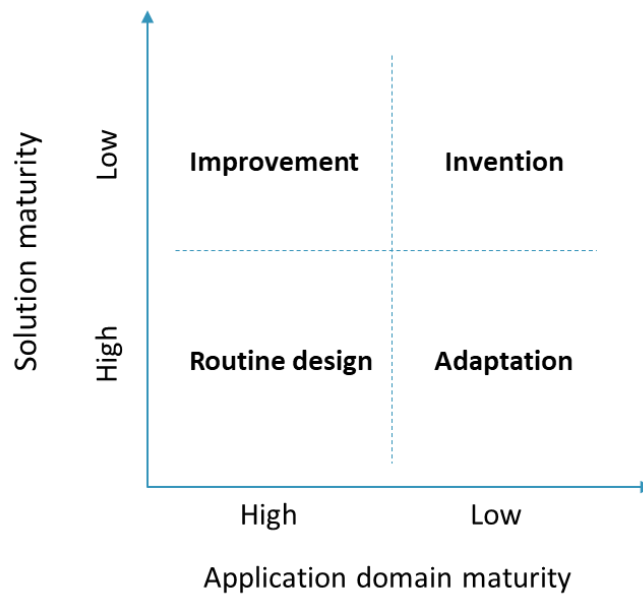


Figure 4-2: DSR Knowledge contribution framework

4.2.2 Systems design methodologies

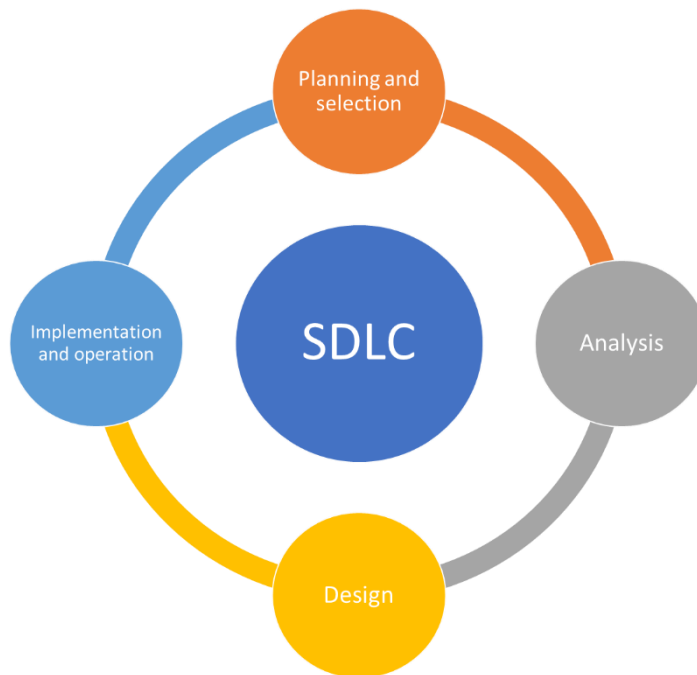
Wieringa (2014) defined architectural frameworks as a collection of elements (systems) that interact with each other to form a 'whole'. Each system can be decomposed into components (sub-systems) that interact to produce overall system behaviour. One of the research goals of this type of structures, is to discover a system architecture based on observations of its behaviour. An architectural framework supports case study research to investigate individual cases (sub-systems), study their architecture and observe how each component performs case by case.

Backlund (2000) defined a system as a set of independent but interrelated elements comprising a unified whole. Most systems share the following characteristics (Avison and Fitzgerald, 2003; Valacich, George and Hoffer, 2013):

- A system is made up of components. A component, also called **subsystem**, is an irreducible part or aggregation of parts.
- A **boundary** establishes the limit of a system, separating it from other systems.
- A system should have a **purpose**, and the components work together to achieve it.
- A system exists within an **environment** – everything outside the system that influences it.
- A system has **constraints** – limitations on what it can do and achieve.
- The system has a set of **inputs**, and a set of processes and send the resulting **output** to the environment.

A system architecture decomposes complex designs into simpler subproblems. Decomposing a system, and its subsystems, results in smaller and less complex pieces that are easier to understand. By experimenting with different components and their capabilities, it is possible to explore different design options and produce an overall system behaviour. It is important to specify the system requirement, constraints and specifications on each one of these components, as they are important for the understanding of the System Development Lifecycle (SDLC). The system development lifecycle (SDLC) is a set of activities describing the process for understanding, designing, testing and implementing an information system. The most popular SDLC methodologies are the Waterfall life cycle (Royce, 1970) and the Structured Systems

Analysis and Design Methods (SSADM) (Ashworth, 1988). The latter being the standard UK government analysis and design methodology, covering data, functional and logical views of a system (Chester and Athwall, 2002) . Although there are many variants of the SDLC, Valacich, George and Hoffer (2013) identified four main stages in a SDLC usually found in the different approaches: planning and selection, analysis, design, and implementation and operation (Figure 4-3).



Adapted from Valacich, George and Hoffer (2013)

Figure 4-3: Systems Design Lifecycle

While it appears to be a sequential set of phases, in any stage of the SDLC, it is possible to return to an earlier stage if necessary. Many authors describe it as an iterative process, where the stages are repeated until an acceptable design is achieved. Some systems methodologies even consider the process to be spiral, such as the prototyping (Lantz, 1986) and spiral (Boehm, 1988) methods. The first stage, planning and selection, involves identifying the purpose and specifications of the system. In this stage, existing systems are analysed, and the scope of the proposed system is

determined. The systems analysis stage is concerned with setting the specifications of the proposed system. From the specifications, the structure of the system is created, along with an alternatives design. The third phase, systems design, would convert the recommended alternative into logical and physical specifications. The different components of the system are described independently, and the different technologies associated with it are described. The final stage is a two-step process: implementation and operation. During the implementation stage, the different components of the system are tested separate with the purpose of identifying and correcting errors. After improving the system design, the operation stage applied the system in a real-world environment.

4.2.3 The study research design

The output of this research is a conceptual framework that integrates different components (sub-systems) that could potentially support Traffic management centres to improve the overall accuracy of traffic prediction and incident detection systems. The contribution to knowledge of the proposed framework falls under the improvement quadrant of the DSR knowledge contribution. As stated by Gregor and Hevner (2013), an improvement in DSR can make contributions in more than one level, as shown in Figure 4-1. In this study, methods will be used to describe processes and algorithms within the different components of the framework. Lastly, instantiations are used to demonstrate the level of improvement, compared to existing artifacts.

The SDLC is spread along phases 2, 3 and 4 of the DSR methodology selected for this research (See section 1.3). During planning and selection stage, the purpose of the framework was derived from the findings from the literature review related to

challenges of existing traffic management systems with regards to high alarm rates, low performance and sensor malfunction. The proposed framework incorporates social media and simulation tools to tackle these issues and improve the overall performance of TMS. To this end, during the second stage, different techniques for exploiting Twitter data and imputing missing traffic parameters are investigated with the purpose of generating different alternative designs. In the design stage, the different components of the framework are decomposed into sub-systems. Flow diagrams are used to describe the flow of data through the different components and to understand the steps to perform a process. The last stage is divided in two tasks: testing and implementation. During testing, the different components proposed in the design stage are tested. In this task, different classification algorithms and natural language processing techniques are implemented to find the most suitable ones for the context of the social component. Moreover, the performance of the proposed optimisation procedure for imputing missing traffic data is also tested during this task. Finally, the proposed procedures are implemented in a real-world environment thus it is possible to measure the overall performance on real conditions.

4.3 Smart traffic management framework

Existing TMS collect traffic related data from different sensors (roadway-based and/or probe-based), and process this data using algorithms to detect traffic events and to forecast traffic conditions. Finally, based on the traffic events identified, actions are taken to mitigate the effect of the event, such as lane closure, accident warnings and signal timings. Figure 4-4 portrays the three main phases of existing TMS, based on findings from the literature review (Djahel *et al.*, 2015; De Souza *et al.*, 2017).

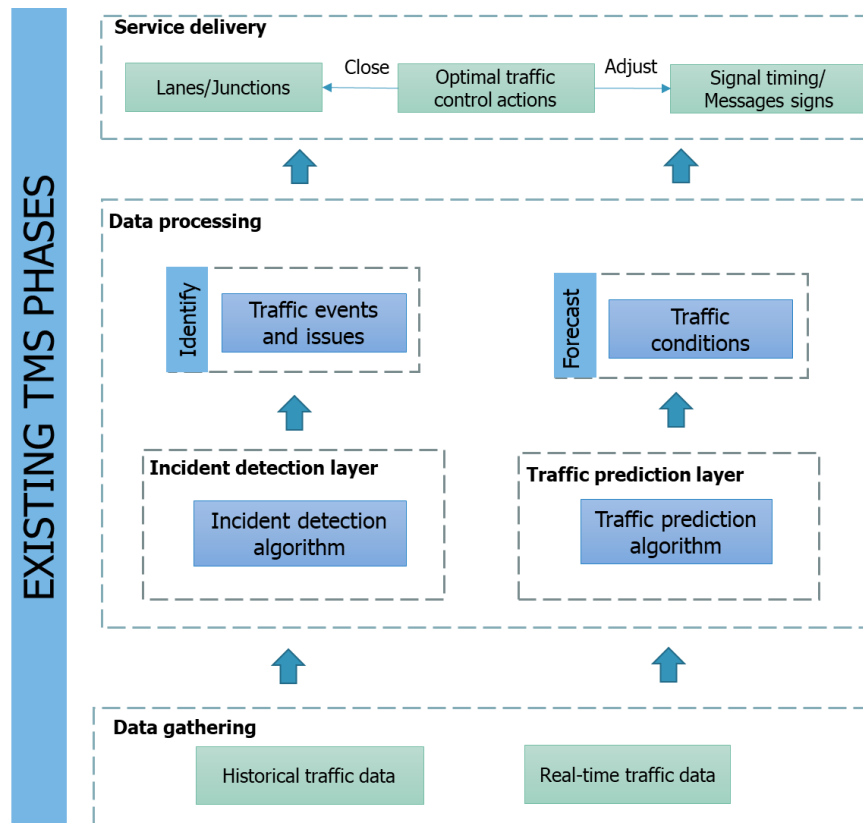


Figure 4-4: Existing TMS phases

This thesis proposes a smart traffic management framework that integrates data from heterogeneous data sources and implements a novelty imputation methodology to find missing traffic data. Figure 4-5 outlines the proposed smart traffic management framework and its different components. The framework is based on findings in the literature suggesting the incorporation of social media and real-time simulations to existing TMS (in section 2.5.3). Similar to the TMS phases outlined in Figure 4-4, the proposed framework is divided in three main stages: data gathering, data processing and service delivery. The framework proposes the integration of social media data during the information gathering stage, and a social media component that uses state-of-the-art NLP and machine learning techniques to process such data. In addition, it introduces a simulator component in the data processing stage, with the purpose of

providing large-scale real-time simulations of the transport network. To tackle the missing traffic data issue, the simulator component introduces an imputation methodology using constrained optimisation. Both proposed components have been highlighted in Figure 4-5, and further detailed on sections 4.3.1 and 4.3.2. The uniqueness of the framework lies in the following areas:

- **Plug-n-play architecture:** Components in the framework are replaceable for any component with a similar function. For instance, on the social media component, it is possible to replace the text classification algorithm with another with similar accuracy.
- **Automation:** The entire process from data acquisition to processing is automated. The proposed framework works with real-time data, and the main components do not depend on one another.
- **Flexible:** It is easy to apply to any existing TMS to improve prediction and traffic incident detection. The different components of the framework can be easily integrated to existing systems used in traffic management centres.

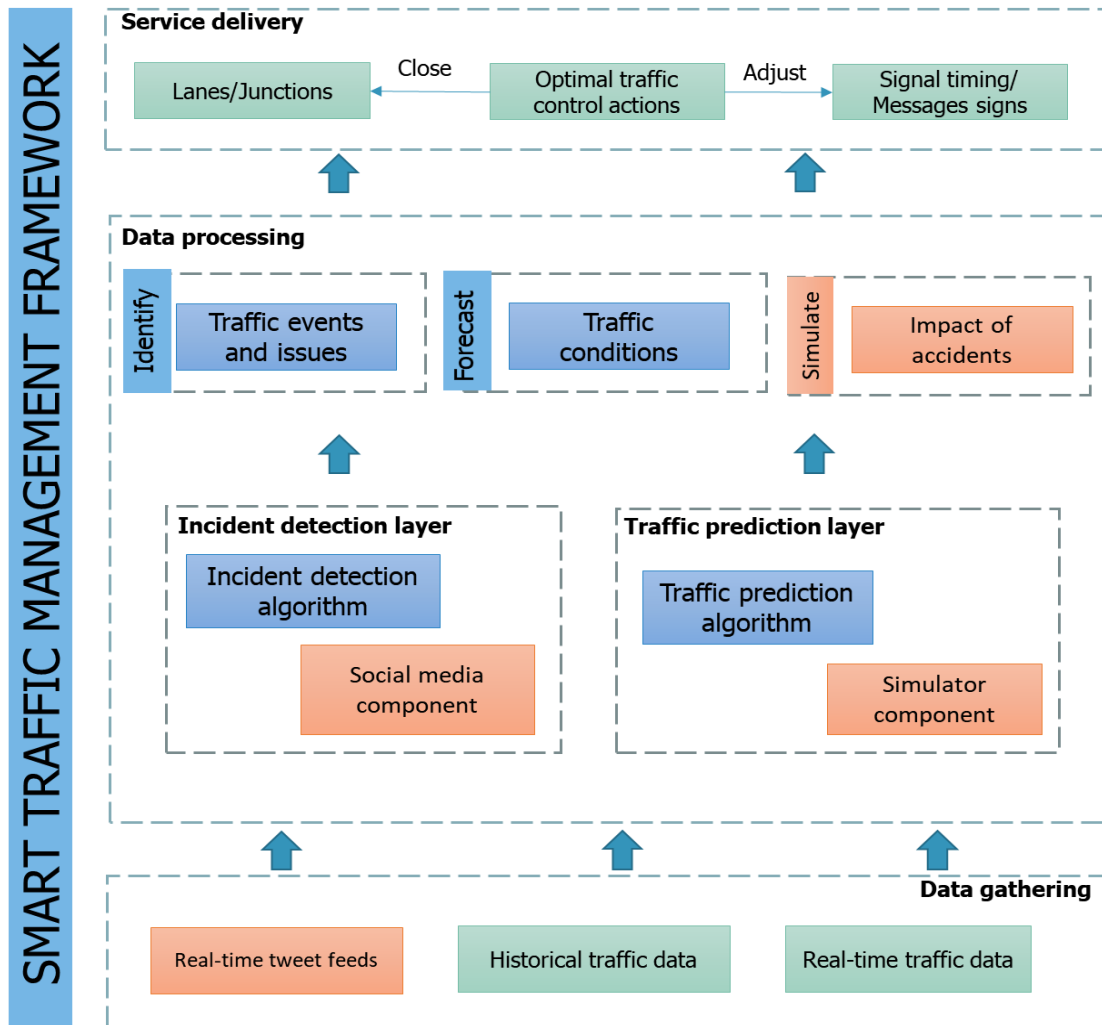


Figure 4-5: Smart traffic management framework

4.3.1 Social media component

The social media component has the potential to identify traffic events and issues on real-time using Twitter data. The proposed component aims to tackle some of the issues faced by researchers when mining social media data (see section 3.4). As discussed in section 3.3, existing methodologies for detecting traffic incidents from Twitter streams fail to integrate more than two of the following features: real-time feed, user-centred approach, incident geolocation and sentiment analysis. The proposed social media component aims to enhance existing methodologies by taking

all these properties into consideration. Figure 4-6 depicts the proposed processing system for the social media component.

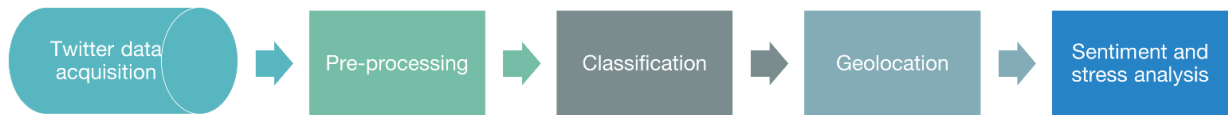


Figure 4-6: Social media data processing system

The different elements of the social media component were derived from findings in the literature as to which were the most effective tools for processing twitter content (as seen in sections 3.2 and 3.3). In terms of data gathering, while studies in the literature have employed the Search API due to the advantages that it offers in filtering by both location and keywords, the Streaming API has been selected as the data gathering tool for this research. This is with the purpose of obtaining a real-time twitter feed of events enclosed within a specific area. Because a tweet can be about anything, the data received will contain a wide amount of noise tweets (e.g. spams, non-traffic related). To this end, additional data processing techniques are implemented for obtaining only tweets mentioning traffic related keywords (see section 5.2.1).

During the data processing stage, NLP techniques discussed in section 3.2.3.1 are implemented to pre-process the tweets before they are fed into the classifier. This step includes the following tasks: tokenisation, stop-words and special characters removal. Tokenisation is the task of transforming a character sequence into pieces, called tokens, and at the same time removing certain characters. During this step, the tokeniser removes mentions, hashtags, URLs, punctuation and emoticons, and splits each tweet into a set of words ('tokens'). Stop words are those common words that have little value in helping characterise a text, such as articles, conjunctions, and prepositions.

As stop words are not very meaningful when deciding if a tweet is traffic related or not, they are removed in this stage. A detailed example of a tweet going through the pre-processing stage can be found in section 5.2.2.

After obtaining a sample of pre-processed tweets, the next step is to classify the information into traffic related or not. There are a broad range of text classification algorithms, which given a training data set can classify into the correct category. Authors in the literature have successfully implemented techniques such as Naïve Bayes , SVM, and kNN (Gu, Qian and Chen, 2016; D'Andrea *et al.*, 2015; Schulz, Ristoski and Paulheim, 2013). In order to select the most accurate algorithm for the proposed methodology, four text classification algorithms were evaluated in section 5.4.2.

The next stage in the pipeline encompasses extracting the location of the already classified tweet sample. In order to identify location mentions in tweets, Named Entity Recognition (NER) is used to label words in a text into entities (organization, person, and location). However, due to the short and informal characteristic of twitter messages, it is necessary to train the NER with similar data thus it can identify specific location mentions. After identifying a location entity, entity disambiguation is used to link an entity to a unique location through a knowledge base. In order to find the most accurate combination, a custom trained NER and different knowledge base (e.g. Bing, Google) were tested in section 5.4.3. Section 5.2.4 provide a more detailed description, as well as some examples of the operation of the geolocation techniques mentioned above.

The last step consists of analysing the emotions within the tweet. To this end, sentiment analysis and stress and relaxation analysis are performed. Sentiment analysis predicts the positive or negative sentiment within a text. By assigning a

polarity, it is possible to know if it is a negative occurrence (delays, roadworks, accident) or a positive one (lanes re-opened, accident cleared). Then, the level of stress and relaxation within the traffic tweet will be detected. This will help in perceiving the users' perception over the state of the network. Section 0 provides some examples on sentiment and stress analysis extraction from traffic-related tweets.

4.3.2 Simulator component

The main purpose of the simulator component is to aid existing TMS in the traffic prediction and incident detection tasks. This component has two different tasks: imputation and real-time simulations of the network. The first stage is an imputation methodology for determining missing traffic data. One of the main issues faced by TMS is the unavailability of traffic data due to malfunction or unavailability of sensors technologies in the road network. Moreover, existing imputation methodologies have been designed for straight sections of highways, and their implementation on more complex networks have not been validated. The simulator component aims to improve the accuracy of the traffic prediction task by integrating an optimisation-based imputation methodology to existing TMS. The proposed imputation procedure is divided in two main tasks: optimisation and validation (presented in section 7.4). The optimisation task consists of a constrained optimisation routine, where the values that best satisfy the objective function are searched for. This function is optimised taking into consideration some constraints on the missing variables. Since there are a wide range of possible solutions that can satisfy the conditions set in the optimisation routine, it is necessary to find the values that will generate low discrepancy between simulated and real data. During the validation task, the simulated values are compared with the

real dataset. This is an iterative process, where if the error does not meet the minimum requirement, the optimisation routine will generate a new set of values. The imputation methodology was outlined and tested in section 7.4.

The purpose of the simulator component in the incident layer is to simulate the impact of accidents in the transport network. After an accident, it is difficult to select what would be the ideal action to mitigate its impact as quickly as possible. This task is concerned with exploring different scenarios to achieve an optimal solution. It provides a real-time road traffic simulation tool, where it is possible to simulate the impact of accidents and to explore the different alternatives to alleviate the congestion. After finding the missing sensor values during the optimisation stage, accidents that have been identified either by the social media component or AID algorithms, can be simulated on this step. Section 8.3 explores the potential of the simulator component for modelling the impact of accidents detected by the incident detection layer.

4.3.2.1 Selection of traffic simulator tool

Three main simulation packages were identified from the literature: AIMSUN, VISSIM and PARAMICS. As discussed in section 2.3.3, several studies have evaluated the strengths and weaknesses of these simulation tools, as well as their ability to simulate real-world traffic conditions (Boxill and Yu, 2000; Gettman and Head, 2003; Young *et al.*, 2014; Mahmud *et al.*, 2018). Some of these studies have concentrated in the ability, features and performance of these simulation packages when modelling traffic incidents. However, it has been concluded that none of the existing packages is superior to any other, but that its selection depends on the scope of the project. The

strengths and weaknesses of these three simulation packages are summarised in Table 4-2.

Table 4-2: Strengths and weaknesses of simulation packages

Simulation tool	Strengths	Weaknesses
AIMSUN	<ul style="list-style-type: none"> - Full trip distribution capabilities - It can simulate a wide variety of traffic incidents over a certain period of time - Suitable for large-scale networks - Flexible facilities to output simulation results - External programming interface 	<ul style="list-style-type: none"> - Much time required for coding a network - Traffic signal methodology more confusing - Total link delays not explicitly calculated - Less user friendly
PARAMICS	<ul style="list-style-type: none"> - Enhanced vehicle actuated control - Suitable for large networks - Good car-following model - Comprehensive visualisation system - External programming interface 	<ul style="list-style-type: none"> - Rely on O/D matrix for estimating traffic flow - Limited options in modelling accidents - Limited options in modelling traveler information/guidance
VISSIM	<ul style="list-style-type: none"> - More detailed network and traffic models - More realistic models for pedestrians and bicycles - It can model the effect of temporary lane blockage - External programming interface 	<ul style="list-style-type: none"> - Sophisticated input and output capabilities - Data coding and input process is challenging and time consuming - Does not model vehicle trajectories in a realistic way

The selection of the tool for the simulator component of this research lies on the ability to implement user-defined algorithms through a programming interface, the convenience of storing output files, flexibility for inputting traffic data, and the capability to simulate a wide variety of traffic events. As seen in the table above, all three simulation packages allow external codes to modify parameters and to implement user-defined applications. However, Guo (2013) argue that while PARAMICS and VISSIM use the graphical interface to record outputs of the simulations,

AIMSUN has more flexible facilities for storing the outputs by generating ASCII files. A significant disadvantage of PARAMICS, compared with AIMSUN and VISSIM, is that it relies on origin-destination matrices to derive traffic volumes, and it provides limited options in modelling incidents and road-works. While AIMSUN has a less user-friendly interface and requires more time to code a network, VISSIM has a more time-consuming data input process which makes VISSIM more suitable for smaller networks. In the assessment of VISSIM and AIMSUN developed by Xiao *et al.* (2005), both simulators obtained a similar score, but AIMSUN overtook VISSIM on incident management capabilities. Based on the above criteria, it was concluded that AIMSUN met the specific requirements of this research.

4.4 Testing and implementing the components

The two different components were tested first (Chapters 5 and 7), and then implemented on real-time (Chapters 6 and 8). In the case of the social media component, four different machine learning algorithms were compared, and the different tools selected to perform NLP techniques, location extraction and sentiment and stress analysis were evaluated during the testing stage of the methodology (Chapter 5). The most suitable techniques from the testing stage were then implemented on real-time in Chapter 6. In the case of the simulator component, Chapter 7 presents the modelled network and proposed imputation methodology and Chapter 8 perform the real-time implementation of the component.

The study area selected for the implementation of the proposed components is the West Midlands county in the United Kingdom. The West Midlands is the second most populated county in England after greater London. It consists of three cities:

Birmingham (largest city), Wolverhampton and Coventry, and the boroughs of Sandwell, Walsall, Dudley and Solihull. Due to its central location, the West Midlands is one of the most accessible counties. Therefore, it is well connected to the other regions of England both by road and rail networks. Many motorways serve the area, such as the M5, M6, M40, M42 and M54, as well as some arterial roads such as the A454, A38, A456 and A491. It is also home of the M6 toll, also called the Birmingham North Relief Road, which is the UK's first toll. A survey by the Office for National Statistics determined that the West Midlands has the highest proportions of people travelling to work by car, when compared to other regions. It was estimated that around 79% of journeys to work were made by car, and only 9% were made on public transport. As a result, it does not come as a surprise that motorways in the West Midlands area suffer from high congestion, especially on peak time of the days. In fact, the M6 is considered one of the busiest motorways and the second least favourite in the UK (Aulakh, 2018).

Twitter statistics in the country of implementation is an important aspect to take into consideration when trying to exploit social media data. For this reason, existing research for mining user generated tweets for traffic incident detection has been applied in countries with high levels of Twitter usage, such as the United States, Italy, and Germany. In the UK, Twitter has over 15 million active users and more than 80% of them access Twitter from their mobile devices (Knight, 2016). In this study, Tweets were collected using an uninterrupted connection with the Streaming API with a geolocation filter using the West Midlands coordinates.

Highways England provide live traffic information from all the motorways in the UK. It provides functions to query real-time data from the WebTris⁹ traffic API. For the implementation of the simulator component, traffic data from the major motorways surrounding the city of Birmingham was extracted using the WebTris traffic API. The motorways included were the M6, M6 toll, M42, and the M5.

4.5 Summary

This chapter presented the Smart Traffic Management Framework proposed in this thesis. The framework proposed the incorporation of two components into existing TMS: social media and simulator. The main purpose of both components is to support the AID and traffic prediction tasks of existing TMS. The social media component aims to identify traffic events and issues by using real-time twitter data. It is composed of a processing pipeline that uses NLP techniques, location extraction and sentiment and stress analysis. On the other hand, the simulator component has two main purposes: imputation of missing flow data and simulation of accidents. Findings from the literature review in Chapter 2, identified missing traffic data as one of the main issues faced by traffic management centres. The simulator component proposes an optimisation-based imputation methodology to solve the missing traffic data problem. It also supports real-time simulations of the network for identifying optimal solutions to existing issues in the transport network. The study area selected for the implementation of the both components was the West Midlands in the UK. Moreover, AIMSUN was selected as the simulation tool for this component, due to its flexible

⁹ <http://webtris.highwaysengland.co.uk/>

facilities to output simulation results and incident modelling capabilities. This chapter has identified DSR as the methodology followed by this research. The contribution to knowledge of this research falls under the improvement quadrant of the DRS methodology, which constitutes a better solution for a known problem. The next chapter will present the social media component and the selected tools for the different stages of the processing pipeline.

Chapter 5: Social media processing system

5.1 Introduction

Chapter 4 presented the smart traffic management framework proposed in this research. This framework aims to incorporate data from Twitter and real-time simulations of the transportation network. To that extent, it proposed the integration of a social media component and a simulator component into existing TMS. The social media component is composed of a data processing system that implements a series of stages for processing and detecting traffic information from tweets. This chapter presents the evaluation of the different techniques selected for the implementation of the social media processing system proposed in section 4.3.1. With the purpose of selecting the most accurate algorithm from the literature, the performance of five text classification algorithms have been compared on a binary classification task. A custom trained NER was also tested, along with different geolocation techniques. Lastly, sentiment and stress analysis were performed on the dataset. The chapter starts by outlining the overall methodology along with the different tools used on each stage of the data processing system (section 5.2). It also includes a brief description of the different text classification algorithms evaluated, as well as the different parameters employed. Section 5.3 describes the datasets and the different evaluation metrics used to measure the performance of the classification algorithms. Sections 5.4 and 5.5 then present the results and a discussion on which are the more appropriate techniques for the implementation of the framework on a real-time basis.

5.2 System architecture

The social media component aims to support existing AID algorithms in the detection of incidents in the transportation network. Section 4.3.1 outlined the proposed system for identifying traffic events from Twitter data within the social media component, as shown in Figure 4-6. Figure 5-1 shows the detailed architecture and the different tools proposed for each stage of the data processing system. The first step, acquiring and filtering, entails obtaining the tweets from the Twitter API, and filtering them using the coordinates of the West Midlands area in the UK. In addition to this, a traffic keyword dictionary is employed for filtering the tweets to obtain only those mentioning traffic related words (e.g. M6, accident, traffic). The next stage, tweet pre-processing, is achieved by using different NLP techniques to pre-process the tweet and prepare the dataset for the classification task. In the classification task, a text classification algorithm is trained to identify whether a tweet is traffic related or not. When a traffic tweet has been identified, the process geolocation and sentiment analysis are triggered. In the following sub-sections, the tasks and the proposed tools are described more in depth.

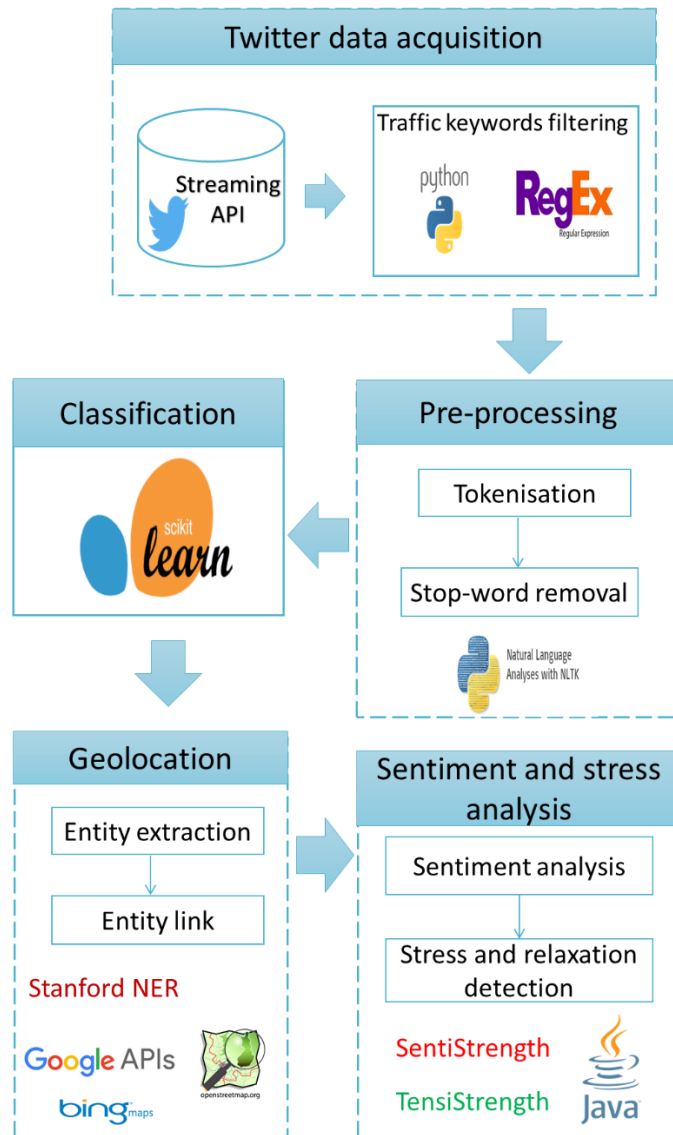


Figure 5-1: System architecture of the social media component

5.2.1 Twitter data acquisition

The aim of the social media component is to process a real-time stream of tweets and detect traffic incidents from them. As discussed in previous sections (sections 3.3 and 4.3.1), the majority of authors in the literature employed the Search API in their studies as it allows to filter by both location and keyword. However, this tool does not provide a real-time feed of tweets. For the purpose of a real-time implementation, the Streaming API was selected for the data acquisition stage of this research. As

presented in section 3.2.2 of the literature review, the Streaming API offers two options for streaming real-time tweets: statuses/filter and PowerTrack. Each option provides different filtering capabilities, rate-limits and rules (see Table 3-1). The free version of the Streaming API (statuses/filter) provides access to about 1% of the firehose in real-time. This means that the total percentage of tweets received on real-time depends heavily on the current tweet traffic. The enterprise edition (PowerTrack) allows to filter with more than one rule and provide access to the full firehose. Entry level prices include up to one million Tweets over a 40-day period and starts at \$1250. The enterprise edition is aimed to track an event/topic/brand over a specific period of time, thus using it for a real-time implementation would not be cost efficient. For this reason, the Statuses/Filter edition was the version employed for this research.

Ideally, in order to get tweets only mentioning traffic words inside a specific area, tweets should be filtered by both location and keyword. However, due to the limits imposed by the Streaming API, it is not possible to perform such filtering. Filtering only by keywords can return tweets from anywhere around the world, which can result in reaching the maximum quota with a small sample of relevant tweets. Therefore, the parameter selected for filtering the incoming real-time tweets was location. Figure 5-2 specifies the bounding box selected to track tweets from the West Midlands area.

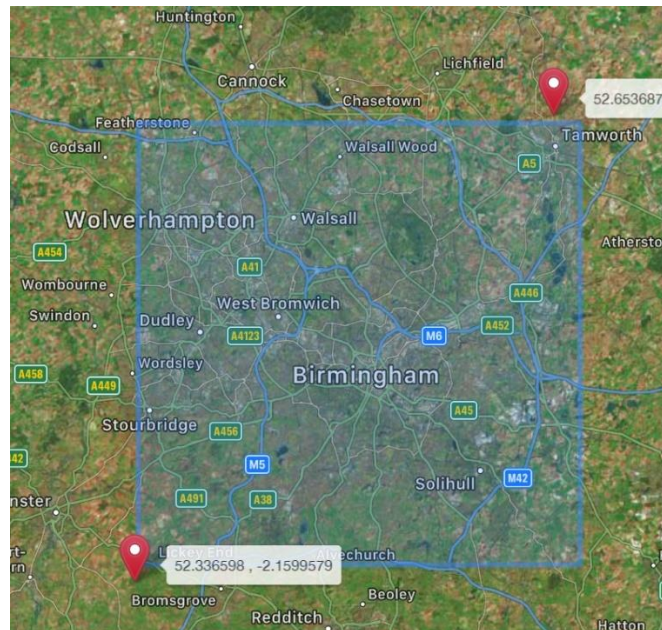


Figure 5-2: Map view of the location bounding box

The Streaming API data is delivered in JSON format. As discussed in section 3.2.2, a tweet JSON format contains different attributes of the tweet such as author, text, coordinates, timestamp, retweet status, a unique ID and, sometimes, even geo metadata shared by the user. Table 5-1 presents the most relevant attributes extracted from a tweet in JSON format for the social media component. Besides the text and the coordinates of the user, other features such as the time it was created, retweet status and language are important attributes to take into consideration. The time it was created is important for identifying at what time the traffic incident is taking place. The retweet status helps to eliminate those tweets that are only a retweet, as they contain repeated information. Lastly, the language attribute facilitates the elimination of non-English tweets.

Table 5-1: Main attributes of a tweet used for the social media component

Attribute	Description
Text	The actual tweet message
Coordinates	Represents the location of the tweet as reported by the user reports it. A set of latitude and longitude coordinates or a bounding box
Retweeted status	It helps identifying if a tweet has been a retweet or not
Language	Filters by an specific language
ID	A unique user ID, represented by a long number
Created at	The time and date the tweet was posted

Since tweets were only filtered by location, additional data processing techniques were implemented for obtaining only those mentioning traffic related words. To this end, a dictionary of the main highways and arterials roads within the West Midlands (e.g. M6, A449, M5) was created. The dictionary also includes traffic incident related keywords such as "congestion", "accident", "roadworks" and "delays". This dictionary was used as a filtration criteria to obtain a second dataset of tweets mentioning at least one of these traffic-related keywords. For this step, a regular expression filter was employed to filter the text attribute within the acquired tweets using the roads and traffic words in the dictionary. In addition, further filtering at this data processing step removed all those tweets containing a retweeted status attribute, as they only consist of repeated information. Appendix 5-A comprises the list of traffic related keywords employed during this step.

5.2.2 Pre-processing

Due to their informal nature, tweets usually contain mentions, hashtags, links, special characters and emoticons. This information needs to be removed before tweets are further processed by the classification stage of the architecture. Figure 5-3 depicts a tweet going through the different NLP techniques employed in the proposed pipeline. For a definition of the NLP techniques presented in this section, see section 3.2.3.1.

The first step is tokenisation, and it consists in transforming the tweets into a set of tokens (words) and removing non-alphanumeric characters. While there are a wide range of tokenisation tools, they fail to recognise special tweet features such as mentions, emoticons, URLs and hashtags as individual tokens. For this reason, a pre-processing chain based on regular expressions that considers all these aspects was implemented (See Appendix 5-B). During this step, the tokeniser removes mentions, hashtags, URLs, punctuation and emoticons, and splitting each tweet into a set of words ('tokens'). The set of tokens then goes through the last step of the pre-processing stage, stop word removal. This final step consists in eliminating those words that do not help to characterise a text (e.g. articles, prepositions). The full list of English stop words from the Natural Language Toolkit¹⁰ (NLTK) library was used to remove stop words from the set of tokens.

¹⁰ <https://www.nltk.org/>

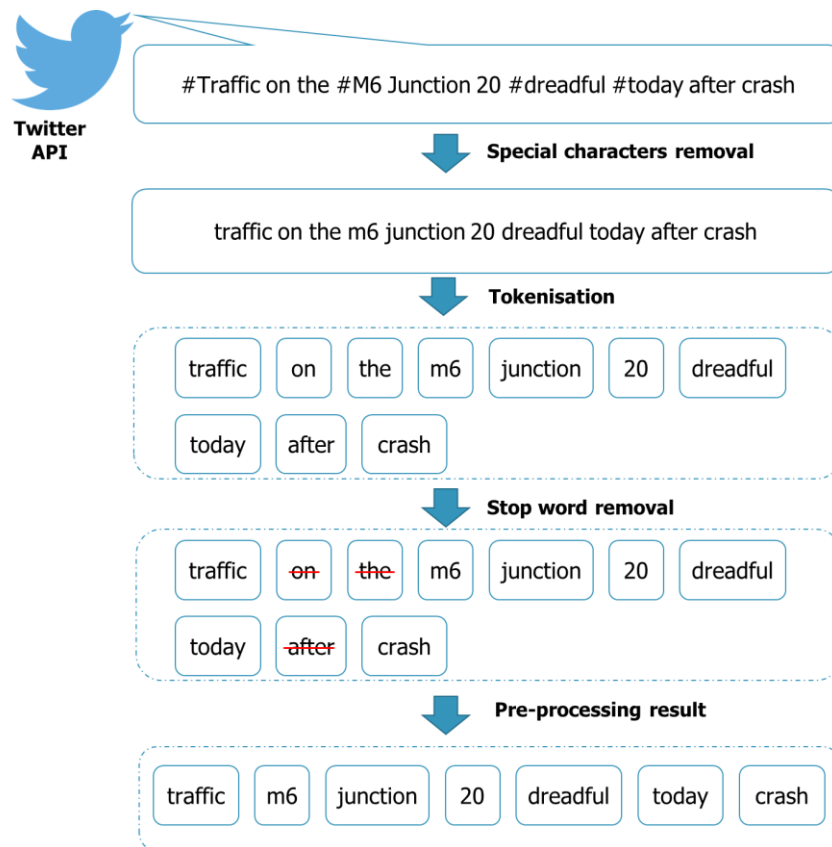


Figure 5-3: Tweet pre-processing example

5.2.3 Classification

Once tweets have been pre-processed, they were classified into traffic related or not. Studies in the literature have employed and compared a wide range of text classification algorithms for incident detection using Twitter data (Wanichayapong *et al.*, 2011; Schulz, Ristoski and Paulheim, 2013; E. D'Andrea *et al.*, 2015; Gu, Qian and Chen, 2016). Text classification consists in assigning a set of predetermined categories to a text. This is achieved through machine learning and NLP techniques to automatically classify the text into specific categories. There are a broad range of text classification algorithms, which given a training data set are capable of successfully classifying into the correct category. These algorithms learn from the different features in the training data and associate these features with a particular output (or category).

The first stage of training a text classification algorithm is feature extraction. During this step, the text is transformed into a numerical representation in the form of a vector. Then, the training data is fed to the classifier in the form of pairs of feature sets (vectors) and associated tags (e.g. traffic, non-traffic). Once the machine learning algorithm is trained, the same feature extraction is implemented to transform unseen data into feature sets, thus the classifier could label them (e.g. traffic and non-traffic). Similar to humans, machine learning algorithms learn from past examples. Therefore, text classification algorithms depend entirely on the training data and the feature extraction fed to them. Figure 5-4 depicts the process for the creation of the text classification model.

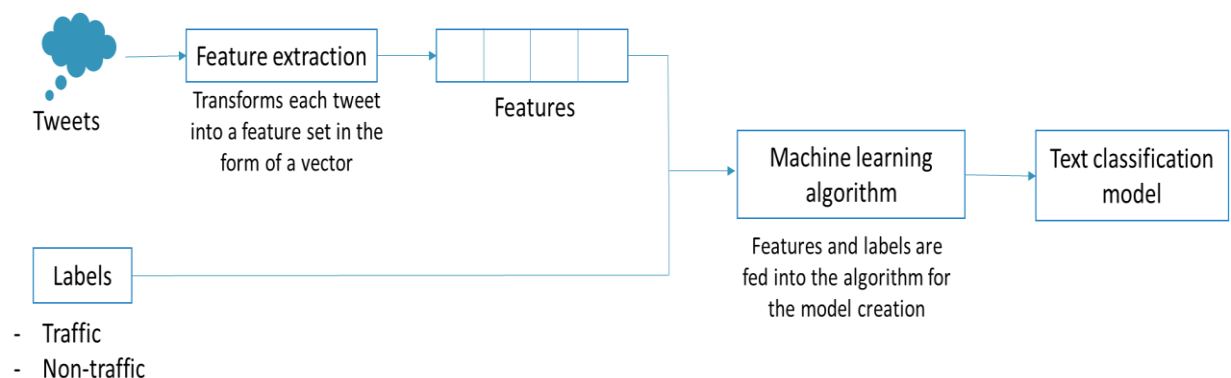


Figure 5-4: Creating a text classification model

The aim of a good text classification algorithm is to make accurate predictions on data that the model has never seen. However, one of the biggest causes of poor performance is data overfitting. Overfitting occurs when instead of learning the general patterns that can be found in the data, the classifier learns the details and noise of the training data in a too literal manner. This prevents the model from correctly classifying unseen data, as the concepts learned by the model do not apply to this data. While

there is no general solution for overfitting, Brownlee (2016) recommends two techniques to limit overfitting when evaluating machine learning algorithms:

- Implement a resampling technique
- Use a subset of the data as a validation dataset

The most popular resampling technique is k-fold cross validation. When using this method, the model is trained and tested k-times on different subsets. This allows to have an estimate of how well the classifier is performing. After the model has been trained, the accuracy of the classifier can be evaluated using a validation dataset. A validation dataset consists on keeping a subset of the training data, with the purpose of having a final idea of how the model will perform on unseen data.

For this study, machine learning algorithms supported by Ridge Regression Classification (RRC), Naïve Bayes (NB), k-Nearest Neighbour (kNN), Multilayer Perceptron (MLP) and Support Vector Machine (SVM) were developed. With the purpose of finding the most appropriate feature set for each classifier, these algorithms were combined and evaluated on different features on the training dataset using a k-fold cross validation. Before applying the overfitting techniques, the tweets extracted from the Streaming API were divided in training and testing dataset. For these tasks, the machine learning library ScikitLearn¹¹ was employed. In the following subsections, the text classification algorithms, as well as the different parameters used are described.

¹¹ <https://scikit-learn.org/stable/>

5.2.3.1 Employed classification algorithms

kNN (Cover and Hart, 1967) is one of the simplest text classification algorithms. It assigns to a test sample the most common class of its k nearest training samples (Peterson, 2009). The parameter k is used to specify the number of neighbours.

RRC (Hoerl and Kennard, 1970) is a data modeling technique for solving the multicollinearity problem in multiple regression. Multicollinearity occurs when there is a strong linear relationship between two or more predictor variables in a multiple regression model. RRC is a regularised least square method that deals with the multicollinearity problem by modelling the linear dependency between class specific samples and a new test sample (He *et al.*, 2014).

NB (Friedman, Geiger and Goldszmidt, 1997) is a supervised learning approach based on the Bayes theorem with naïve independence assumptions between features. It assumes that the presence of a particular feature is unrelated to any other feature. Although this assumption is false in most real-world problems, NB frequently performs classification very well (McCallum and Nigam, 1998).

MLP (Rosenblatt, 1961) is a type of artificial neural network consisting of a system of interconnected neurons, representing a nonlinear mapping between an input and an output vector (Gardner and Dorling, 1998). MLP networks generally involve several layers of neurons, including one or more hidden layers and an output layer.

SVM (Cortes and Vapnik, 1995) is a supervised learning approach that constructs a hyperplane in a high-dimensional space. In a SVM, each data item is plotted as a point in n-dimensional space, where the value of each feature is the value of a particular

coordinate. It then performs a classification by finding the hyper-plane that differentiates the two classes (Srivastava, 2014).

5.2.3.2 Features

For training the classifiers, two features were taken into consideration: Word n-grams and a Term-frequency times inverse document-frequency (Tfidf) representation. N-grams are a continuous sequence of n items used to characterise texts. A count vectoriser was used to convert the tweets into token counts based on different combinations of n-grams. The count vectoriser keeps track of how many times these n-grams occur in a text. The n-grams used were:

- Unigrams, or words (n-gram size = 1)
- Bigrams, or terms compounded by two words (n-gram size = 2)
- Trigrams, or terms compounded by up to three words (n-gram size = 3)

The count matrix (vectoriser) is transformed to a Tfidf representation, which is a weighting factor used to reflect how relevant a word is in a document. The purpose of Tfidf is to determine the importance of those words appearing less often in the tweet sample. It also aims to reduce the impact of those words that appear very frequently but are less relevant.

5.2.4 Geolocation

One of the biggest challenges faced by authors in the literature is assigning a location to the tweet (see section 3.4). Due to privacy issues, informal language or just omitting the location of the incident, it is very difficult to geolocate a traffic related tweet. Even when the location is specified in the tweet, it can pose a challenge to link it to a unique

place, as one concept could refer to more than one location. In fact, most authors in the literature have mainly focused in the classification task, while few of them have actually employed techniques for geolocating the incident (as shown in Table 3-5). The proposed pipeline uses a NER to identify the location entity from the traffic tweet, and entity disambiguation to link the concept to a unique location.

A NER labels words in a text into specific categories, such as person, company or location. The Stanford NER¹² is one the most popular models used for entity extraction and therefore, the tool selected for this research. The Stanford NER is trained to label text into three classes: Person, Organisation and Location. It implements a Conditional Random Field (CRF) algorithm, trained on an already tagged dataset. However, this model presents very low accuracy when identifying location mentions on tweets due to their informal nature. To tackle this, the Stanford NER model was retrained using a sample of the training data. For this process, traffic related tweets from the training data were manually labelled into two categories: Location and Other. First, every tweet was converted into a set of tokens. Then, each token that represented the name of a motorway, arterial road, roundabout, junction, or city, was manually labelled as a location ('LOC'). This data was then used to build the custom CRF classifier designed to extract the geolocation from tweets. A sample of the manually labelled entities used to train the model can be found in Appendix 5-C.

After identifying these location mentions, a knowledge base is used to map the entities to geographical coordinates. Many online maps providers offer API services where is

¹² <https://nlp.stanford.edu/ner/>

possible to use their information to locate specific places. Geopy¹³ is a python library that provides several popular geocoding services. They use third-party geocoders and other data sources to locate the coordinates of addresses, cities, countries and landmarks around the world. With the purpose of finding the most accurate knowledge base, the performance of geocoders from Google, OpenStreetmap (Nominatim) and Bing were evaluated during this stage. Table 5-2 has an example of the process for extracting the location in a set of traffic tweets.

Table 5-2: Example of Geolocation stage

Entity extraction	Entity Link
Highlight of the day, Catthorpe Interchange. Really good now, roadworks finished, 10/10 would use again.	Catthorpe_Interchange
#traffic on the #M6 junction 20 dreadful #today after crash	M6_J20
The M6 northbound before and after Junction 18 in the roadworks is awful - lanes 1 and 2 are full of massive pot holes!	M6_J18

5.2.5 Sentiment and stress analysis

5.2.5.1 Sentiment analysis

The next steps in the system architecture consists in assigning a polarity to the tweets. To achieve this, sentiment strength detection has been performed using SentiStrength. Sentiment strength detection predicts the strength of positive or negative sentiment within a text (Thelwall, Buckley and Paltoglou, 2012). Thelwall *et al.* (2010) presented SentiStrength, a classifier that uses additional linguistic information to detect sentiment strength in short informal text. For each text, the SentiStrength will output two integers: one for positive sentiment strength and another for negative sentiment strength. The scores range from 1 to 5 for positive sentiment and -1 to -5 for the

¹³ <https://geopy.readthedocs.io/en/stable/#>

negative one. As an example, Table 5-3 depicts some traffic related, and their sentiment strength analysis performed with the online version of SentiStrength 2. While the first tweet is a compliment thus it has a positive polarity, the other two have a negative one due to complaints about traffic and roadworks.

5.2.5.2 Stress and relaxation analysis

For the last step of the system, TensiStrength have been used for analysing the level of stress or relaxation within the tweet. TensiStrength uses a lexical approach to detect indicators of stress and relaxation expressed in short text messages (Thelwall, 2017). Similarly to SentiStrength, TensiStrength will output two values on a scale from 1 to 5 for relaxation and -1 to -5 for stress. Table 5-3 also shows the results of the stress and relaxation analysis made with the online version of TensiStrength. It can be perceived that in the first tweet, the user is obviously happy and relaxed over the state of the network, thus the relaxation score is high (3) and shows not to be stressed (-1). As for the other tweets, they are obviously complaining, which makes their stress level higher.

Table 5-3: Sentiment and stress analysis using SentiStrength and TensiStrength

Tweet	SentiStrength		TensiStrength	
	+	-	+	-
Highlight of the day, Cattrhorpe Interchange. Really good now, roadworks finished, 10/10 would use again.	4	-1	4	-1
#traffic on the #M6 junction 20 dreadful #today after crash	1	-4	1	-4
The M6 northbound before and after Junction 18 in the roadworks is awful - lanes 1 and 2 are full of massive pot holes!	1	-4	1	-4

5.3 Description of the datasets

Using an uninterrupted connection with the Twitter Streaming API, 4 million tweets were collected from March 1st, 2017 to May 31st, 2017. From this data, the regular expressions filter extracted 13,410 tweets, using a dictionary of 265 road names and traffic related keywords. From the three-month period, May obtained the highest amount of traffic related Tweets. This was influenced by a high volume of tweets on the 15th and 16th of that month, due to the identification of an undetonated WWII bomb in the city centre of Birmingham. Further confirming the applicability of Twitter for detecting real-world events. Table 5-4 has some examples of traffic and non-traffic related tweets on the dataset.

Table 5-4: Example of labelled tweets

Tweet	Label
Brum traffic chaos all entry and exit slip roads to m6 at spaghetti junction and the whole a38m are closed due to a bomb being found #ww2	Traffic
Massive car crash on pedmore road by merry hill going towards halesowen road all shut off so avoid it	Traffic
Traffic chaos bingo big delays in #birmingham #ww2bomb #aston	Traffic
Just heard...interview car crash is an understatement	Non-traffic
After a few rough days following my crash im working hard staying positive to get fixed for	Non-traffic

Tweets were then manually labelled into traffic and non-traffic related, with a total of 1161 traffic related tweets. It is important to mention that only tweets that specified the place where the incident is taking place were considered to be traffic related. Because the portion of tweets from the non-traffic related category was much larger than the traffic related one, the classifier could develop a bias towards the former. To avoid this, under-sampling was employed. This was done by selecting a random portion of the non-traffic related class, until a balanced dataset is reached. That is,

when both classes represent 50 percent of the training dataset. As a result, 1161 tweets were randomly selected from the non-traffic related sample. These traffic tweets were further divided into two datasets with a 75/25 split ratio between the following datasets:

- Training: This is the portion of tweets used to train and validate the text classification algorithms. It consisted of 871 traffic related tweets and 871 non-traffic related.
- Testing: To show the effectiveness of the classifiers on a different dataset, a testing dataset of 348 traffic tweets and 348 non-traffic related tweets was built.

5.4 Results

5.4.1 Evaluation metrics

With the purpose of evaluating the performance of the classifiers and the custom NER, standard classification metrics were employed. These are: accuracy, precision, recall, and F-measure. Generally, these evaluation metrics are derived from a matrix, called 'confusion matrix', which portrays the number of examples correctly and incorrectly classified for each class. Table 5-5 presents the general outline of a confusion matrix for a binary classification problem (e.g. traffic and non-traffic related). True Negative (TN) and True Positive (TP) correspond to the tweets that were classified correctly as non-traffic and traffic related, respectively; while False Negative (FN) and False Positive (FP) tweets are those that were misclassified as non-traffic and traffic tweets.

Table 5-5: Confusion matrix of a binary classification problem

	Predicted class	
True class	Traffic	Non-Traffic
Traffic	TP	FN
Non-traffic	FP	TN

The accuracy, precision, recall, and, F-measure are then derived from the values of the confusion matrix, as shown in Table 5-6. Accuracy is the overall efficiency of the classifier and corresponds to the fraction of correctly classified tweets by the total number of tweets. Precision of a class represents the fraction of correctly classified tweets within that class. Recall of a class is the fraction of correctly classified tweets over the total number of tweets that belong to that class. F1-score is the harmonic mean of precision and recall.

Table 5-6: Evaluation metrics

Metric	Formula
Accuracy	$\text{accuracy} = \frac{(TP+TN)}{(TP+FP+TN+FN)}$
Precision	$Prec = \frac{TP}{TP + FP}$
Recall	$Rec = \frac{TP}{TP + FN}$
F1 score	$F1 = \frac{2 \ vP \ vR}{P + R}$

5.4.2 Classification task

Each classifier was tested with different n-gram values on the training dataset. The purpose of this was to identify which feature works best with the different machine learning algorithms. For this step, a k-fold cross validation methodology was employed. K-fold cross validation randomly divides the dataset into k equally sized subsets. One

of these subsets was retained for testing the model, while the remaining k-1 were used as training data. This process is repeated k times, using each of the k subsets exactly once as testing data. K-fold cross validation was implemented on the training dataset with $n = 10$, for each classifier/n-gram variation.

Table 5-7 shows the overall accuracy obtained from the K-fold cross validation of the training data using the different n-gram ranges for each classifier. For each classifier, the 10-fold cross validation using unigrams, bigrams, unigrams and bigrams, and unigrams, bigrams and trigrams was performed. This accuracy was calculated with the formula presented in Table 5-6 using the values of the confusion matrix for each classifier/n-gram variation. The overall accuracy presented in Table 5-7, represents the average of the 10 values of accuracy obtained in the cross validation. It can be said that most of the classifiers had higher performance using unigrams or the combination of the three features, while the worst performance amongst all was observed on the trigrams. This can be related to the short nature of traffic tweets, where unigrams and bigrams contain more factual keywords to describe traffic events in short texts. The values in bold correspond to the n-gram variation with the highest accuracy for each algorithm.

Table 5-7: Classifiers accuracy vs Word n-gram features

Model	Unigrams	Bigrams	Trigrams	Unigrams and Bigrams	Unigrams, Bigrams and Trigrams
RRC	91.20%	84.99%	64.75%	89.79%	88.62%
KNN	87.39%	76.02%	52.52%	88.56%	88.12%
NB	88.01%	78.40%	59.03%	88.75%	88.62%
MLP	88.56%	85.43%	65.25%	89.48%	89.79%
SVM	90.52%	84.93%	65.51%	89.67%	88.38%

The feature with the highest accuracy on each classifier was selected for further evaluation on the test dataset. Table 5-8 depicts the evaluation metrics for each classifier on the testing dataset. The classifier with the highest accuracy was the RRC with 93.11%. MLP and SVM had marginally inferior performance to the Ridge classifier both with 90.05% and 91.58% respectively, while the KNN was the one with the lowest accuracy with 88.01%. These results show that the classifiers were not overfitted for the labelled examples in the training dataset. The classifiers had more precision predicting non-traffic related tweets, but less recall. This shows that while the model identified a higher number of traffic related tweets, it had more precision identifying non-traffic related ones.

Table 5-8: Classifiers evaluation metrics on the test dataset

Model	Traffic			Non-Traffic			Accuracy
	Prec	Rec	F1	Prec	Rec	F1	
RRC	90.43%	96.17%	93.33%	96.17%	89.79%	92.88%	93.11%
KNN	89.84%	85.71%	87.73%	86.34%	90.31%	88.28%	88.01%
NB	86.11%	94.90%	90.29%	94.32%	84.69%	89.25%	89.80%
MLP	85.52%	96.43%	90.64%	95.91%	83.67%	89.37%	90.05%
SVM	87.21%	97.45%	92.05%	97.11%	85.71%	91.06%	91.58%

Results from the testing dataset suggest that a RRC, MLP or a SVM model would obtain high accuracy on classifying tweets into traffic related or not. However, there are other aspects that needed to be taken into consideration, such as the training and prediction time. Table 5-9 contains the training and prediction time (seconds) of each algorithm on the test dataset. RC and SVM are the fastest in both training and prediction both with 0.04s and 0.008s respectively. However, although MLP obtained one of the highest accuracy scores, it needed 43.53s to train. This is more than 1000 times of

what was needed by the RRC and the SVM. Contrary to RRC and SVM, MLP was more accurate when using unigrams and bigrams, instead of only unigrams, which increased the computing time. However, training time does not really affect the performance of a real-time implementation, as the classifier only needs to be trained once. In terms of prediction, all classifiers obtained promising results, with the highest prediction time being only 0.048s (48 milliseconds) by the KNN. These results further demonstrate the suitability of the classifiers for a real-time implementation.

Table 5-9: Training and prediction time (sec)

Algorithm	Training time	Prediction time
RRC	0.044	0.008
KNN	0.149	0.048
NB	0.176	0.039
MLP	43.53	0.02
SVM	0.04	0.008

5.4.3 Geolocation

5.4.3.1 Entity Recognition

The Stanford NER was retrained using the traffic related tweets from the training dataset. As highlighted before, the traffic related tweets from the training dataset were manually labelled into two categories: Location (LOC) and Other (O). Location referred to all location mentions referring to cities, roads, roundabout and junctions. Other referred to anything that was not relevant in the identification of the location of the traffic event. The custom trained NER model was then implemented on the test dataset. Table 5-10 shows the overall accuracy of the trained NER model. The accuracy measures the number of locations identified from the dataset, divided by the total number of tweets with a location attribute. From the 290 tweets on the test data, the

proposed custom NER model was able to successfully extract location entities from 278 traffic tweets. The overall accuracy of the model was 95.86%.

Table 5-10: NER model accuracy

Predicted	290
Actual	278
Accuracy	95.86% (278/290)

5.4.3.2 Entity link

After identifying location mentions in the tweets, the next step was to map these entities to specific locations. Different geolocation engines were compared with the purpose of identifying the one with the highest performance. The entity link component was tested with the 278 tweets which location entity was previously extracted by the NER model in section 5.4.3.1. Table 5-11 shows the performance of the different geocoders, with regards to the evaluation metrics presented in section 5.4.1. Google API had the highest performance with a precision of 98.54% and an accuracy of 97.12%. Nominating and Bing engines resulted with an accuracy of 58.27% and 69.42%, respectively. Unlike Google and Nominatim, Bing does not allow to filter by country. This might be the reason why its precision was the lowest amongst the geocoders, as many locations were outside the UK. Some of the locations identified by the geolocators, while they were correctly classified, were outside of the West Midlands area.

Table 5-11: Performance of Geolocation engines

	Google API	Nominatim	Bing
Predicted	274	168	266
True positive	270	162	193
False Positive	4	6	73
Precision	98.54%	96.42%	72.55%
Accuracy	97.12%	58.27%	69.42%

5.4.4 Sentiment and stress analysis

5.4.4.1 Sentiment analysis

Sentiment analysis was implemented in all the traffic related tweets, both in the training and testing sample. A total of 1161 tweets were analysed calling the SentiStrength java version through a python subprocess. Table 5-12 and Table 5-13 portray the measures of frequency of the sentiment analysis results. Each positive/negative score in these tables comprises the count of the occurrence of these values in the dataset. The percent values represent the percentage of the individual frequencies with respect of the total frequency. It can be perceived that the sample of traffic related tweets did not obtain high values in terms of positive and negative sentiment. The positive sentiment obtained only 3.9%, while the negative sentiment attained 8.9% in scores of more than three. This shows that while there is more negative sentiment within the tweets, it is still a very low value which says that tweets in the sample did not portray strong sentiments. This conclusion can be further confirmed with the measures of central tendency (Table 5-14), where it can be appreciated that the mean of both positive and negative sentiment was lower than 2.

Table 5-12: Frequency of positive sentiment results

Score	Frequency	Percent	Cumulative Percent
1.00	1024	88.2	88.2
2.00	91	7.8	96.0
3.00	42	3.6	99.7
4.00	4	0.3	100.0
Total	1161	100.0	

Table 5-13: Frequency of negative sentiment results

Score	Frequency	Percent	Cumulative Percent
-5.00	6	0.5	0.5
-4.00	29	2.5	3.0
-3.00	68	5.9	8.9
-2.00	614	52.9	61.8
-1.00	444	38.2	100.0
Total	1161	100.0	

Table 5-14: Measures of central tendency for SentiStrength

Measure	Positive	Negative
Mean	1.161	-1.742
Median	1.000	-2.000
Mode	1.000	-2.000

5.4.4.2 Stress and relaxation analysis

Similarly to SentiStrength, the traffic related tweet sample was used to perform stress and relaxation analysis by calling the TensiStrength java version through a python subprocess. Table 5-15 and Table 5-16 present the count of occurrence of relaxation and stress scores (frequency), and their correspondent percentage of the total frequency. From Table 5-15, it can be perceived that the frequency of relaxation leaned towards low values, with 90.5% of the sample having level 1 of relaxation. On

the other hand, Table 5-16 shows that around 67% of the tweets showed a stress strength of more than 3. The mode and the median (Table 5-17), illustrate that indeed relaxation tended to be in a low level (1), while stress inclined to higher values (3).

Table 5-15: Frequency of relaxation results

Score	Frequency	Percent	Cumulative Percent
1.00	1051	90.5	90.5
2.00	87	7.5	98.0
3.00	19	1.6	99.7
4.00	4	0.3	100.0

Table 5-16: Frequency of Stress results

Score	Frequency	Percent	Cumulative Percent
-5.00	4	0.3	0.3
-4.00	140	12.1	12.4
-3.00	630	54.3	66.7
-2.00	76	6.5	73.2
-1.00	311	26.8	100.0

Table 5-17: Measures of central tendency for TensiStrength

Measure	Positive	Negative
Mean	1.118	-2.526
Median	1.000	-3.000
Mode	1.000	-3.000

5.5 Discussion

Findings in the literature, in both event and incident detection, suggested the SVM as the most accurate classification algorithm (Sakaki, Okazaki and Matsuo, 2010; Schulz, Ristoski and Paulheim, 2013; D'Andrea et al., 2015; Gutierrez et al., 2015). However, results from the experimental implementation of the social media component

presented on this chapter suggested the RRC as the most accurate classification algorithm in this dataset, with an overall accuracy of 93.13%. Reflecting upon the initial comparison of the different Twitter based incident detection studies in the literature on section 3.3, Table 5-18 compares the results of the implementation of this test against other authors. Results from the text classification algorithm outperformed the most relevant studies in the literature, those of Schulz *et al.* (2013), Gu, Qian and Chen (2016), and Zhang *et al.* (2018). Indeed, D'Andrea *et al.* (2015) and Gutierrez *et al.* (2015) obtained a higher accuracy in their proposed SVM models, with 95.75% and 95.50% respectively. However, D'Andrea *et al.* (2015) tested their methodology on the training dataset, and Gutierrez *et al.* (2015) only took into consideration tweets from UK agencies with the purpose of notifying drivers about the status of the network. Because tweets from these agencies are more structured and possess more useful information, it would be easier for an algorithm to detect a traffic tweet from an agency than from a human. The scope of this research is on using user-generated content to identify issues on the transport network, as traffic agencies spread information that has already being processed and communicated to the users.

Table 5-18: Comparing results with other studies in the literature

Authors	Classifier accuracy	Entity recognition	Entity link
Schulz, Ristoski and Paulheim (2013)	89%	95.5%	87%
Gutierrez <i>et al.</i> (2015)	95.50%	80.98%	
D'Andrea <i>et al.</i> (2015)	95.75%		
Gu, Qian and Chen (2016)	90.50%	82%	
Zhang <i>et al.</i> (2018)	85%		
This work	93.13%	95.86%	97.12%

The geolocation components showed very promising results in both entity recognition and entity link. The entity recognition was able to extract location entities with an overall accuracy of 95.86%. As illustrated in Table 5-18, results from the geolocation stage outperformed other studies in the literature, such as those in Gu, Qian and Chen (2016) and Gutierrez *et al.* (2015); and obtained similar results as Schulz *et al.* (2013). While it proved to be very accurate, it is necessary to use further NLP techniques to improve the results of the entity extraction. For instance, sometimes tweets tend to have repetition of the location entities with the hashtags, which would end in having repetitive locations on one tweet. An example of this is the following tweet: "M6 shut due to serious accident in junction 14 southbound #M6". The location extraction from this tweet was: M6 junction 14 southbound M6. This is something quite common in the dataset, where the use of hashtags led to repetitions in the entity extraction.

During the entity link stage, three different geolocators were compared: Google API, Nominatim and Bing. Google API outperformed the others with a prediction accuracy of 97.12%, while Nominatim and Bing obtained 58.27% and 69.42%. Schulz, Ristoski and Paulheim (2013) were the only ones to perform entity link to the extracted locations, and results from the Google API in this study outperformed their accuracy. Even with the good performance achieved by the Google API, there are still some challenges associated with its implementation. Sometimes users failed to specify the exact position of the accident. For instance, a tweet might say: 'Standstill traffic at the M6'. The NER component extracted the location as M6, and the Google API found the location to be the "M6 Motorway, United Kingdom". While both components correctly identified the location of the tweet, it is impossible to know in which part of the M6

motorway the incident is taking place. This is the case of many tweets in the dataset, where the output of the three geolocators did not refer to a specific junction due to lack of information. The opposite to this was also observed; while users specified along which exact junction of the M6 the incident was taking place, the geocoder failed to identify the precise location. The output in these cases was "M6, United Kingdom", and a generic set of coordinates for this motorway. Although it is correct, it is not possible to determine the precise location of the incident from this information. The results also showed that some of the tweets were outside of the boundaries set by the geolocation filter. This could be related to the fact that sometimes users set their city in their Twitter profiles, but it does not mean that it corresponds to the location from which they are tweeting from.

Sentiment analysis results suggested that both the positive and negative sentiment were not significant in the dataset. While the main goal of this study was to capture whether the traffic event was a positive or negative occurrence using SentiStrength, the majority of the tweets showed neutral sentiment. This is because tweets tended to have a score of 1 in both positive and negative (Figure 5-5). Opposed to SentiStrength, TensiStrength had a better performance when identifying negative emotions. This was observed when the comparison between the measures of central tendency showed negative (stress) with a mode of 3, and positive (relaxation) with a mode of 1. This can be confirmed in Figure 5-6, where it is clear that stress was predominant on the dataset, when compared with relaxation. These findings are corroborated by Thelwall (2017), where it was concluded that TensiStrength was more accurate in detecting sentiment in short texts. None of the studies in the literature

have included sentiment analysis as part of their Twitter-based traffic incident detection systems.

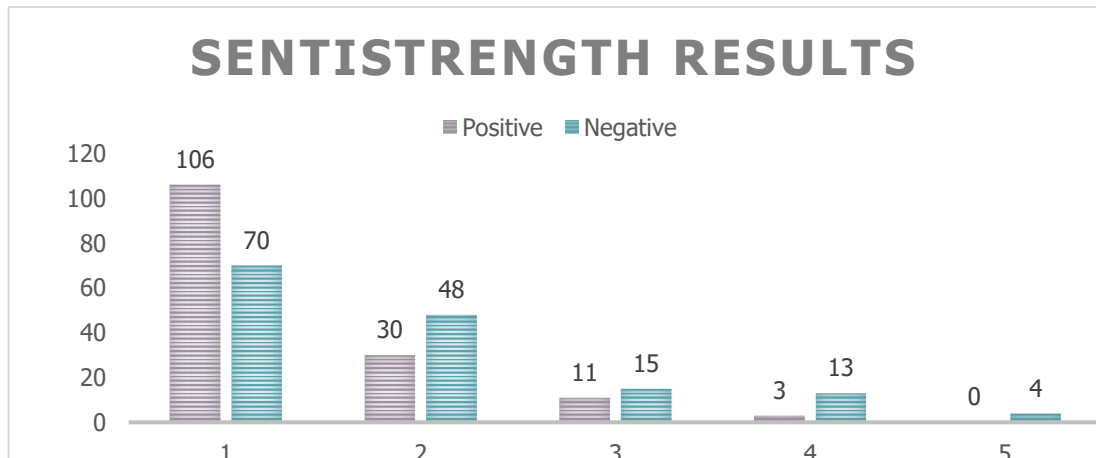


Figure 5-5: Comparison of Positive and Negative sentiment

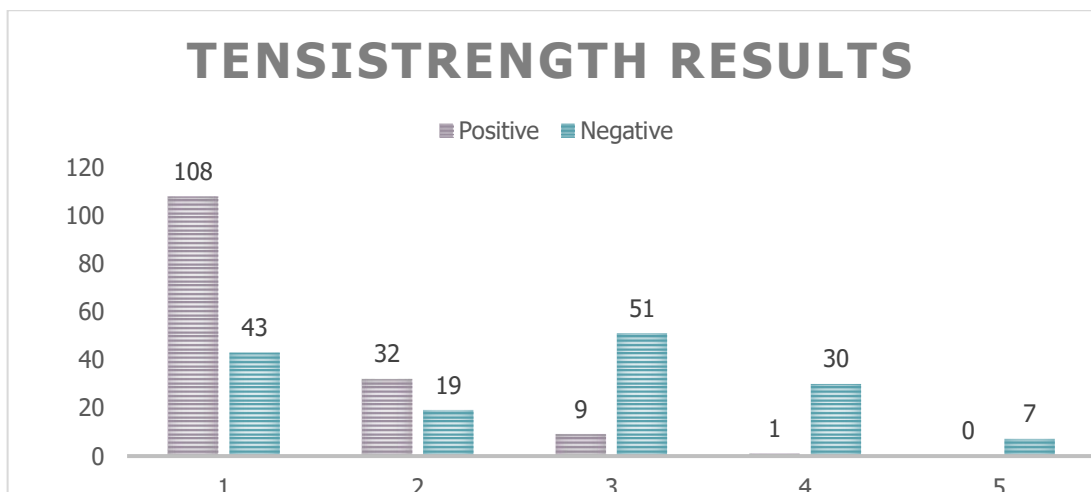


Figure 5-6: Comparison of Stress (negative) and Relaxation (positive)

5.6 Summary

This chapter presented the social media component that was developed as part of the traffic management framework proposed by this research. This chapter presented a detailed system architecture for classifying, geolocating and extracting emotions from Twitter. The dataset for this experiment consisted of tweets from a three-month period around the West Midlands region. From the five machine learning algorithms compared,

RRC showed the highest accuracy in both the training and test dataset. In the geolocation task, the custom NER model was able to successfully identify location mentions within the tweets with a 95.86% of accuracy. From these location mentions, Google API was able to geolocate them with a 97.12% accuracy, outperforming both Bing and Nominatim engines. Lastly, Sentiment strength analysis identified neutral emotions within the tweets, with both positive and negative sentiment leaning towards 1. In contrast, TensiStrength presented promising results when identifying stress, with more than 60% of the sample showing a significant amount of stress. While the proposed social media component has been tested on Twitter data, it can be adapted to other types of user-generated content from alternative crowdsourcing applications. Moreover, it is a plug-n-play architecture where is possible to interchange any component of the framework for another one more suitable for other purposes. The purpose of this chapter was to identify the most accurate techniques for the real-time implementation of the system. The next chapter presents the real-time implementation of the social media component, using a Ridge classifier as the text classification algorithm, Google API as the geolocation engine, and TensiStrength for stress analysis of the dataset.

Chapter 6: Real-time social mining for TMS applications

6.1 Introduction

Chapter 5 presented the proposed social media component based on different machine learning algorithms, geolocation, sentiment and stress analysis techniques. Results from the testing phase outperformed similar studies in the literature, in terms of text classification and geolocation techniques (as shown in sections 5.4 and 5.5). This chapter further demonstrates the effectiveness of the proposed social media component using a real-time twitter data processing pipeline. The performance of the components is evaluated with a real-time application using the most accurate tools and techniques from chapter 5. The chapter starts by outlining the updated system architecture, based on results and discussions from the case study application, in section 6.2. It then explains the data processing pipeline and the different tools used for data collection. The performance of the system is evaluated using metrics for quantitative evaluation such as accuracy, precision and kappa coefficient. Lastly, the traffic events identified in the dataset were compared with traffic news from official news websites in section 6.3.

6.2 Real-time processing system

With the findings from the testing stage (sections 5.4 and 5.5), the proposed system architecture was revised and updated with the most accurate techniques on each stage of the social media component. The best performing classifier in section 5.4.2, Ridge Regression, was selected as the text classification algorithm. The traffic tweets sample

described in Chapter 5 was employed as the training dataset for the machine learning algorithm. For sentiment strength detection, TensiStrength showed more potential in the identification of emotions in traffic related tweets. Lastly, Google API was designated as the main tool for geolocating the tweets. Figure 6-1 illustrates the revised architecture for the real-time processing system.

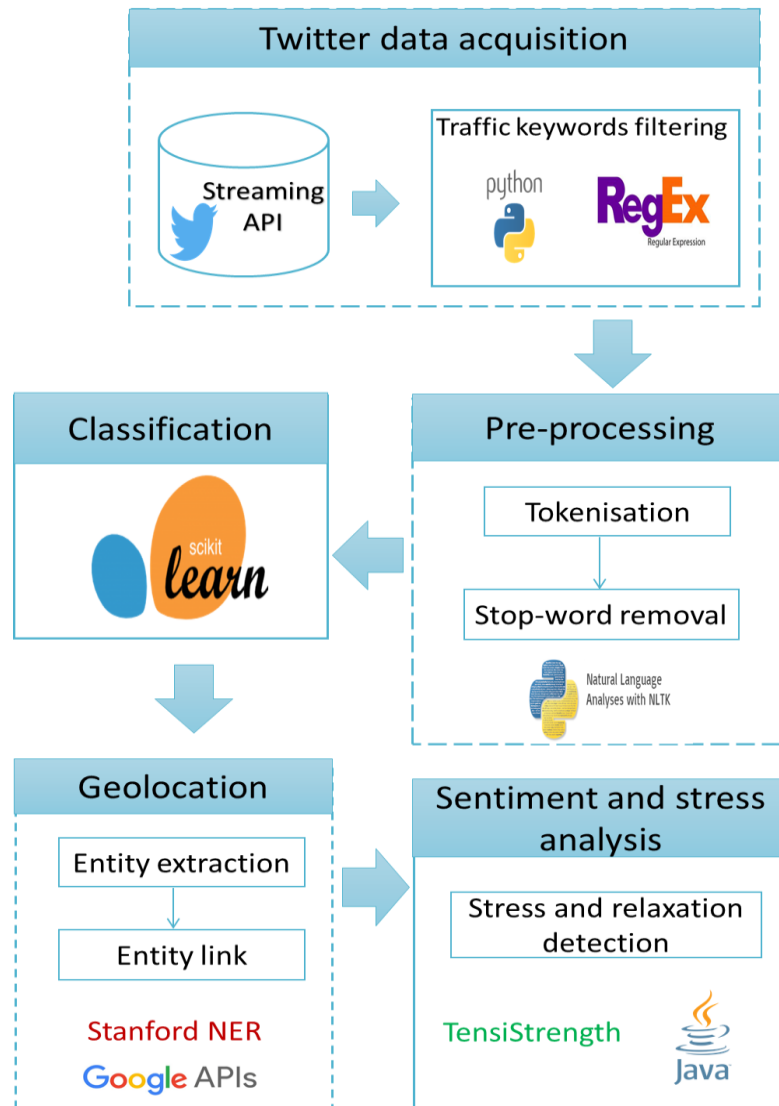


Figure 6-1: Revised system architecture for social media component

The revised system was implemented as real-time through a data processing pipeline using a MongoDB¹⁴ database for storage, and Apache Kafka¹⁵ as the messaging platform. Apache Kafka is a streaming platform that allows to publish, store and process a stream of records as they occur. It is generally used to build real-time processing pipelines that process data through different applications. The employed processing pipeline obtained real-time tweets from the Twitter Streaming API and further published them into a Kafka topic using the Kafka Producer API. The Kafka Consumer API consumed the real-time tweets received in the Kafka topic and used a regular expression filter to check whether the tweet contained a traffic related keyword. If it contained a traffic related keyword, the tweet was then processed through the pipeline and stored in the MongoDB database. The processing pipeline performed the following operations:

- Step 1.* Remove emoticons, hashtags, URLs, and stop words, and convert the tweet into a set of tokens.
- Step 2.* Classification of the tweet using a Ridge Regression classifier. If the tweet is non-traffic related, go to step 6.
- Step 3.* Stress and relaxation detection using TensiStrength
- Step 4.* Location extraction with a custom train NER
- Step 5.* Obtain the coordinates of the traffic event using Google API
- Step 6.* Store the information in the MongoDB database

¹⁴ <https://www.mongodb.com/>

¹⁵ <https://kafka.apache.org/>

The processing pipeline was implemented for a period of 37 days, starting the 21st of September 2018. The regular expression filter obtained 1653 tweets mentioning traffic related keywords. From these tweets, the Ridge Classifier identified 394 as traffic related and 1259 as non-traffic related. To understand the performance of the system, tweets were manually re-labelled in traffic and non-traffic related. In the traffic related sample, 285 were found to be true positive, while 109 were considered false positive. For the non-traffic related class, 1233 tweets were true negative, and 26 were found to be false negative. These initial results show that while the classifier faced challenges when correctly identifying traffic related tweets, it had very good performance in detecting noise. Table 6-1 illustrates the confusion matrix of the results discussed above. The sum of the rows portrait the actual traffic and non-traffic related tweets after manually labelling the dataset, while the sum of the columns represent the tweets identified by the algorithm on each class.

Table 6-1: Confusion matrix of the classification task

		Predicted result		
		Traffic	Non-traffic	
Actual result	Traffic	285	26	311
	Non-traffic	109	1233	1342
		393	1259	

Table 6-2: Performance measure from real-time implementation

Class	Class precision	Class recall	Class F1	Accuracy	Kappa
Traffic	72.26%	91.61%	80.80%	91.83%	0.76
Non-traffic	97.93%	91.88%	94.81%		

Table 6-2 depicts the class precision, recall and overall accuracy of the classification algorithm. The algorithm had a precision of 72.26% and a recall of 91.61% in the traffic related class. This shows that while it returned many results in this class, some of the predictions were incorrect when compared to the actual labels. The non-traffic related class achieved a high precision and recall, 97.93% and 91.83% respectively, which relates to the fact that it returned many results with almost all of the predictions labelled correctly. The overall accuracy of the machine learning algorithm was 91.83%. However, accuracy can be misleading in an imbalanced class problem. An imbalanced class problem occurs when a class is represented by a large majority, while the other has few instances (Chawla, Japkowicz and Kotcz, 2004). Most classification problems do not have exactly the same number of instances on each class, but a small difference is not significant. In this dataset, 80% of the tweets are linked to the non-traffic related class, whereas the traffic related class constitutes a 20% the total dataset. This is an imbalanced dataset, where the ratio between classes is 80:20, or more simplified 4:1. In this case, the accuracy only reflects the accuracy in predicting the class with higher proportion (non-traffic), rather than the overall performance of the algorithm. Abma (2009) argues that precision, recall and F1 can give a better understanding of the performance of the classifier in an imbalanced dataset problem. However, there are other statistic measures that can provide more insights on the accuracy of the model. Kappa, or Cohen's Kappa (Cohen, 1960), is a statistic measure of the accuracy for imbalanced datasets. Originally, it was used for measuring the level of agreement between two observers of psychological behaviour, compared to the agreement expected by chance. More recently, it has been found that Cohen's Kappa can be used to measure the accuracy of a classifier by evaluating the degree of agreement

(accuracy) between the classifier and reality (Ben-David, 2008). It describes how well the classifier performs compared to above the baseline of performance. Cohen's Kappa is defined as:

$$k = \frac{p_o - p_e}{1 - p_e}$$

Where p_o is the observed agreement, or accuracy in this case, and p_e is the expected agreement. Cohen's kappa ranges between values of -1 to 1, but research findings have concluded that most classifiers in real-world dataset, tend to score values higher than 0 (Margineantu and Dietterich, 2000). While there is no standardised interpretation of the values of kappa, Landis and Koch (1977) created a scheme to indicate the level of agreement between different values of kappa. Table 6-3 visualises the different ranges of their interpretation of kappa.

Table 6-3: Interpretation of Kappa

Kappa	Agreement
<0	Less than chance agreement
0.01 -0.20	Slight agreement
0.21-0.40	Fair agreement
0.41-0.60	Moderate agreement
0.61-0.80	Substantial agreement
0.81-0.99	Almost perfect agreement

The kappa score of the real-time implementation of the social media component was calculated using the machine learning library ScikitLearn. In Table 6-2, the Kappa score is presented with a value of 0.76, which represents a substantial accuracy in the scale provided by Landis and Koch (Figure 6-2). This result suggests that the Ridge Classifier is performing substantially well, and close to an almost perfect agreement.

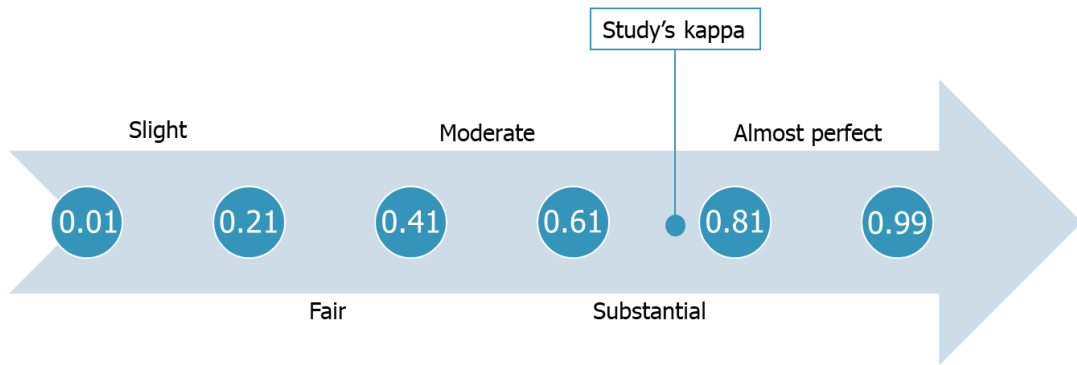


Figure 6-2: Interpretation of the study's kappa score

In terms of stress and relaxation analysis, results were similar to those obtained in test implementation in Chapter 5 (Table 6-4, Table 6-5 and Table 6-6). More than 96% of the tweets had a relaxation score of less than 2, with only 1 tweet obtaining a score of 4. On the other hand, while almost 50% of the dataset obtained a stress strength of more than -3; the mode, median and mean show that tweets in the dataset tended to have low emotions associated with them. When manually inspecting the dataset, it was perceived that some tweets that were assigned a stress strength of -1, in reality expressed some emotion within it that TensiStrength was not able to identify. For instance, the entries: "What a joke is the M6 today" or "The M6 is the biggest car park in Europe", expressed negative emotions towards the M6. However, TensiStrength identified a negative emotion of -1 and a positive of 2, for both tweets. Because it is in the form of sarcasm, it is difficult for TensiStrength to acknowledge it as negative emotions. Sarcasm usually involves using words that while they sound relaxing, they express quite the opposite. As a result, many tweets in the dataset were assigned low values (-1), as the negative emotions were commonly expressed through sarcasm.

Table 6-4: Frequency table for positive emotions

<i>Score</i>	Frequency	Percent	Valid Percent	Cumulative Percent
1.00	223	78.2	78.2	78.2
2.00	50	17.5	17.5	95.8
3.00	11	3.9	3.9	99.6
4.00	1	0.4	0.4	100.0

Table 6-5: Frequency table for negative emotions

<i>Score</i>	Frequency	Percent	Valid Percent	Cumulative Percent
-5.00	8	2.8	2.8	2.8
-4.00	36	12.6	12.6	15.4
-3.00	77	27.0	27.0	42.5
-2.00	39	13.7	13.7	56.1
-1.00	125	43.9	43.9	100.0

Table 6-6: Measures of central tendency for TensiStrength

Measure	Positive	Negative
Mean	1.2632	-2.1684
Median	1.0000	-2.0000
Mode	1.00	-1.00

The custom NER identified the location of 250 entries from the 285 traffic related tweets identified by the machine learning classifier. The top 100 word-cloud inspections for the identified locations within the tweets are shown in Figure 6-3. Words with larger font sizes are more weighted features than the ones with smaller font sizes. More than 188 location mentions were associated to major motorways (e.g. M6, M40, M42), while the others were spread between arterials and urban roads, roundabouts and cities. The M6 was the most mentioned motorway with more than 86 entries associated with it. Figure 6-4 illustrates the stress strength perceived by TensiStrength on the motorways mentioned on the dataset. The M40 had the most

stress associated with it, with almost 60% of the mentions having a stress of more than -3. When manually assessing the tweets, it was perceived that this increase in stress was due to a lorry that caught fire on the M40. Similarly, increases of stress on the M6 were related to complaints about roadworks being carried out in specific sections of the motorway. These findings support the idea that the emotions detected on traffic tweets can be linked to specific incidents or complaints taking place in specific sections of the transport network.



Figure 6-3: Top 100-word cloud inspection

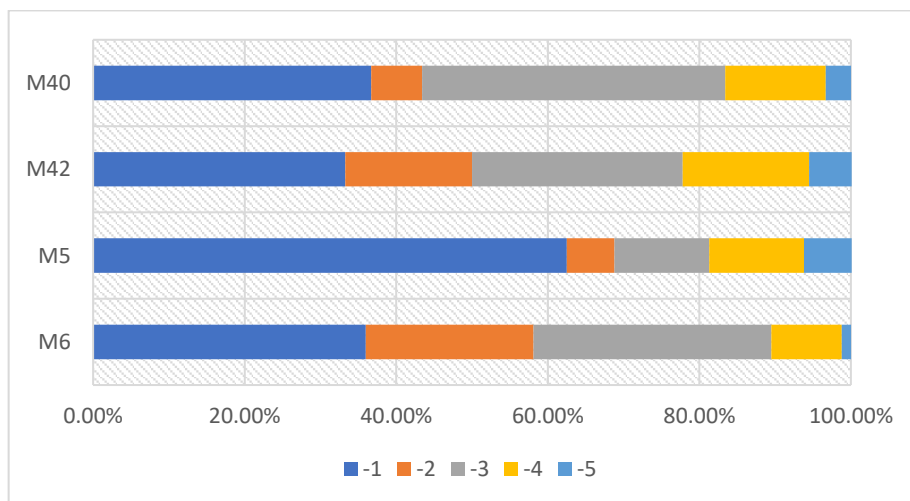


Figure 6-4: Stress perception on specific motorways (percentage)

When it comes to geolocating, Google API was capable of finding a location for the 250 entries identified by the NER. Some of the issues associated with the testing stage in chapter 5 were also found on this dataset. The most notable one was the fact that many users still do not identify in which part of the motorway or arterial road they are at. While they specify that there is some traffic event taking place at the M6, it is not possible to identify to which junction they are referring to. And even if they do, in some instances, Google maps does not have a specific location assigned for junctions and returns a set of coordinates corresponding to the general location of the motorway. Future research could test the influence of developing a custom knowledge base that includes the coordinates of all the junctions within the motorways in the UK, thus it would be possible to identify the exact coordinates of the incident. The coordinates obtained from the Google API were geolocated in the map in Figure 6-5. The points in red symbolise a stress of -4 or -5, yellow indicates a stress strength of -3, while green represents a score of -1.

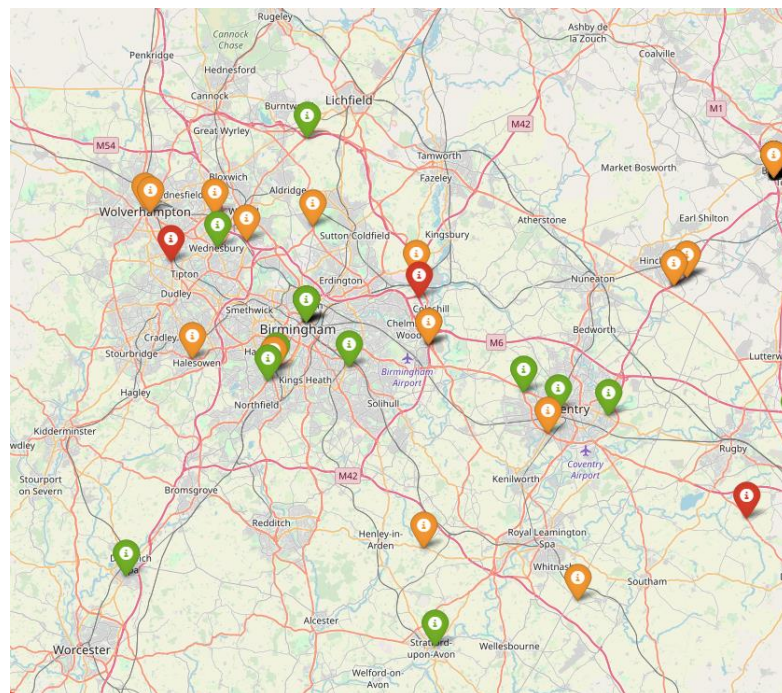


Figure 6-5: Geolocation results linked to stress analysis

6.3 Validation

With the purpose of confirming the veracity of the traffic events detected in the dataset, each event is properly validated. According to D'Andrea *et al.*, 2015, validation can be performed by comparing data obtained from social media against:

- Reports from police and fire departments;
- Official Twitter account of traffic management agencies;
- Real-time traffic news from official channels;

When selecting the most appropriate source for validating the dataset, the nature of the different options was evaluated. Reports from police and emergency services (e.g. fire department) are ideal as this would provide the exact time when the traffic event was detected. However, these reports are either confidential or not available after a specific timeframe. In the case of the official Twitter accounts of traffic management agencies (e.g. Highways West Midlands), tweets are posted after the traffic event has already been processed and police/emergency services are already in the scene. This would mean that the time of the tweet does not necessarily relate to the time when the incident was detected. Official news websites (e.g. local newspapers) offer real-time information of the different events happening across the transport network. In fact, it has been observed that official news websites gather evidence from police, emergency services and traffic management agencies, and publish this information on real-time in their websites (D'Andrea *et al.*, 2015). They tend to provide a timeline of the event that usually includes time of the incident, time it was notified to emergency/police and official traffic management agency statement. For these reasons,

it can be concluded that traffic news websites are a reliable traffic event source of information.

Traffic news from local websites such as Birmingham Mail, Daily Mail, Coventry Telegraph and Warwickshire news, were used for validating the traffic events obtained from Twitter during the real-time implementation. In Table 6-7, the major traffic events identified in Twitter are validated with information from the local news. The time from the 'Twitter' column refers to the first tweet obtained related to the incident, while the time in the column 'News' refer to the time where the emergency services call was made. It can be perceived that tweets had always a delay of at least 15 minutes, when compared to the time that the incident was first detected. These results confirm findings from the literature in regards of Twitter being more a tool to support TMS, rather than an incident detection system (Schulz, Ristoski and Paulheim, 2013; Zhang, *et al.*, 2018). While many studies have concluded that people are motivated to share real information by the desire to help others (Ghaisani, Handayani and Munajat, 2017), not all travellers tweet when they are passing a specific incident for their own traffic safety. These results suggest that the purpose of the social media component within the proposed architectural framework, leans towards the identification of issues and perception of the road network, rather than an incident detection tool itself. Future work should consider using the full firehose (PowerTrack API) during the data gathering stage, with the purpose of further analysing the suitability of Twitter data for the earlier detection of traffic events.

Table 6-7: Validation of traffic events detected from Twitter

Traffic event	Detection of the event		
	Twitter	News	Difference
Lorry fire in the M40	03:23	02:49	+34min
M6 accident involving lorry and pedestrian	07:46	06:25	+81min
Cows on the M6	19:10	18:55	+15min
M6 car crash	09:23	08:40	+43min
M42 car crash	01:47	01:17	+30min
Multicar crash on the M6	09:59	09:30	+29min
Road traffic incident on M69	13:31	12:45	+74min
Road traffic collision M42	00:08	23:00	+68min
Road traffic incident M40	17:05	16:38	+27min

One of the most observed characteristics of a sample of tweets in the dataset was the sentiment affiliated with the desire to help others. Indeed, many tweets were associated to delays in the motorways from the previous outlined accidents. However, there was a sample of tweet dedicated to highlight flaws of the transport network to official traffic accounts. Many users tweeted about traffic lights not working, matrix signs not updated, unexplained traffic events or unscheduled roadworks. Table 6-8 illustrates a sample of tweets in the dataset devoted to report these incidents. The majority of these were about issues on arterial and urban roads. As highlighted in the literature review, most of the existing TMS have been developed for highways, not for arterial roads. These types of tweets could support TMS in the detection of issues in areas of the transport network where there is less coverage. Moreover, this data can potentially help in identifying specific traffic events happening in urban roads, as some of the traffic events described in these tweets do not even reach traffic news. Finally, the information obtained from these users' could be used as feedback for the redesign of existing areas of the network. For instance, some users are complaining about the timings on the traffic lights being out of sync, or the timings not being ideal for the junction. With these complaints, local governments and traffic management agencies

can evaluate and check the validity of the information and derive possible solutions to the reported issue. The complete sample of tweets can be found in Appendix 6-A.

Table 6-8: Users' perception of the transport network

Tweet	Issue
Once again we see the effect of a pedestrian crossing within 20 yards of A45 Kenilworth Road junction, accident waiting to happen!	Pedestrian crossing
Matrix sign A14 w/b at M6/M1 junction not advising of M1 n/b closure. Drivers could avoid if diverted on to M6 for M69	Matrix sign not updated
why don't somebody turn the lights off on the roundabout on the A5-A42 island and let the traffic flow freely ????	Traffic lights timing
Usual awful traffic management between 23 and 25 of the M1 - nobody working!	Roadworks
M1 sth jcn 11-10 Matrix showing Ln 1 closed, all fully open. Satellite delay again??	Satellite delay
So the M6 has roadworks on it and 40 min delays. When do you find this out? When you get to it.	Roadworks
Traffic lights out on Binley Road at the cross junction, drive safely everyone, nearly got took out by a bus	Traffic lights not working
Who is responsible for the signaling on M1 J24 A453 island? This is a complete joke	Signalling
A453 at junction 24 of m1. Traffic lights out of sync. Miles of queues on a453 southbound. Can you sort please	Traffic lights
Stuck in standstill traffic with no explanation on the A1. Any ideas?	Unrecorded traffic event
#M6 J10 closed roadworks once again. No signage during the build up to tonight's work	No signage
Disappointed to see that the light on the sign at Pirehill is still flashing. Enough flashing lights on the A34 at the moment!	Traffic light not working
There is a broken down van on Five Ways roundabout currently causing traffic chaos!	Traffic incident
Please AVOID COMPTON this evening. Four way traffic lights at bottom of Holloway and traffic gridlock	Traffic lights not working
Matrix sign at j15 M1 says j13 20 mins, google and Waze saying M1 blocked at j14 southbound with 57min delay	Matrix sign not updated
Hi team. Trying to get from Birmingham to Worcester but there seems to be no diversion from M42 - M5. Can you help?	Matrix sign not updated

6.4 Summary

This chapter presented results from the analysis of the real-time operation of the social media component. The real-time processing system was presented, using a Ridge Regression as a classifier, Google API for geolocation and TensiStrength for stress analysis. The processing pipeline obtained 1653 tweets mentioning traffic related keywords in a total of 37 days. From these tweets, the classifier labelled 393 as traffic related and the rest as non-traffic related, with an overall accuracy of 91.83%. While the classifier had a high precision and recall identifying non-traffic related tweets, the precision identifying traffic tweets was only 72.26%. Because the accuracy only reflects the high precision in detecting non-traffic related tweets due to having an imbalanced dataset, the Kappa score was proposed to further validate the findings. The kappa score of the classifier was 0.76, which relates to a substantial accuracy, according to studies in the literature. When analysing the stress in the dataset, it was perceived that TensiStrength was not able to capture very well the sentiment expressed in the tweets. Mainly, because many of the users expressed themselves with sarcasm, and it poses a challenge to capture emotions from these messages. For this reason, TensiStrength did not identify stress in more than 43% of the dataset. In terms of geolocation, the custom NER identified locations in more than 87% of the sample, with the most mentioned motorway being the M6. The stress identified by TensiStrength was mapped to the motorways in the dataset, with the M40 having the most associated stress to it. Lastly, validation was performed to confirm the events identified in the dataset with local news. The major events identified on Twitter were mapped to news report with a minimum delay of 15 minutes, further confirming findings in the literature about Twitter not being suitable for automatic incident detection. More interestingly, it was found that a sample

of tweets in the dataset were about users reporting issues related to flaws or complaints about the traffic network. It was concluded that this information can be beneficial for local governments and traffic management centres. In order to find the most appropriate actions to ease congestion, traffic incidents or issues identified from the social media component could be simulated in the simulator component of this research. The next chapter introduces the simulator component, along with the proposed imputation methodology for missing traffic data.

Chapter 7: Optimisation-based traffic data imputation

7.1 Introduction

After testing and implementing the social media component in chapters 5 and 6, this chapter presents the simulator component. This component has two main purposes: support traffic prediction and provide real-time simulations of the transport network. While there is availability of real-time traffic information through Webtris, some of these sensors are either providing incorrect readings or not sending information at all. Moreover, this traffic information corresponds to highways, whereas there is no traffic information from the arterial roads that these highways are connected to. The simulator component aims to support traffic prediction by estimating the missing flow distribution on roundabouts connecting an arterial road and a highway. After the model has all the necessary traffic information, real-time simulations of the transport network can be performed. In order to prepare the simulation model for the above tasks, there are a number of features that need to be set and specified in the traffic simulation software (AIMSUN). This chapter contains the description of the modelled network, as well as the different steps undertaken to set up the model, in sections 7.2 and 7.3. Section 7.4 outlines the methodology for the imputation of missing roundabout flow data using constrained optimisation. Lastly, results and findings from the different experiments are presented in section 7.5.

7.2 Model description

The network modelled for the simulator component of this research represents the section of motorways that surround the city of Birmingham. Figure 7-1 shows the full network of the city of Birmingham versus the network modelled for the simulations. The sections in blue on the complete Birmingham network are those motorways that were created in AIMSUN for the purpose of this research.

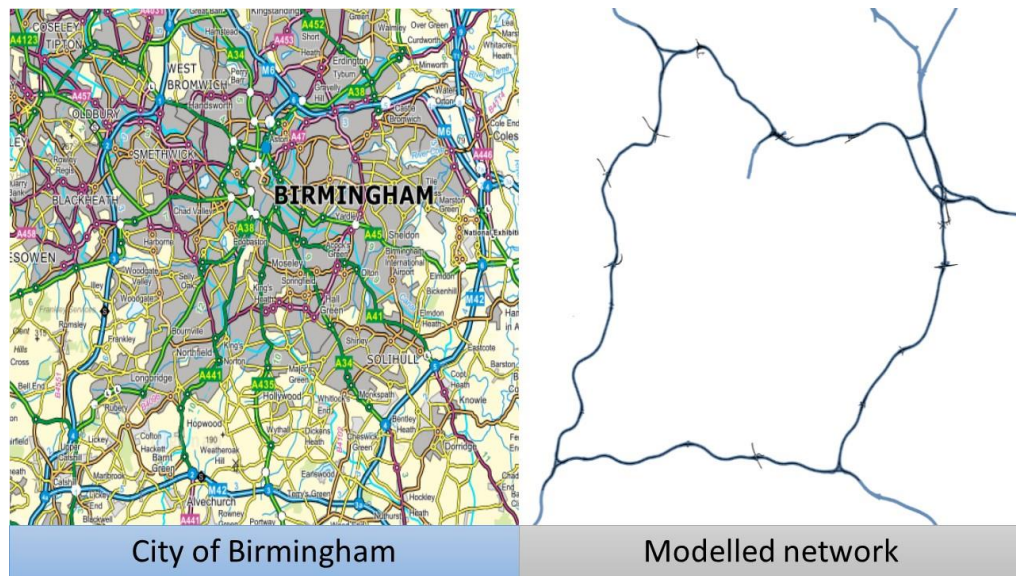


Figure 7-1: City of Birmingham vs Modelled network

The modelled network has about 301km of road, 18 major junctions and 551 sections. These sections include the M6, M6 Toll, M5, M42 and M40 motorways, as well as 18 arterial roads that intersect with the previous motorways. The whole network contains a total of 893 sensors, with some sections reaching 16 sensors while others do not have any sensors associated with it. A summary of all the features of the network can be found in Table 7-1.

Table 7-1: Features of the modelled network

Length	301km
Sections	551
Intersections	210

Junctions	18
Sensors	893
Motorways	5
Arterials	12

From the 18 junctions, there are five that are considered major junctions as they represent two or more motorways and arterials intersecting, while the others are roundabouts of a motorway meeting an arterial road. Figure 7-2 and Figure 7-3 have screenshots of the major intersections and roundabouts that can be found in the model. While major intersections possess traffic information from the main motorways in the model, there is no traffic information associated with roundabouts in the network for both the incoming traffic from the arterial road, and the distribution inside the roundabout. This case of missing traffic data affects the development of a realistic simulation model.

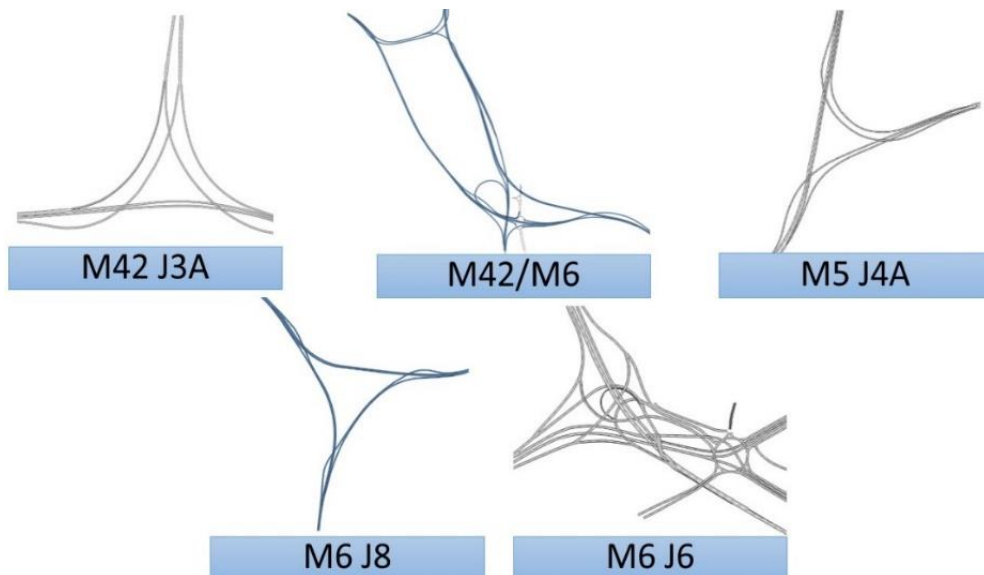


Figure 7-2: Major intersections in the model

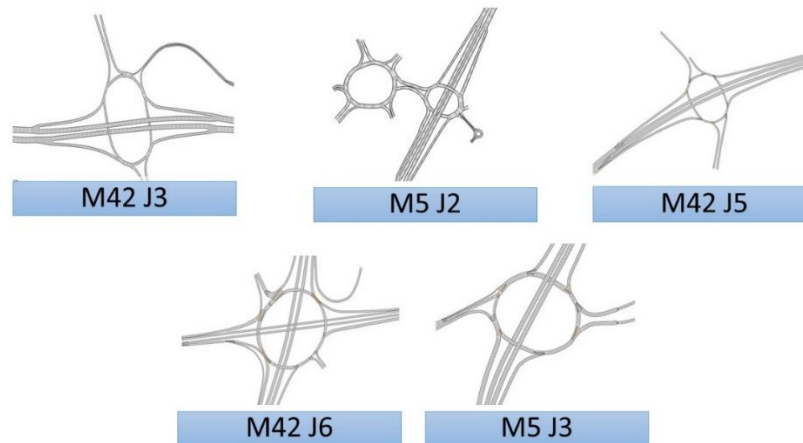


Figure 7-3: Examples of roundabouts in the model

7.3 Preparing the network

One of the main advantages of AIMSUN is that it allows its users to modify the model, import/export data, perform calculations, and run simulations through Python scripts and the AIMSUN API. The following subsections describe the different actions undertaken in this research to set up the model using scripting in the Python programming language.

7.3.1 Placing detectors

It is possible to manually place sensors in the graphical interface of AIMSUN. However, this would be a difficult task to perform manually as there are 893 sensors in the modelled network. In addition, it would pose a challenge to locate these sensors as the graphical interface does not allow you to locate them using coordinates. To cope with the above, this research created an automated process through a Python script that performed the following actions:

- Step 1.* Loop through all the sensors with a geolocation filter using the coordinates of the model.

Step 2. For each sensor, iterate through all the road sections until it finds the closest to the sensor coordinates.

Step 3. For the closest section, iterate through all the points in the section until it finds the closest point to the sensor coordinates.

Step 4. Create a sensor, adding the name and its appropriate description.

Sensor information (e.g. name, coordinates) were stored in a MongoDB collection, and queried from the Python script within AIMSUN. The above process successfully placed 893 sensors in the modelled network, along with the corresponding name and description. After all the sensors were placed, they were manually checked to find those that were misplaced. Those misplaced sensors, where either allocated to the corresponding section or eliminated from the model. The latter happened when the sensor was not associated with any section within the model. Figure 7-4 shows a detector on the network and its attributes. The python script created to locate the sensors can be found in Appendix 7-A.

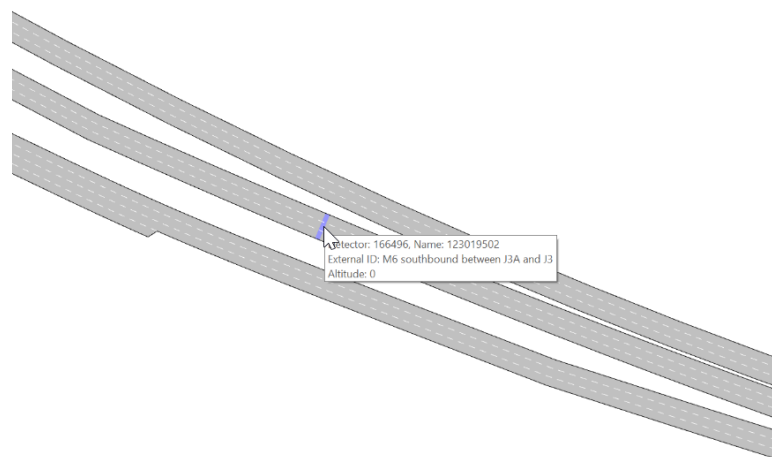


Figure 7-4: Example of a detector and its corresponding description

7.3.2 Importing traffic data and running simulations

The traffic demand in a network constitutes the number of vehicles travelling through each road section, and the corresponding turns at the road junctions. AIMSUN offers two means of importing traffic data into the model: Traffic States and OD matrices. The first, a traffic state is a description of the state in each section of the network. It is composed of a set of flows at every input section in the network and a set of turn proportions at every junction. An Origin Destination (OD) matrix is composed of a matrix describing the number of trips from one centroid to another. A centroid represents a source of vehicle, usually an edge of a network connection. An OD matrix requires an estimate of the number of trips between centroid, which can be derived using observed flow and turn count data. However, this information may be scattered and not available for all turns and sections, as it would entail obtaining the route choice of the different vehicles in the edges of the network. Models based on traffic state are simpler to create and more suitable for management, optimisation and detailed road design. Therefore, traffic states were selected as the traffic demand object for this research.

There are two problems associated with inputting the traffic data from the sensors into a traffic state. First, sensors in the network provide a flow, but not the turn proportion. This would mean having to manually add the flows and turns for every simulation, instead of running them on a real-time basis. It is also normal to find some discrepancies in the traffic data. For instance, according to Figure 7-5, you would expect that $A + B = C$, but this is not necessarily true when you get real data, as there is always a margin of error or missing sensor information. Finally, simulations are triggered manually, and due to the graphical content that it generates, it takes some

time to produce results. For such a large-scale network as the one modelled for this study, it would take a considerable amount of time to run a simulation using the graphical interface.

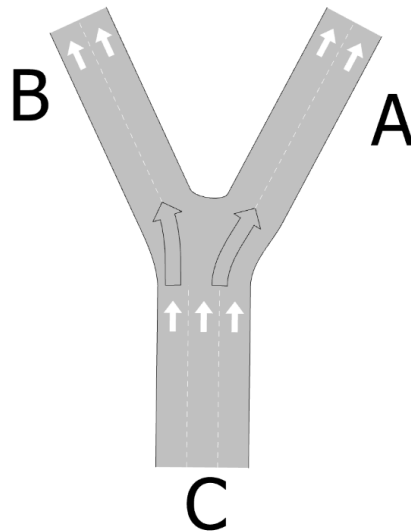


Figure 7-5: Flow distribution example

To tackle the challenges described above, this research created a methodology that imports the sensor data into the traffic states, and automatically triggers the simulation without using the graphical interface. The process can be summarised in the following steps:

- Step 1.* It calculates the error between flows and distribute it equally, adding or subtracting were necessary.
- Step 2.* If a section has no flow, it finds its value by adding or subtracting the flow of the corresponding sections.
- Step 3.* It calculates the turn percentages based on the corrected values.
- Step 4.* Creates a traffic state and imports the entrances/turn percentages into the AIMSUN model.

Step 5. Run the simulation

Step 6. Export the simulation results

The above described process reduces the time of the simulation significantly. In fact, during testing stage, it was observed that a simulation without the graphical interface is 18 times faster than with graphics. Moreover, instead of having to check the information from the sensors manually, this methodology automates the detection of errors in the traffic data. The python script created for this methodology can be found in Appendix 7-B.

7.4 Optimisation-based traffic data imputation

The process described in the previous section works well in major intersection within the model, as they have sensors in most of their sections, which is enough to calculate the missing flow values. However, roundabouts have no entry flow information from the arterial roads that they are connected to nor data of how the traffic splits within the roundabout. It is not a problem of traffic data missing for specific period of times, but rather the unavailability of loop detectors in the specified sections. As a result, roundabouts in the model have at least 12 missing variables needed to simulate the model. As discussed in section 2.3.3, data from all road sections in the model are essential to complete a simulation model, as they require the traffic flow measurements and the turn proportions to be specified as inputs for the simulation. This research developed an imputation methodology to treat the missing data as an optimisation problem, where the values that best satisfy the objective function are searched for. Mathematical optimisation is used to find the best value of a function by taking into consideration some criterion and inputs. The purpose of the proposed

methodology is to automatically find those missing traffic values; thus it would be possible to create a realistic simulation model. Figure 7-6 illustrates the flowchart of the proposed imputation approach. The main steps are summarised as follows:

- Step 1.* Generate random initial values for the missing variables.
- Step 2.* Missing data are estimated using Sequential Least Squares Programming (SLSQP).
- Step 3.* Convert results from optimisation into turning percentages.
- Step 4.* Update traffic states input (urban roads) and turning percentages (roundabouts).
- Step 5.* Calculate Mean Absolute Percentage error (MAPE) between estimated data and actual data.
- Step 6.* Judge whether the stopping criteria ($MAPE < 1\%$) is reached, otherwise return to step 1.

The full process runs inside AIMSUN, through a Python script that can be found in Appendix 7-C. The methodology is divided in two main tasks: optimisation and validation. In the following sub-sections, the activities of these tasks are described more in depth.

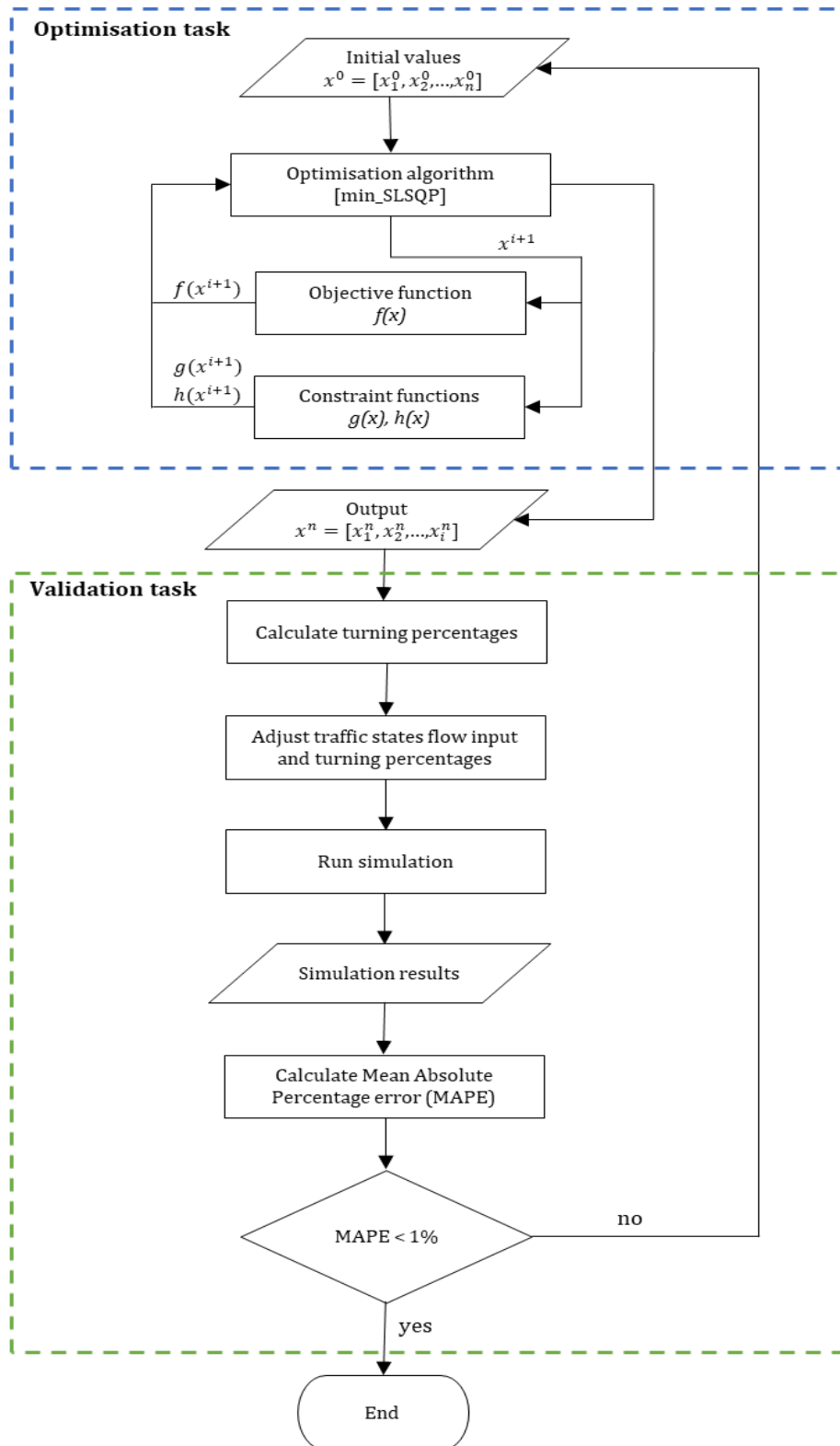


Figure 7-6: Proposed optimisation-based imputation methodology

7.4.1 Optimisation task

The simulator needs as an input the flow from all the entrances to the network, and the split percentages of all the turns. These variables all depend of one another, as split percentages need to sum 100 and the value of one flow affects all the others. There are a wide range of optimisation algorithms available that can find the most optimal solution either by minimizing or maximizing a function. Some of the most commonly used optimisation algorithms are Nelder-Mead simplex, genetic algorithms and differential evolution. However, some of these algorithms need information that would be difficult to provide with this type of problem, such as the hessian and gradient. To take these limitations into consideration, a constrained minimisation algorithm was selected. Constrained algorithms can optimise a function with respect of some constraints on those variables. In this research, Sequential Least Squares Programming (SLSQP) optimisation subroutine originally implemented by Kraft (1988), was chosen as the algorithm for the optimisation task. SLSQP minimises a function of several variables with any combination of bounds, equality and inequality constraints. This algorithm allows to deal with constrained minimisation of the form (Kraft, 1988):

$$\begin{aligned} & \min f(x) \\ & \text{Subject to} \quad g_j(x) = 0, \quad j = 1, \dots, m_e \\ & \quad \quad \quad g_j(x) \geq 0, \quad j = m_e + 1, \dots, m, \\ & \quad \quad \quad x_l \leq x \leq x_u \end{aligned}$$

where m is the number of equality and inequality constraints, and m_e represents the number of equality constraints.

Kitschke (2014) explained it by reformulating the problem using the Lagrangian method and function $L(x, \lambda)$:

$$L(x, \lambda) = f(x) - \sum_{j=1}^m \lambda_j g_j(x)$$

$$\min_{x, \lambda} L(x, \lambda) \text{ such that } x_l \leq x \leq x_u$$

The reformulated optimisation problem is solved by using a quadratic approximation of the Lagrange function and a linear approximation of the constraints:

$$\min_{d \in \mathbb{R}^n} \frac{1}{2} d^T B_k d + \nabla f(x_k) d$$

such that

$$\nabla g_j(x_k) d + g_j(x_k) = 0, \quad j = 1, \dots, m_e$$

$$\nabla g_j(x_k) d + g_j(x_k) \geq 0, \quad j = m_e + 1, \dots, m$$

where $B := \nabla_{xx}^2 L(x, \lambda)$

In each iteration, the SLSQP algorithm minimises the quadratic approximation of the Lagrange function by solving the equivalent linear least squares problem:

$$\min_{p \in \mathbb{R}^n} \| D_k^{\frac{1}{2}} L_k^T p + D_k^{-\frac{1}{2}} L_k^{-1} \nabla f(x_k) \|^2$$

such that

$$\nabla g_j(x_k) d + g_j(x_k) = 0, \quad j = 1, \dots, m_e$$

$$\nabla g_j(x_k) d + g_j(x_k) \geq 0, \quad j = m_e + 1, \dots, m$$

with $B = LDL^T$. For more examples of such problems see Kraft (1988).

The objective function for the proposed optimisation routine is the residual sum of squares (RSS) between the estimated data and the actual data, while the constraints are equality equations derived from the dependency of the variables in a roundabout. In the following sections, the different constraints and parameters used for the implementation of the optimisation routine are presented more in detailed.

$$RSS = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

7.4.2 Validation task

There are a wide range of possible solutions that can satisfy the conditions set in the optimisation routine. However, it is necessary to find the right combination of values that will generate low discrepancy between simulated and real data. This stage involves inputting the results obtained from the optimisation task into the simulator and comparing the outcomes from the simulation with real data. The first step encompasses adjusting the traffic states. Variables obtained from the optimisation routine will be converted into the right format to adjust the existing traffic state. After the traffic states have been adjusted, the traffic simulation is triggered. The above would be achieved with the python script presented in in section 7.3.2, which calculates the turn percentage from a set of flows and simulates the network without the graphical interface. Results from the simulation are exported and compared with the real dataset. This is achieved by calculating the Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100\%}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right|$$

where A_t is the actual value, and F_t is the simulated one. This is an iterative stage, where if the MAPE is more than 1%, the optimisation routine will generate a new set of values and run the simulation with the new values. Until reaching this criterion, the process will not stop. Every instance where the MAPE is less than 1%, would be considered one iteration. The last iteration of the optimisation process will happen when the error of both variables is less than 1%.

7.4.3 Experiments

The section of the network used for the experiment represents a 4.5km section of the M42 motorway in Birmingham, United Kingdom. M42 is a major motorway that runs from Bromsgrove (Worcestershire) to Ashby-de-la-Zouch (Leicestershire). The model features two consecutive roundabouts: M42 Junction 4 and M42 Junction 5. In junction 4, the M42 meets Stratford road, while in Junction 5 is where it encounters the Solihull A41. Users take both roundabouts to go to different districts of the Solihull town. Figure 7-7 magnifies the location of the modelled network, as well as the junctions within it.

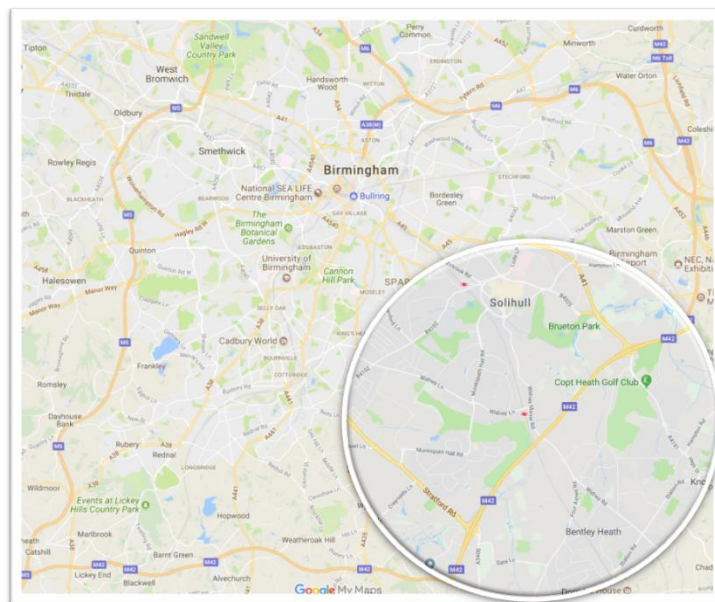


Figure 7-7: Location of the experiments

Figure 7-8 outlines a detailed flow chart of the optimisation-based imputation methodology adapted to the M42 J5 roundabout. As it can be appreciated in the figure, the roundabout has 12 missing variables that correspond to the entrances and exits from the A41, and the turnings inside the roundabout. The only known values are those that correspond to the exits (C_1 and C_3) and entrances (C_2 and C_4) from the motorway (M42). As explained in section 7.4.1, the objective function selected was the RSS between the estimated data and the actual data. In this example, the RSS function uses the flow from the exits as performance measures (C_1 and C_3), while traffic data from the entrances (C_2 and C_4) were used as input for the equality constraints. The known exits of the system (C_1 and C_3) were assumed as two unknown variables that the optimisation algorithm had to find (X_{13} and X_{14}). The equality constraints were derived based on the dependency that the variables have on the different intersections of the roundabout. For instance, in the intersection numbered 1 (see Figure 7-8), the incoming flow of X_1 and X_5 needs to be equal to the flow in X_6 . The same logic applies to the other seven intersections on the system, which leads to a total of eight constraints in the roundabout. After establishing the optimisation problem, based on the objective function $f(x)$ and the eight constraints; a set of random values is generated for all the missing variables. These values constitute the initial set that is used to trigger the optimisation algorithm to find the actual values that minimise the objective function. The outputs from the optimisation algorithm are then imported into AIMSUN, and the simulation is initiated. With the simulated outputs of C_1 and C_3 , the MAPE is calculated for both variables. Only if both $\text{MAPE}(C_1)$ and $\text{MAPE}(C_3)$ are less than 1%, the procedure terminates. Otherwise it returns to the optimisation task, where another set of initial values is generated and the optimisation

algorithm find another set of values for the missing variables. In the following subsections, the objective function, the different constraints and the results from the experimental implementation are explained more in detail.

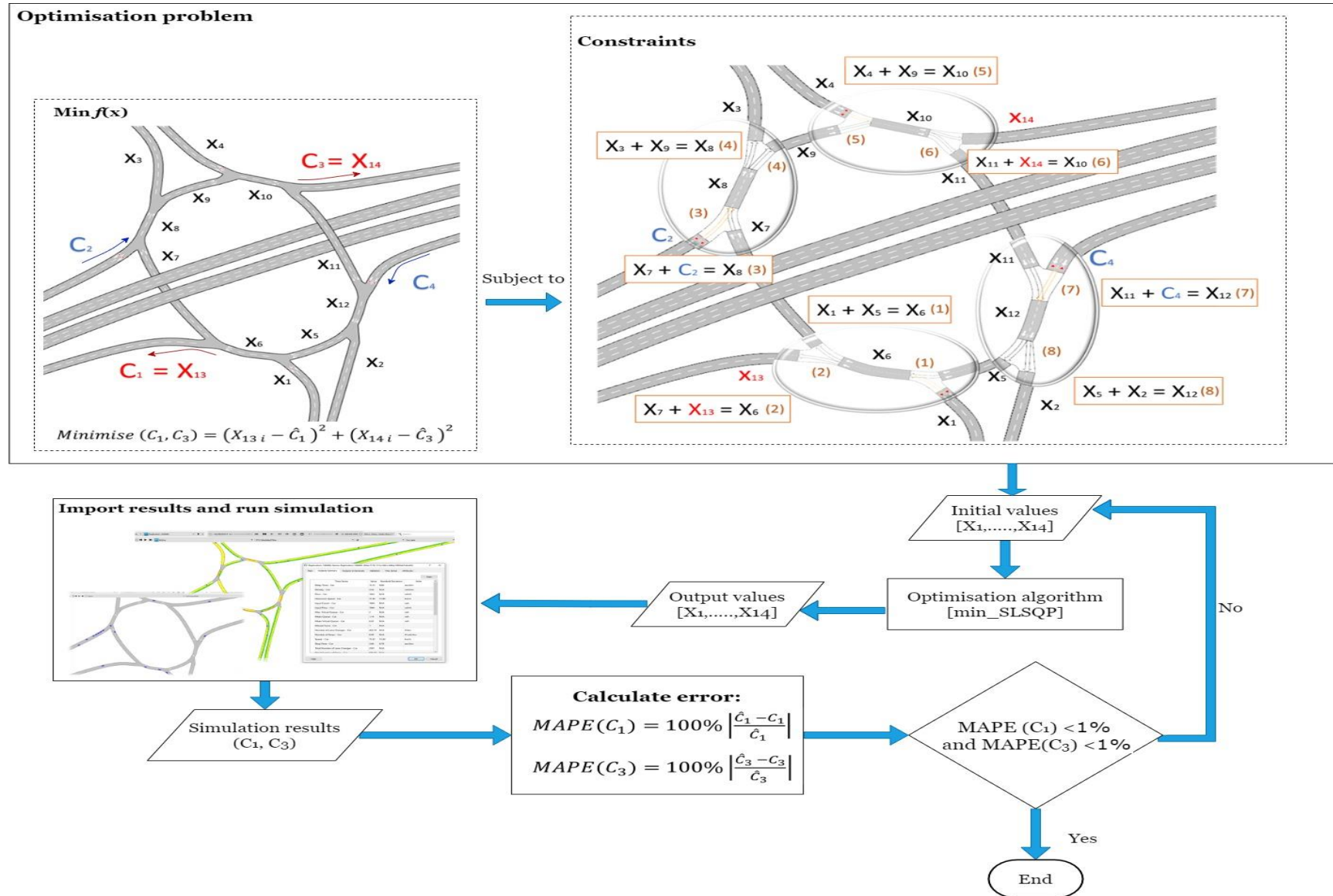


Figure 7-8: Optimisation-based traffic data imputation detailed flow chart

7.4.3.1 M42 Junction 5

M42 Junction 5 is where the M42 motorway meets the Solihull A41. Users take this roundabout to go directly to Solihull town centre. It is in the Mid-East area of the AIMSUN network. Figure 7-9 shows the location in the model and the view from Google maps.

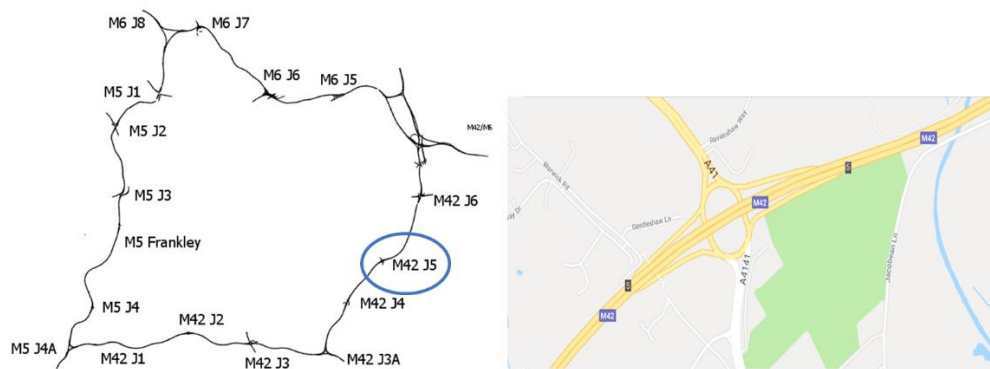


Figure 7-9: M42 J5 location in AIMSUN and Google maps

The roundabout has 12 variables that correspond to the entrances and exits from the A41, and the turnings inside the roundabout. Constraints are based on the concept that in a turn there are three variables dependant on each other. Two of them, must add up to the third one. For instance, Figure 7-10 illustrates the flow distribution in an intersection of the roundabout. From this example, it can be derived an equality constraint based on the dependency of x_{10} , x_{13} and x_4 (see Figure 7-10). The constrained is based on the concept that the sum of both proportions (x_{10} and x_{13}) is equal to the input flow (x_4) on this intersection. Using this logic, eight constraints were elaborated based on the dependency of the variables on each intersection. Figure 7-11 shows the distribution of variables in the roundabout and the eight equations used as constraints for the optimisation problem.

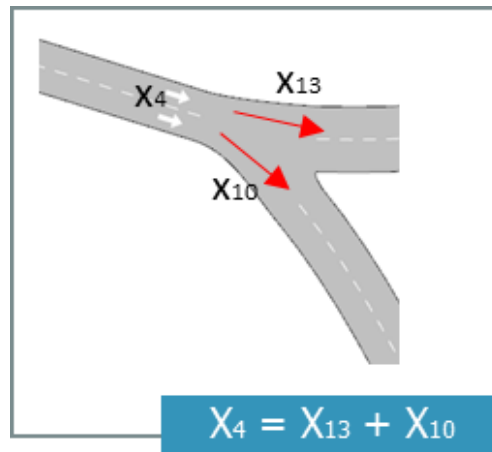


Figure 7-10: Turn proportion on intersections within the M42 J5

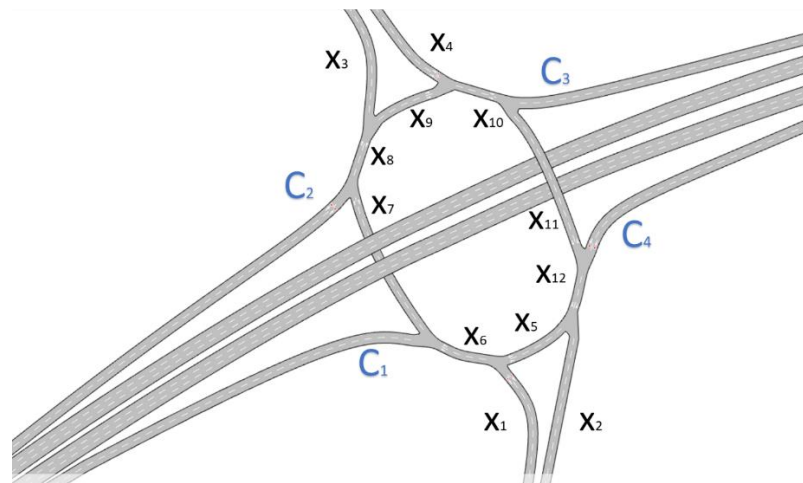


Figure 7-11: Variable distribution for M42 Junction 5

In this study, the objective function, RSS, uses the flow from the exits (C_1 and C_3) as performance measures. Traffic data from the entrances (C_2 and C_4) were used as input for the equality constraints, while C_1 and C_3 were assumed as two unknown variables (X_{13} and X_{14}). The optimisation problem can be summarised as follows:

$$\text{Minimise } (C_1, C_3) = (C_{1i} - \hat{C}_1)^2 + (C_{3i} - \hat{C}_3)^2$$

where

$$\hat{C}_1, \hat{C}_3 = \text{actual value of } C_1 \text{ and } C_3$$

C_{1i}, C_{3i} = predicted value of C_1 and C_3

i = number of the iteration

such that

$$X_1 + X_5 - X_6 = 0 \qquad X_9 + X_4 - X_{10} = 0$$

$$X_6 - X_7 - X_{13} = 0 \qquad X_{10} - X_{11} - X_{14} = 0$$

$$X_8 - X_7 - C_2 = 0 \qquad X_{12} - X_{11} - C_4 = 0$$

$$X_3 + X_9 - X_8 = 0 \qquad X_5 + X_2 - X_{12} = 0$$

Traffic data was obtained from the sensors corresponding to C_1 , C_2 , C_3 and C_4 on the 25th of January 2017, from 10am to 11am. During the trial stage, it was observed that the algorithm solved the minimisation problem by assigning values of 0 to some of the variables. As a result, the initial values of the variables for each iteration are random numbers greater or equal to 50. The input flows for the traffic state are C_2 , C_4 , X_1 and X_4 , while the other values were converted into turning percentages. Optimisation terminated successfully after 6 iterations in 22.23 seconds, this is when the value of the MAPE, for the simulated values of C_1 and C_3 was less than 1%. Table 7-2 and Table 7-3 present the initial values, and results from the optimisation routine. The error for C_1 was 0.88%, while for C_3 was only 0.27%. Table 7-4 contains the MAPE for each variable.

Table 7-2: Initial values for optimisation

C₁	748 veh/h
C₂	529 veh/h
C₃	570 veh/h
C₄	437 veh/h

Table 7-3: Results from optimisation routine

Optimisation terminated successfully	
Success	True
Time	22.23s
Iterations	6
X₁	478.13 veh/h
X₂	126.44 veh/h
X₃	152.11 veh/h
X₄	152.42 veh/h
X₅	547.66 veh/h
X₆	1025.79 veh/h
X₇	277.79 veh/h
X₈	806.79 veh/h
X₉	654.68 veh/h
X₁₀	807.10 veh/h
X₁₁	237.10 veh/h
X₁₂	674.10 veh/h
X₁₃	748 veh/h
X₁₄	570 veh/h

Table 7-4: MAPE of simulated results

Variable	Actual value	Simulated Value	MAPE
C₁	748	750	0.88%
C₃	570	575	0.27%

7.4.3.2 M42 Junction 4

M42 Junction 4 is where the M42 motorway meets the A34 (Stafford road). Users take this roundabout to go directly to Shirley and other districts of Solihull. Figure 7-12 shows the location in the model and the view from Google maps.

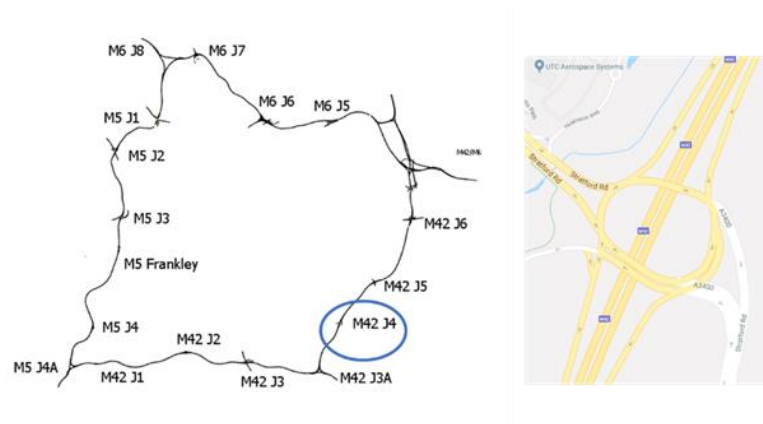


Figure 7-12: Location in the AIMSUN model vs Google Maps

This roundabout has more complicated geometrical characteristics, thus it contains more unknown variables than the M42 J5. Instead of two entrances and two exits from the arterial road, the M42 J4 contains three entrances and three exits. As a result, it has a total of 15 missing variables that correspond to the entrances and exits from the A34, and the turnings inside the roundabout. However, the turnings at the intersections of this roundabout have a more complicated distribution than those in the M42 J5. In Figure 7-13 it can be appreciated the turning distribution in an intersection on each roundabout. While in the M42 J5 the turn proportion has only two possible outcomes, in the M42 J4 there is incoming flow from two sections which is distributed amongst two turns. As a result, it was necessary to divide the incoming flow from these sections in two additional variables. This resulted in four additional constraints and variables at each intersection of the roundabout (See Figure 7-14). Because the association between the new variables constituted the flow of the two exits of the intersection. For instance, the sum of the right turn of C3 (X_{22}) and the right turn of X7 (X_{20}), represents the flow of the exit X_8 (See Figure 7-14).

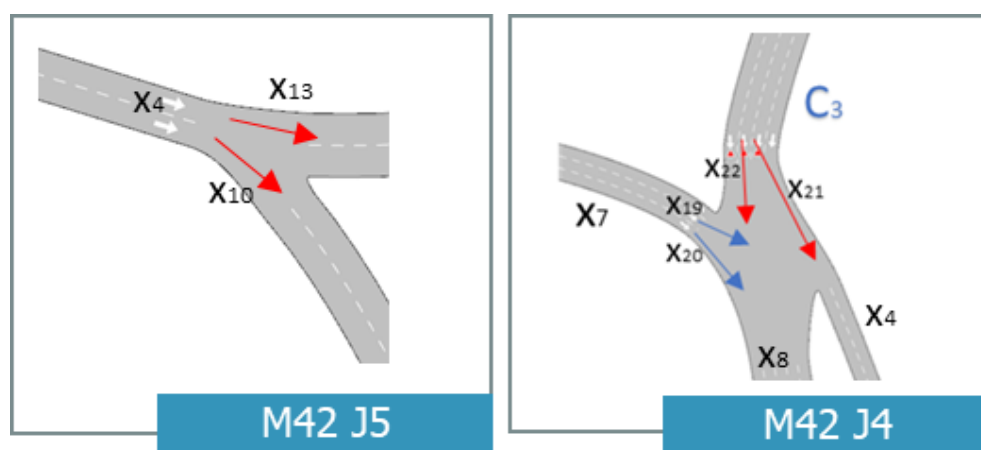


Figure 7-13: Comparison of variable distribution inside intersections on the M42 J5 and M42 J4

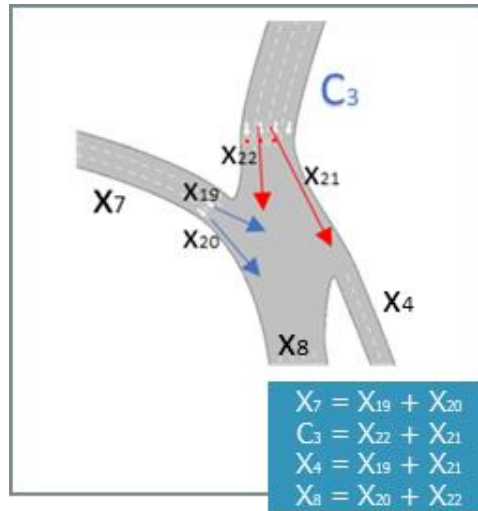


Figure 7-14: Turn proportion on intersection within the M42 J4

In the end, when all the possible turnings were taken into consideration, the optimisation problem had a total of 30 variables and 20 constraints. Figure 7-15 illustrates the variables inside the roundabout.

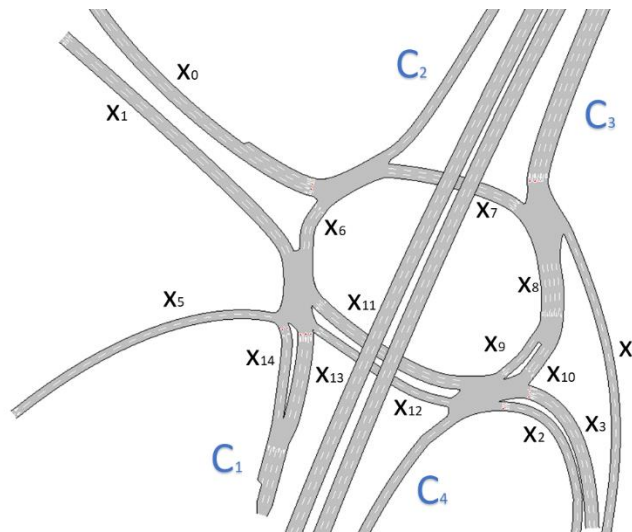


Figure 7-15: Variable distribution on the M42 J4

Similar to the M42 J5, the objective function, uses the flow from the exits (C_2 and C_4) as performance measures. Traffic data from the entrances (C_1 and C_3) were used as input for the equality constraints, while C_2 and C_4 were assumed as two unknown variables (X_{31} and X_{32}). The optimisation problem can be summarised as follows:

$$\text{Minimise } (C_2, C_4) = (C_{2i} - \hat{C}_2)^2 + (C_{4i} - \hat{C}_4)^2$$

where

\hat{C}_2, \hat{C}_4 = actual value of C_2 and C_4

C_{2i}, C_{4i} = predicted value of C_2 and C_4

i = number of iteration

such that

$$X_0 - X_{15} - X_{16} = 0$$

$$X_8 - X_{20} - X_{22} = 0$$

$$C_1 - X_{14} - X_{13} = 0$$

$$X_6 - X_{17} - X_{18} = 0$$

$$X_8 - X_9 - X_{10} = 0$$

$$X_5 - X_{14} - X_{12} = 0$$

$$X_{31} - X_{15} - X_{17} = 0$$

$$X_{11} - X_9 - X_3 = 0$$

$$X_{13} - X_{27} - X_{28} = 0$$

$$X_7 - X_{16} - X_{18} = 0$$

$$X_{10} - X_{23} - X_{24} = 0$$

$$X_{11} - X_{29} - X_{30} = 0$$

$$X_7 - X_{19} - X_{20} = 0$$

$$X_2 - X_{25} - X_{26} = 0$$

$$X_1 - X_{27} - X_{29} = 0$$

$$C_3 - X_{21} - X_{22} = 0$$

$$X_{32} - X_{23} - X_{25} = 0$$

$$X_6 - X_{28} - X_{30} = 0$$

$$X_4 - X_{19} - X_{21} = 0$$

$$X_{12} - X_{24} - X_{26} = 0$$

Traffic data was obtained from the sensors corresponding to C_1 , C_2 , C_3 and C_4 on the 25th of January 2017, from 10am to 11am. Similar to the previous optimisation problem, the initial values of the variables were random numbers greater or equal to 50. The input flows for the traffic state are C_1 , C_3 , X_0 , X_2 and X_3 , while the other values were converted into turning percentages. Optimisation terminated successfully after 44 iterations in 2.3 minutes, this is when the value of the MAPE, for the simulated values of C_2 and C_4 was less than 1%. Table 7-5 and Table 7-6 present the initial values, and

results from the optimisation routine. The error for C_2 was 0.30%, while for C_4 was only 0.56%. Table 7-7 contains the MAPE for each variable.

Table 7-5: Initial values for optimisation

C₁	428veh/h
C₂	329 veh/h
C₃	581 veh/h
C₄	179 veh/h

Table 7-6: Results from optimisation routine

Optimisation terminated successfully	
Success	True
Time	140.6s (2.3min)
Iterations	44
X₁	280.10 veh/h
X₂	389.86 veh/h
X₃	344.68 veh/h
X₄	268.49 veh/h
X₅	543.91 veh/h
X₆	460.49 veh/h
X₇	412.36 veh/h
X₈	363.46 veh/h
X₉	400.55 veh/h
X₁₀	165.59 veh/h
X₁₁	234.96 veh/h
X₁₂	434.08 veh/h
X₁₃	400.64 veh/h
X₁₄	368.14 veh/h
X₁₅	59.85 veh/h
X₁₆	51.73 veh/h
X₁₇	228.36 veh/h
X₁₈	277.26 veh/h
X₁₉	135.09 veh/h
X₂₀	235.12 veh/h
X₂₁	128.34 veh/h
X₂₂	308.78 veh/h
X₂₃	272.21 veh/h
X₂₄	59.01 veh/h
X₂₅	175.95 veh/h
X₂₆	119.98 veh/h
X₂₇	224.69 veh/h
X₂₈	124.96 veh/h
X₂₉	243.17 veh/h
X₃₀	264.90 veh/h
X₃₁	169.18 veh/h

Table 7-7: MAPE of simulated results

Variable	Actual value	Simulated Value	MAPE
C₂	329	330	0.30%
C₄	179	180	0.56%

7.5 Discussion

One of the biggest challenges associated to traditional sensor technologies is the absence of information due to sensor malfunction or unavailability of hardware in specific parts of the network. This can pose a challenge, not only for traffic forecasting and incident detection techniques, but also for the development of traffic simulation models. As seen in the literature (section 2.4), many researchers have developed different methodologies for the imputation of missing traffic data. However, the methods described in the literature work well for the prediction of random errors, not for when data is missing for extended period of times. Moreover, most of these methods have been employed in straight sections of freeways and their suitability for more complex networks have not been validated (Shang *et al.*, 2018).

The model presented in this chapter uses traffic flow data from Highways England to set up the simulator component. While there are many sensors for the motorways in the model, there is no information for the arterial roads that these motorways are connected to. Due to the way that traffic is imported into the model, and the logical movement of flow in roundabouts, an optimisation procedure for determining missing roundabouts and urban roads flow distribution has been proposed. The methodology uses as constraints the relationship between the known and unknown variables, and

the turn proportions in the intersections. A constrained optimisation algorithm uses these limitations to minimise a function and find the optimal solution.

The proposed procedure was implemented in two roundabouts in the model: M42 J5 and M42 J4. These two cases were selected because of their different geometrical characteristic; thus it would be possible to assess the methodology in different scenarios. The first case, M42 J5, had 12 variables and 8 constraints. The complete procedure took 22 seconds to find the best solution to the optimisation problem. On the other hand, the M42 J4, lasted more than 2 minutes to find the most suitable result. This is due to a more complicated optimisation problem that had 31 variables and 20 constraints. But then, when running the optimisation problem again, the procedure lasted for 7 minutes. Because there are many possible solutions for the optimisation problem, it is difficult to find one that would satisfy the simulated results with an error lower than 1%. In order to reduce the processing time, the optimisation was implemented with a higher error tolerance (2% instead of 1%), and it took less than 50 seconds to converge.

Results from the implementation indicate the potential of the proposed procedure to find missing traffic flow values. Existing imputation methodologies have been investigated using methods based on time series, interpolation and statistical techniques (Li, Li and Li, 2014). However, due to the particularity of the problem, it can pose a challenge to apply these methods to impute the distribution of roundabouts and arterial roads in the model. Mainly, because it is not possible to guarantee a correlation between neighbouring detectors. Moreover, there is no historical traffic flow data from the arterial roads or the turn proportions inside the roundabout. A similar

study in the literature, Muralidharan and Horowitz (2009) experienced the same issues when trying to impute missing ramp flow data for a macroscopic simulation model. There was no information available about the flow and split ration of the on-ramp and off-ramp. Similar to this research, with the purpose of completing the simulation model, they developed an iterative process that minimised the error between the model calculated flows and real measurements. Their imputation methodology was implemented in a 26-mile straight section of highway, with maximum flow errors of 3.58%. While these are promising results, their proposed methodology was validated on a straight section of highway, while the imputation methodology proposed in this research can be applied in more complex networks. The imputation methodology proposed in this chapter can aid not only in the development of traffic simulation models, but also to impute missing traffic information from complex networks that can further support TMS such as traffic prediction and incident detection techniques. In this stage, the roundabouts were tested isolated with the purpose of evaluating the methodology in a secluded environment. For the real-time implementation of the methodology, both roundabouts will be treated as one optimisation problem.

7.6 Summary

This chapter outlines the different steps undertaken to prepare the simulator component, as well as the proposed methodology for the imputation of missing arterial road data. The study area selected for the simulator component was the motorways surrounding the city of Birmingham. Traffic sensors from Highways England were mapped to the different sections of the model using different methodologies developed in the Python programming language. In addition, other tasks automated with Python

scripts included automatic simulation and importing the traffic data into AIMSUN. While a section of motorway could have up to 16 sensors, arterial roads in the model had no traffic information linked to them. To tackle this, an optimisation procedure to impute the missing traffic flow data has been developed. The methodology uses constrained optimisation to minimise a function with respect of some limitations. Results from the implementation suggest the potential of the proposed procedure to impute missing data in complex networks. With the purpose of highlighting the relevance of the simulator component for TMS, the next chapter presents the real-time implementation of the proposed methodology.

Chapter 8: Data imputation for real-time TMS applications

8.1 Introduction

Chapter 7 presented an optimisation-based traffic data imputation methodology for missing traffic flow data in roundabouts. With the purpose of assessing the performance of the proposed methodology, two different roundabouts were simulated in isolation. This chapter is concerned with evaluating the usability of the methodology in a more complex simulation model composed of numerous intersections, and on a real-time basis. The chapter starts by outlining the constrained optimisation problem, along with a description of the simulated model, in section 8.2. This section introduces the real-time processing pipeline used to implement the methodology on a 24-hour period. Results from the optimisation routine and recommendations for future improvements are also presented. Finally, the relevance of the proposed simulator component is further confirmed by modelling the impact of incidents in the modelled network in section 8.3.

8.2 Real-time imputation of missing flow in complex networks

The section of the M42 that contains the junctions 3A, 4 and 5; was selected for the validation of the proposed imputation methodology (Figure 8-1). The selected network has a total of 46km and 30 intersections. The purpose of this experiment is to evaluate the performance of the methodology in a complex model comprised of more than one intersection, instead of each being in isolation.

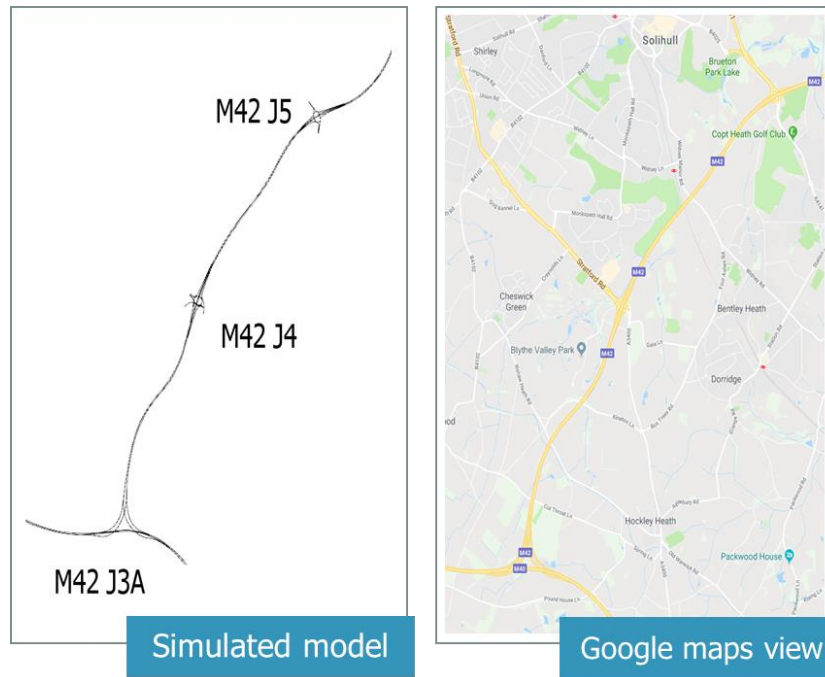


Figure 8-1: Simulated network vs Google Maps view

The constrained optimisation problem has been rewritten in terms of the constraint equations presented for each roundabout in Chapter 6. The M42 J4 had a total of 30 variables and 20 constraints, and the M42 J5 had 8 constraints and 12 variables. As a result, the revised optimisation problem has a total of 28 constraints and 42 unknown variables. Figure 8-2 illustrates the variable distribution in both roundabouts. C_2 , C_4 , C_5 and C_7 represent the exits, while C_1 , C_3 , C_6 and C_8 are the entrances of both roundabouts. The objective function has been built using exits from both roundabouts as unknown variables (X_{31} , X_{32} , X_{45} and X_{46}), and their corresponding entrances as input for the equality constraints.

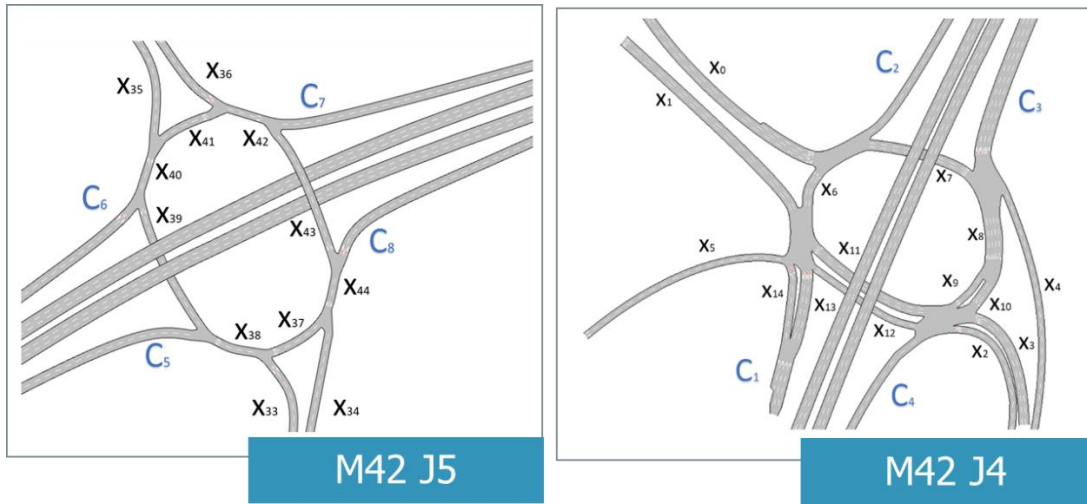


Figure 8-2: Variable distribution in both roundabouts

The optimisation problem can be summarised as follows:

$$\text{Minimise } (C_2, C_4, C_5, C_7) = (C_{2i} - \hat{C}_2)^2 + (C_{4i} - \hat{C}_4)^2 + (C_{5i} - \hat{C}_5)^2 + (C_{7i} - \hat{C}_7)^2$$

where

$\hat{C}_2, \hat{C}_4, \hat{C}_5, \hat{C}_7$ = actual value of C_2, C_4, C_5 and C_7

$C_{2i}, C_{4i}, C_{5i}, C_{7i}$ = predicted value of C_2, C_4, C_5 and C_7

i = number of iteration

such that

$$\begin{array}{llll} X_0 - X_{15} - X_{16} = 0 & X_8 - X_{20} - X_{22} = 0 & C_1 - X_{14} - X_{13} = 0 & X_{39} + X_{45} - X_{38} = 0 \\ X_6 - X_{17} - X_{18} = 0 & X_8 - X_9 - X_{10} = 0 & X_5 - X_{14} - X_{12} = 0 & X_{40} - C_6 - X_{39} = 0 \\ X_{31} - X_{15} - X_{17} = 0 & X_{11} - X_9 - X_3 = 0 & X_{13} - X_{27} - X_{28} = 0 & X_{40} - X_{35} - X_{41} = 0 \\ X_7 - X_{16} - X_{18} = 0 & X_{10} - X_{23} - X_{24} = 0 & X_{11} - X_{29} - X_{30} = 0 & X_{41} + X_{36} - X_{42} = 0 \\ X_7 - X_{19} - X_{20} = 0 & X_2 - X_{25} - X_{26} = 0 & X_1 - X_{27} - X_{29} = 0 & X_{42} - X_{46} - X_{43} = 0 \\ C_3 - X_{21} - X_{22} = 0 & X_{32} - X_{23} - X_{25} = 0 & X_6 - X_{28} - X_{30} = 0 & X_{43} + C_8 - X_{44} = 0 \\ X_4 - X_{19} - X_{21} = 0 & X_{12} - X_{24} - X_{26} = 0 & X_{33} + X_{37} - X_{38} = 0 & X_{44} - X_{37} - X_{34} = 0 \end{array}$$

While the missing values and constraints remain the same, the optimisation results of one roundabout will affect flows on others. This is because the exit of the M42 J5 is connected to the values of the entrances of the M42 J4, and vice versa. Therefore, it would pose a challenge to find a solution that satisfies the condition of an error lower than 1% in both roundabouts. In order to find the optimum error tolerance that would result in low processing time, the routine was tested several times using different error thresholds. It was observed that when the error of the exits that connected the roundabouts was low, the other two exits had even lower values. For instance, when looking at Figure 8-3, it is possible to appreciate that the value of C_5 and C_2 , directly influence the input of both roundabouts. Because eventually C_5 would influence the value of the entrance C_3 , and C_2 the value of the entrance C_6 . This is, that the most realistic the turning proportions are in both optimisation problems, the main exits of both roundabouts (C_4 and C_7) will have a lower error. On the contrary, when there is a high error in any of the two optimisation problems, this will drag the error to the following roundabout, and consequently to the exits of the system. It was also noticed that using an error condition of less than 5%, resulted in very high processing times. However, an error tolerance of less than 10% had low computing times and tended to include values of less than 5% in at least two out of four exits. As a result, the validation task was revised, and the error allowance was changed to 10%. Finally, it was noticed that the time it took to simulate the model was significantly higher than during the isolated experiments. For this reason, instead of one-hour intervals, the procedure was employed every 15 minutes.

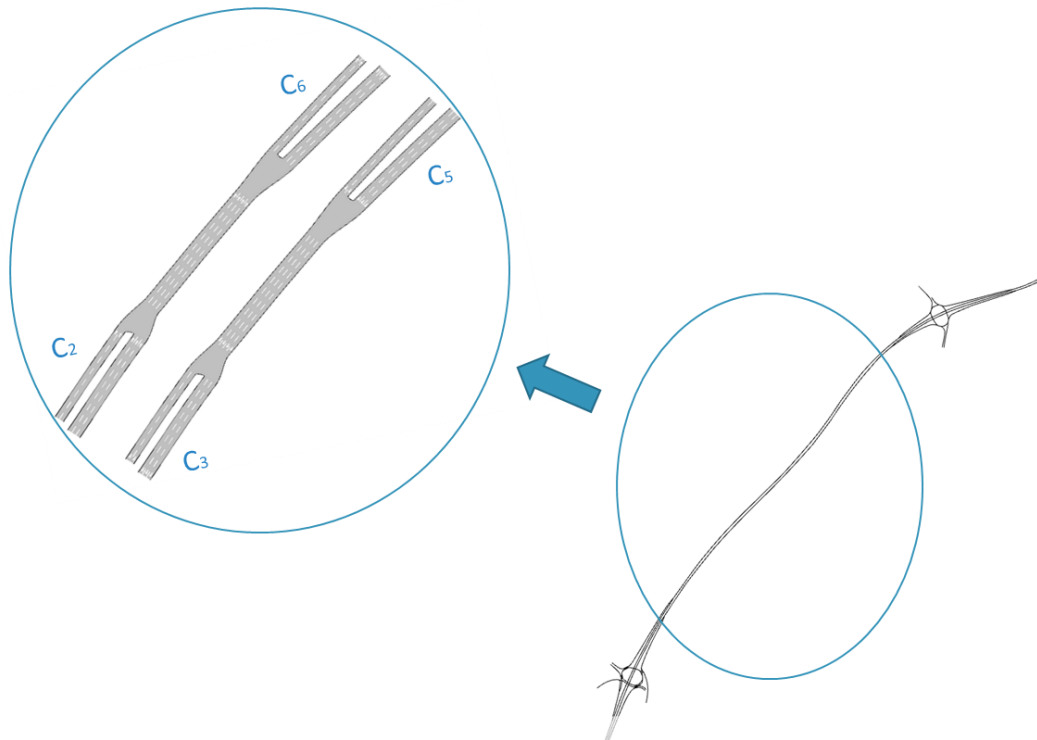


Figure 8-3: Relationship between inputs and exits

With the purpose of validating the imputation methodology on real-time, an implementation pipeline was employed. The pipeline was built based on the scripts presented in Chapter 6 (section 7.3.2), with regards of automatically importing traffic data and running simulations in AIMSUN. The pipeline takes into consideration the possibility of the optimisation procedure not finding a solution for the problem, as well as the veracity of the traffic data obtained from the traffic sensors. The latter can be because the error distribution between associated sections is too high for the procedure to find the optimal values. Every 15 minutes, the following steps are followed:

- Step 1.* Obtain traffic information from the last 15-minute interval
- Step 2.* Calculate the error between flows and distribute it equally, adding or subtracting were necessary.
- Step 3.* Import the corresponding flow values into the optimisation problem

- Step 4.* Optimisation is performed. If optimisation is not successful, the process is terminated.
- Step 5.* Create a traffic state and imports entrances/turn percentages into the AIMSUN model.
- Step 6.* Run the simulation.
- Step 7.* Calculate MAPE estimated data and actual data.
- Step 8.* Judge whether the stopping criteria ($MAPE < 10\%$) is reached, otherwise return to step 4.

The aforementioned process was employed for a period of 24-hours, on Wednesday 25th of September 2018. The procedure was able to impute missing traffic data from 10 hours, between 06:00 to 22:00. Table 8-1 presents an extract of the results from the validation of the imputation methodology, in 15-minute intervals. As highlighted before, while the error tolerance was set to 10%, most of the values tended to be below 5%. In fact, in some cases the MAPE was less than 1%, and even 0% in at least two of the variables. This can be justified by the fact that these sections are connected to each other, and a low percentage error in C_4 and C_7 , would generate a low value in the two main exits of the system (C_2 and C_5). On the other hand, it can be appreciated that the number of iterations and processing time was higher in some instances, with respect to others. In some intervals, optimisation lasted for almost 14 minutes (750s) and iterated more than 40 times. This could be due to a more difficult optimisation procedure, where it was particularly challenging to find values that would satisfy the validation stage. However, for many time intervals it was the opposite: the optimisation problem only needed one iteration to converge, in less than 12 seconds.

Table 8-1: Results from the validation of the imputation methodology

	Time interval		MAPE				Iteration	Time
			C ₂	C ₄	C ₅	C ₇		
06	06:15	06:30	2.86%	8.89%	8.57%	8.45%	9	109.88
	06:45	07:00	4.46%	7.59%	8.50%	8.26%	7	109.4
07	07:00	07:15	3.57%	3.53%	9.66%	0.84%	1	16.38
	07:15	07:30	9.63%	2.86%	6.28%	9.05%	2	39.30
	07:30	07:45	8.63%	9.09%	9.84%	4.49%	5	91.99
	07:45	08:00	9.30%	1.61%	6.13%	7.81%	1	18.75
08	08:00	08:15	5.86%	4.76%	3.65%	9.52%	9	165.6
	08:15	08:30	9.31%	8.25%	8.88%	9.64%	3	56.02
	08:30	08:45	5.71%	2.74%	1.26%	5.56%	42	744.8
	08:45	09:00	5.94%	1.23%	2.42%	1.27%	1	17.68
09	09:00	09:15	2.22%	1.85%	0.00%	1.82%	9	145.6
	09:15	09:30	1.09%	4.44%	8.72%	9.86%	2	32.84
	09:30	09:45	2.15%	1.28%	6.90%	5.69%	1	16.14
	09:45	09:00	0.61%	2.38%	6.25%	4.00%	14	215.8
10	10:00	10:15	3.39%	8.51%	7.02%	0.00%	11	146.9
	10:15	10:30	2.21%	6.25%	9.65%	9.20%	7	95.28
	10:30	10:45	5.46%	0.00%	7.74%	0.45%	3	42.06
	10:45	11:00	6.35%	2.22%	2.04%	6.52%	7	96.13
15	15:00	15:15	0.45%	3.70%	8.40%	5.77%	3	53.03
	15:15	15:30	9.38%	8.16%	4.13%	1.86%	19	355.2
	15:30	15:45	4.37%	0.98%	2.07%	6.67%	16	290.3
	15:45	15:00	1.05%	2.63%	1.98%	9.26%	37	764.5
16	16:00	16:15	3.31%	7.69%	4.80%	1.43%	8	175.4
	16:15	16:30	4.32%	4.49%	6.25%	9.45%	45	172
	16:30	16:45	5.16%	7.14%	6.88%	8.78%	2	45.5
17	17:15	17:30	8.74%	7.69%	5.69%	9.76%	9	163.30
	17:30	17:45	6.92%	1.74%	9.52%	8.75%	1	18.20
	17:45	18:00	8.53%	1.01%	3.94%	6.97%	13	235.53
18	18:00	18:15	1.75%	4.72%	2.48%	2.70%	13	210.4
	18:15	18:30	0.99%	6.49%	3.25%	7.21%	2	31.79
	18:30	18:45	6.90%	9.80%	9.63%	5.88%	16	267.1
	18:45	19:00	3.35%	1.64%	0.66%	2.38%	1	15.5
19	19:00	19:15	0.76%	8.11%	0.00%	5.56%	1	12.05
	19:15	19:30	5.08%	2.33%	1.74%	6.67%	2	22.57
	19:30	19:45	4.12%	3.23%	9.28%	4.55%	12	128.1
	19:45	20:00	4.30%	7.41%	7.50%	8.11%	7	68.80

There were some time intervals where the procedure terminated without applying the imputation methodology. This can be linked to the optimisation procedure not being able to find a solution for the problem. However, it was noticed that periods with low flow (e.g. 00:00 to 04:00), were the periods where the methodology failed to optimise the values. It was during these time intervals that the traffic data received from the

sensors had very low values that created a difficult optimisation problem. For instance, imagine the intersection of C_3 (Figure 8-3), when manually assessing the data in the intervals where the optimisation failed to find a solution, it was noticed that the flow distribution of C_3 was very low with respect of those in normal flow distribution. Table 8-2 compares the flow distribution of C_3 in periods with low flow and in those with normal flow distribution. It can be noticed that the difference between both is very high, which is the reason why the optimisation could not find a solution to the problem. In contrast, the optimisation had very good performance in finding the missing values in periods with normal distribution and peak time of the days. The complete table of results corresponding to the 17-hour interval, can be found in Appendix 8-A.

Table 8-2: Comparison of periods with low flow and normal flow distribution

Periods with low flow			Normal flow distribution		
From	To	Flow	From	To	Flow
00:00	00:15	3	11:00	11:15	820
00:15	00:30	7	11:15	11:30	728
00:30	00:45	6	11:30	11:45	728
00:45	01:00	6	11:45	12:00	704
01:00	01:15	1	12:00	12:15	745
01:15	01:30	1	12:15	12:30	630

The proposed methodology has the advantage of being easily adapted to different specifications. For instance, conditions such as time interval and error tolerance, can be changed depending of the characteristics of the transport network. The main purpose of the validation experiment was to test the applicability of the proposed procedure on a real-time basis. These results, in terms of processing times and MAPE, confirm the potential of the proposed system to impute missing traffic flow information to support existing traffic forecasting and incident detection systems. This

methodology can also aid in the development of simulation models of complex networks with missing traffic data for extended period of times.

8.3 Modelling the impact of incidents

The proposed simulation component aims to provide simulation of accidents and their related impact on capacity, as well as the evaluation of different alternatives to diminish congestion. Many authors in the literature have already evaluated the impact of AIMSUN in simulating incidents in the transport network (Barceló Bugeda *et al.*, 2002; Barceló *et al.*, 2005; Hadi, Sinha and Wang, 2007). For instance, Barceló Bugeda *et al.* (2002) evaluated the effectiveness of the incident management capabilities of AIMSUN by simulating the impact of traffic incidents and roadworks in urban networks. Hadi, Sinha and Wang (2007) assessed the reduction of capacities in a freeway due to incidents by simulating the network in AIMSUN. These findings further confirm the applicability of the methodology for the creation of realistic simulation models that can serve as a tool for simulating the impact of accidents.

The complete traffic simulation model presented in Chapter 7 was used to generate incidents and evaluate their impact on the network performance. This is with the purpose of highlighting the relevance of the simulator component proposed in this research. Several incidents were modelled in the sections of the M6 and M42 displayed in Figure 8-4. The sections selected for the incidents were the M6 southbound within the M6 Junction 8 and the M42 southbound between junction 3 and junction 3A. Incidents were modelled at 06:30am, and the different scenarios included a base case with no incident, while the remaining were produced by changing duration and severity (e.g. one lane, dual lane) on both incidents.

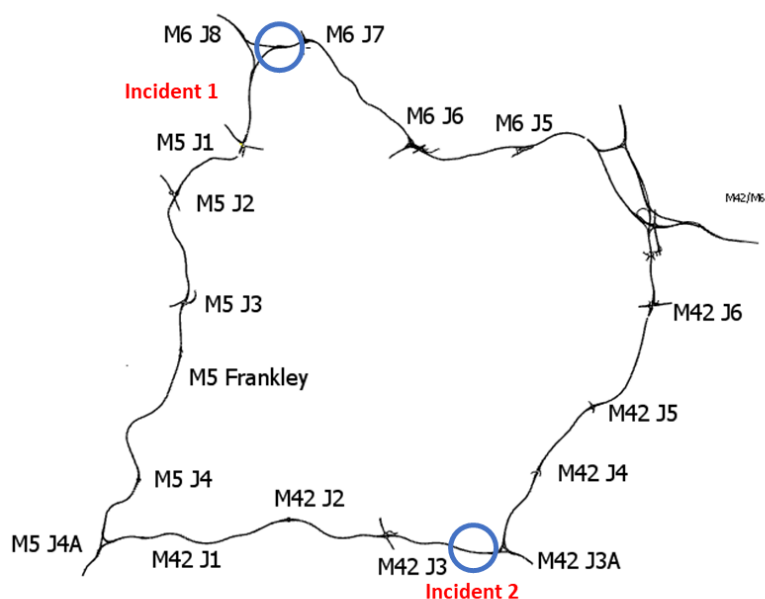


Figure 8-4: Incidents location on the modelled network

In order to compare the impact of the incidents, several key performance indicators were exported from the simulated results. These indicators were evaluated for the sections where the incident happened, and for those relevant sections affected by these incidents. Table 8-3 and Table 8-4 show the percentage of reduction in speed and travel time in the different scenarios, with respect of the base case. It can be appreciated the benefits of reducing the duration of an incident by 15 minutes, as it provides an increase to the average speed of at least 3% and a decrease of travel time of more than 40%. While some benefits in the one lane scenario appear to be small, Dia (2003) estimated that by every 1% reduction in travel time, the corresponding financial cost can be substantial. Moreover, Dia *et al.* (2006), determined the environmental benefits of reducing an incident duration by 15 minutes, showing a 11.2% reduction in fuel cost, 12.7% reduction in CO and 11.2 reduction in CO₂. This confirms the potential benefits of evaluating different incident management strategies

using a simulation model, thus the incident duration can be reduced and its consequent impact on network performance.

Table 8-3: M6 performance indicators (percentage)

Scenario	Severity	Percentage	
		Speed	Travel time
15-minute incident	One lane	-2.6%	2.23%
	Dual lane	-19%	38%
30-minute incident	One lane	-5.8%	6.5%
	Dual Lane	-30%	80%

Table 8-4: M40 performance indicators (percentage)

Scenario	Severity	Percentage	
		Speed	Travel time
15-minute incident	One lane	-13%	20.4%
	Dual lane	-22.3%	41.2%
30-minute incident	One lane	-15.6%	23.8%
	Dual Lane	-35.8%	66.4%

As discussed in Chapter 4, the proposed smart traffic management framework (Figure 4-5) integrates social media data and real-time simulations of the transport network. After finding the missing traffic data during the optimisation-based imputation stage (presented in section 7.4), traffic incidents identified by the incident detection layer can be modelled in the simulator component to explore different scenarios and select the ideal action to mitigate its impact. Results from the real-time implementation of both components (section 6.2 and section 8.2) and the incident scenarios simulated in this section, validate the potential of the smart traffic management framework to support the operation of existing TMS on real-time. Not only for traffic incident detection, but for improving the prediction task through the implementation of the optimisation-based imputation methodology.

8.4 Summary

This chapter was concerned with evaluating the imputation methodology on a real-time application. To this end, a real time processing pipeline has been developed that used the previously presented tasks for automatically importing traffic data, imputation of missing traffic flows and running traffic simulations in AIMSUN. The methodology distributed the error of the traffic data obtained from the sensors, and in cases where the discrepancy was too high, it terminated the process. The real-time processing pipeline was tested in a 24-hour period, every 15 minutes. From this time period, the routine was able to impute missing flow data for a 10-hour period. This was due to a high discrepancy in the traffic data received from the sensors, specifically in times with very low traffic flow or at peak times of the day. Results from this implementation suggest that in most of the cases, the methodology was able to quickly find a solution with less than 10% of error. In few instances, the methodology faced a difficult optimisation problem, where it took more than 14 minutes to find a solution that satisfied the validation task. While the error threshold for the validation task was 10%, most of the cases it converged with an error lower than 5%. This is due to the relation that the different variables have with one another, where the flow distribution of one roundabout influences the values of the other one. These results validate the potential of the methodology to support the operation of TMS. The aim of the proposed simulator component is to integrate real-time simulations of the transport network, to evaluate the impact of accidents and the relevant strategies to ease congestion. To this end, a set of incident scenarios were modelled in the complete network, with the purpose of evaluating the performance of the network with different incident durations.

Chapter 9: Conclusions and contributions to knowledge

9.1 Introduction

This chapter summarises the findings and research contributions of this thesis. The chapter starts with section 8.2 summarising the conclusions of the research. This section highlights the different techniques proposed and implemented to achieve the aim and objectives of the research. Sections 9.3 and 9.4 summarise the main and extended contribution to knowledge of this research. Lastly, section 9.5 outlines the recommendations for future research based on the findings presented on this thesis.

9.2 Conclusions

This thesis presented a smart traffic management framework that integrates social media data and real-time traffic simulation of the transportation network into existing TMS. The framework proposed the integration of two interconnected components: social media and simulator component. An investigation into the suitability of the proposed framework to improve the operation of incident detection and prediction algorithms within TMS, has been carried out through a set of experiments throughout this research.

Chapter 4 introduced the proposed smart traffic management framework that integrates heterogeneous data from systems, devices and traffic simulation tools. The framework incorporated a social media and simulator component with the purpose of supporting AID and the traffic prediction task in existing TMS. Mainly, for providing support on issues related to missing traffic data due to sensor malfunction and poor

performance of existing algorithmic techniques. The social media component seeks to identify incidents and issues in the transport network using real-time Twitter data. To this end, section 4.3.1 outlined the social media processing systems composed of NLP techniques, location extraction and sentiment and stress analysis. In contrast, the simulator component presented in section 4.3.2 has two main tasks: imputation of missing flow data and simulation of incidents on the network. The simulator component proposes an optimisation-based imputation methodology to solve the missing traffic data problem. It also supports real-time simulations of the network for identifying optimal solutions to existing issues in the transport network.

The social media component has been outlined and tested in Chapter 5. The purpose of this was to find the most suitable techniques for the real-time implementation of the social media processing system. The results obtained in terms of classification accuracy, geolocation techniques and sentiment and stress analysis; demonstrated the potential of the component to extract useful traffic information from Twitter streams. In terms of classification, from the five different algorithms compared, a RRC showed the highest accuracy in both training and test datasets (section 5.4.2). This was an interesting finding as most of the studies from the literature have found SVM to be the most accurate text classification algorithm. The accuracy of the RRC outperformed studies in the literature with similar datasets and implementation methodologies. In terms of geolocation, few researchers have extracted the location of the traffic incident from the tweets, and results from this testing stage outperformed those obtained in the existing literature (section 5.5). One of main innovations of this thesis is the incorporation of sentiment and stress analysis into twitter-based incident detection

systems. This is with the purpose of analysing the potential of user generated content for opinion mining in transportation. Results from stress and relaxation analysis demonstrated that over 60% of the tweets evidenced stress when describing the traffic event taking place.

The aforementioned results were further validated in the real-time implementation of the social media component presented in Chapter 6. The overall accuracy of the RRC classifier during the real-time operation was 91.83%. Because this accuracy only reflected the high precision in detecting non-traffic related tweets due to having an imbalanced dataset, the Kappa score was proposed to further validate the findings (in section 6.2). Results from the kappa score demonstrated that the classifier had a substantial accuracy, with an almost perfect agreement. When analysing the emotions within the tweets, section 6.2 concluded that TensiStrength was not able to capture very well the emotions in those tweets that expressed any form of sarcasm. Sarcasm usually involves using words that while they sound relaxing, they express quite the opposite. As a result, many tweets in the dataset were assigned low stress values, as the negative emotions were commonly expressed through sarcasm. The location mentions identified through the custom NER were associated to the stress identified through the strength and relaxation analysis. When manually assessing the dataset, it was noticed that the increase in stress in some motorways was related to actual traffic incidents and complaints about roadworks being carried out in different sections of the motorway. These findings support the idea that the emotions detected on traffic tweets can be linked to specific incidents or complaints taking place in specific sections of the transport network. This information can be further used to identify the factors causing

non-recurrent congestions and to understand the users' perception of specific areas of the transportation network.

Section 6.3 presented the validation of the methodology by confirming the events identified in Twitter with local news. The major events identified on Twitter were mapped to news reports with a minimum delay of 15 minutes, further confirming findings in the literature about Twitter not being suitable for automatic incident detection (Schulz, Ristoski and Paulheim, 2013; Zhang *et al.*, 2018). Nevertheless, one of the most observed characteristics of a sample of tweets in the dataset was the sentiment affiliated with the desire to help others. This sample was dedicated to highlight flaws in the network to official traffic accounts, such as traffic lights not working, matrix signs not updated, or unexplained traffic events. These types of tweets can support TMS in the detection of issues in areas of the transport network where there is less coverage from sensory ITS devices. Moreover, this data can potentially help in identifying specific traffic events happening in urban roads, as some of the traffic events described in these tweets do not even reach traffic news. These findings suggest that the purpose of the social media component within the proposed framework, leans towards the identification of issues and users' perceptions for the road network, rather than an incident detection tool itself.

Chapter 7 outlined the simulator component, and the network created for the implementation of the proposed methodology. Section 7.3 presented the different sub-processes created with the purpose of importing real-time data and automatically running simulations. While a section of motorway could have up to 16 sensors, arterial roads in the model had no traffic information linked to them. Without this information,

it would be impossible to create a realistic simulation model. In order to overcome this issue, section 7.4 introduced an optimisation-based traffic data imputation methodology that uses constrained optimisation to minimise a function with respect of some limitations. Results from the testing stage in section 7.4.3 demonstrated the potential of the proposed procedure to impute missing data in complex networks.

With the purpose of highlighting the relevance of the simulator component for TMS, Chapter 8 implemented the simulator component in real-time on a 24-hour timeframe. To achieve this, section 8.2 presented a real-time processing pipeline that implements the previously presented tasks or automatically importing traffic data, imputation of missing traffic flows and running traffic simulations in AIMSUN. From the 24-hour period, the methodology was able to impute missing traffic data for a 10-hour period. This was because in off peak times of the day with very low traffic flow (early morning), there was not enough traffic data for the imputation methodology to be triggered. In most of the cases, the methodology was able to find a solution with less than 5% error, with the most difficult cases lasting up to 14 minutes. Reflecting on the evidence from the literature that suggest missing traffic data one of the main issues faced by TMS (as discussed in Chapter 2), these findings validate the potential of the optimisation-based methodology to support the real-time operation of TMS. Lastly, section 8.3 evaluated the usefulness of the proposed simulator component for modelling traffic incidents. To this end, a set of incident scenarios is modelled in the complete network, with the purpose of evaluating the performance of the network with different incident durations. Results showed the potential benefits of evaluating different strategies to

reduce the incident duration and its consequent impact to the performance of the transportation network.

The smart traffic management framework presented and implemented in this thesis has the potential to improve the performance of incident detection and traffic prediction tasks in existing TMS. The incorporation of real-time traffic information from Twitter streams, stress detection analysis for opinion mining and an optimisation-based approach for imputing missing traffic data; situates the proposed framework as a key step towards smart transportation.

9.3 Main original contributions

The contribution to knowledge of the proposed smart traffic management framework falls under the improvement quadrant of the DSR knowledge contribution, as it represents an improvement on existing techniques with a low maturity level.

A social media component for transportation services

(Objective 3/Objective 5)

This research proposed a social media component to improved existing methodologies for processing information from social media sensors. It consisted of a processing pipeline using state of the art techniques for processing, classifying, geolocating and extracting sentiment from short texts. A novel classification algorithm trained using public tweets was devised, extending current research which has been constraint with automated and well-structured tweets. The proposed component also advances beyond the existing state of the art in terms of incident geolocation, by introducing a custom NER model to extract location from short texts and a knowledge base to link

this location to a set of coordinates. Finally, this research is amongst the first to analyse the emotions within traffic related tweets for policy making in transportation. The different tasks of the proposed component were tested, in order to find the most appropriate techniques for each sub-component. Five machine learning algorithms were compared, and a classifier based on a RRC showed the highest accuracy in both the training and test dataset, outperforming similar studies in the literature (Salas, Georgakis and Petalas, 2017; Salas Jones *et al.*, 2018). The custom NER model was able to successfully identify location mentions within the tweets with a 95.5% of accuracy. From these locations, the knowledge base was capable of linking the location mention with a 97.12% accuracy. Results from stress analysis showed that more than 60% of the tweets demonstrated a significant amount of stress. Finally, this research provided fresh insights into the role of the users' for identifying and reporting issues on the transport network, and its potential contribution to opinion mining in the transportation domain.

An imputation methodology for missing flow data

(Objective 4/Objective 5)

This research presented a simulator component, along with an imputation methodology for missing flow data in complex networks. One of the major issues associated with creating a realistic simulation model is missing traffic data. A considerable body of research has proposed different imputation methodologies for missing traffic flow data in motorways. However, due to the more complex characteristics of the study site, and a problem of non-existent traffic data for the arterial roads in the system, it would pose a challenge to implement existing imputation

methodologies in the simulated network. To this end, an optimisation procedure to impute the missing traffic flow data has been proposed. The developed optimisation model poses an advancement in the current state of the art, which has investigated primarily data imputation on straight motorway segments. The methodology uses constrained optimisation to minimise a function with respect of some limitations. Results from the case study implementation suggest the potential of the proposed procedure to impute missing data in complex networks. These findings demonstrate the relevance of the methodology not only for the preparation of large-scale simulation models, but to improve traffic prediction and incident detection tasks.

9.4 Extended original contributions

In addition to the novelty methodologies presented in the previous section, this research made some secondary, but nevertheless original contributions to other aspects of smart traffic management.

A review of existing developments on TMS, as well as the current state-of-the-art into the use of Twitter data for incident detection (Objective 1)

This research started reviewing the most relevant sensor technologies and algorithms employed in existing TMS (Chapter 2), and the potential of Twitter data in the transportation domain (Chapter 3). An in-depth review of existing traffic prediction and automatic incident detection systems was undertaken to provide an understanding of the current issues faced by these technologies. This review revealed that these systems still suffer from high false alarm rates due to missing traffic data and sensor

malfunction. As a result, more than 70% of traffic operators consider existing systems as insufficient. Conclusions from this chapter were also drawn into the implementation of simulation models for traffic management tasks. There is a wide amount of research around using simulation models for assessing the impact of accidents in the transport network and evaluating the different solutions to diminish the impact of these accidents (section 2.3.3). However, similar to prediction and incident detection systems, the credibility of a simulation model depends on the validity of the traffic data and other parameters specified in the model. This led to a review on existing methodologies for the imputation of missing traffic data (section 2.4), where it was concluded that these techniques work well on straight sections of highways, and have not been validated in more complex networks. The last part of the review highlighted the role of transportation within the smart city paradigm. It has been established that a future smart city should take advantage of smart devices, in-vehicle technologies, and social media data to provide a real-time integrated mobility service. Hereupon, authors in the literature have suggested the incorporation of social media and real-time simulations of the network as the solution for the issues faced by existing TMS.

Chapter 3 provided an insight into the existing developments of incident detection using Twitter data. The survey of existing techniques revealed that many authors have developed methodologies for exploiting twitter data for traffic event detection. Nevertheless, rather than being implemented on real-time, these techniques have been tested using historical data. While these authors have obtained promising results in terms of algorithm accuracy, there are still many challenges associated with exploiting Twitter data (in section 3.4). These challenges are related to limits on

Twitter data acquisition, NLP techniques, and geolocating the tweet. As a result, it has been concluded that traffic incident techniques based on Twitter data aim to enhance conventional methodologies, rather than substituting them. Lastly, section 3.3 identified the potential of using Twitter data for opinion mining in transportation. There is a general lack of research in using Twitter data to obtain the users' perception of the transport network and its implementation for policy making.

Smart traffic management framework for transportation services (Objective 2)

The research proposed a smart traffic management framework that exploits data from heterogeneous sources to support existing TMS. The proposed framework is divided in two main components: social media and simulator. The aim of the social media component is the identification of traffic events and issues in the transport network, using real-time twitter data. This component is composed of a processing pipeline that uses NLP techniques, location extraction, and sentiment and stress analysis. Meanwhile, the simulator component has been presented with two main purposes: the imputation of missing traffic flow data and the simulation of accidents. To this end, an imputation methodology has been proposed that can support not only in the creation of simulation networks with missing flow data, but it can also aid existing traffic prediction and incident detection systems.

A real-time heterogeneous data processing system for TMS applications (Objective 6)

This research provided contributions in the areas of social media mining and real-time simulations of the transport network. It also contemplated the operation of the proposed components on real-time. To this end, an overall data processing architecture has been introduced, for the real-time implementation in TMS applications. It advances on previous research by providing a real-time operation of the social media component that fetches, processes, classifies, locates and extracts emotions from social media sources (Salas *et al.*, 2017). Results from the application of the real-time pipeline showed a substantial accuracy when identifying traffic events and issues on transport network from short texts.

In terms of the simulator component, a real-time processing pipeline was developed to prepare and calibrate large-scale simulation models. The proposed data processing architecture introduced tasks to automatically link traffic sensors with the relevant sections in the simulated network. Traffic data extracted from these sensors was transformed into the appropriate format of traffic demands stipulated by the traffic simulation tool, taking into consideration inconsistencies found in the traffic data. The optimisation procedure was then triggered and validated through the use of automatic simulations without graphical interface, thus reducing the simulation time. Findings from the implementation suggest the relevance of the processing pipeline for the calibration and validation of complex simulation models.

9.5 Recommendations for future research

From the results of this PhD thesis, a number of future research recommendation have arisen:

- Future work can explore the utilisation of enterprise streaming API (PowerTrack), to further explore the possibility of earlier detection of traffic incidents from Twitter data. The PowerTrack allows full access of real-time Twitter data, as opposed to the subset granted on the standard subscription.
- In this research, Google API was used as the knowledge base for the geolocation of tweets. However, sometimes it was not possible to locate the tweets because Google did not have the specific location of the junction. It will be interesting to develop a custom knowledge base that would possess the coordinates of all the junctions in the transport network.
- One of the main challenges faced by TensiStrength was not being able to identify sarcasm. As a result, a sample of tweets in the dataset were not tagged as negative, because the negative emotions were expressed through sarcasm. This can be taken into consideration for the future development of stress strength detection tools.
- The simulation tool can be easily adapted to other use cases. Future work can include adapting the proposed methodology to networks with different geometric characteristics. In addition, it would be interesting to evaluate the performance of other optimisation algorithms.

The framework could be extended in the future to include other variables that can influence traffic, such as weather information. The NLP pipeline can also be

implemented in other crowdsourcing platforms to identify traffic events from short texts.

References

- Abdelhaq, H., Sengstock, C. and Gertz, M. (2013) 'Eventweet: Online localized event detection from twitter', *Proceedings of the VLDB Endowment*, 6(12), pp. 1326-1329.
- Abel, F., Hauff, C., Houben, G., Stronkman, R. and Tao, K. (2012) *Twitcident: fighting fire with information from social web streams*. ACM, pp. 305.
- Abhik, D. and Toshniwal, D. (2013) *Sub-event detection during natural hazards using features of social media data*. ACM, pp. 783.
- Abma, B.J.M. (2009) *Evaluation of requirements management tools with support for traceability-based change impact analysis*.
- Agarwal, A., Xie, B., Vovsha, I., Rambow, O. and Passonneau, R. (2011) *Sentiment analysis of twitter data*. pp. 30.
- Agarwal, P., Vaithyanathan, R., Sharma, S. and Shroff, G. (2012) *Catching the Long-Tail: Extracting Local News Events from Twitter*.
- Ahmed, F. and Hawas, Y.E. (2013) *A Fuzzy Logic Model for Real-time Incident Detection in Urban Road Network*. pp. 465.
- Al Rillie, I. and Byard, N. (2006) *Investigation of Incident Related Congestion: Phase 1*. Project Report UPR. Available at: <http://www.haresearch.gov.uk/projects/index.php?id=1011> (Accessed: .
- Albino, V., Berardi, U. and Dangelico, R.M. (2015) 'Smart cities: Definitions, dimensions, performance, and initiatives', *Journal of urban technology*, 22(1), pp. 3-21.
- Alonso Raposo, M., Ciuffo, B., Ardente, F., Aurambout Jean, P., Baldini, G., Braun, R., Christidis, P., Christodoulou, A., Duboz, A., Felici, S., Ferragut Martinez Vara De Rey, Jaime, Georgakaki, A., Gkoumas, K., Grosso, M., Iglesias Portela, M., Julea Andreea, M., Krause, J., Martens, B., Mathieux, F., Menzel, G., Mondello, S., Navajas Cawood, E., Pekar, F., Raileanu, I., Scholz, H., Tamba, M., Tsakalidis, A., Van Balen, M. and Vandecasteele, I. (2019) 'The future of road transport', 29748. Available at: <https://ec.europa.eu/jrc/en/publication/eur-scientific-and-technical-research-reports/future-road-transport>
- Alsaedi, N. (2017) *Event identification in social media using classification-clustering framework*. . Cardiff University.
- Analytics (2009) *Twitter study*. Available at: www.pearanalytics.com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009.pdf

Anastasi, G., Antonelli, M., Bechini, A., Brienza, S., D'Andrea, E., De Guglielmo, D., Ducange, P., Lazzerini, B., Marcelloni, F. and Segatori, A. (2013) *Urban and social sensing for sustainable mobility in smart cities*. IEEE

Antoniou, C., Balakrishna, R. and Koutsopoulos, H.N. (2011) 'A synthesis of emerging data collection technologies and their impact on traffic management applications', *European Transport Research Review*, 3(3), pp. 139-148.

Aramaki, E., Maskawa, S. and Morita, M. (2011) *Twitter catches the flu: detecting influenza epidemics using Twitter*. Association for Computational Linguistics, pp. 1568.

Asakura, Y., Kusakabe, T., Nguyen, L.X. and Ushiki, T. (2017) *Incident detection methods using probe vehicles with on-board GPS equipment*.

Ashworth, C.M. (1988) *Structured systems analysis and design method (SSADM)*.

Asif, M.T., Mitrovic, N., Dauwels, J. and Jaillet, P. (2016) 'Matrix and tensor-based methods for missing data estimation in large traffic networks', *IEEE Transactions on intelligent transportation systems*, 17(7), pp. 1816-1825.

Asur, S. and Huberman, B. (2010) Predicting the future with social media. IEEE, pp. 492.

Atefeh, F. and Khreich, W. (2015) 'A Survey of Techniques for Event Detection in Twitter', *Computational Intelligence*, 31(1), pp. 132-164. doi: 10.1111/coin.12017.

Aulakh, S. (2018) Rating the UK's Motorways. Available at: <https://www.insurethegap.com/articles/rating-the-uks-motorways>

Avison, D.E. and Fitzgerald, G. (2003) *Information systems development : methodologies, techniques, and tools*. 3rd edn. McGraw-Hill London

Backlund, A. (2000) 'The definition of system', *Kybernetes*, 29(4), pp. 444-451.

Bae, B., Kim, H., Lim, H., Liu, Y., Han, L.D. and Freeze, P.B. (2018) *Missing data imputation for traffic flow speed using spatio-temporal cokriging*.

Bajpai, J.N. (2016) 'Emerging vehicle technologies & the search for urban mobility solutions', *Urban, Planning and Transport Research*, 4(1), pp. 83-100.

Barceló Bugada, J., Ferrer, J., Casas Pla, J.R., Montero Mercadé, L. and Perarnau, J. (2002) *Microscopic simulation with AIMSUN for the assessment of incident management strategies*.

Barceló, J., Codina, E., Casas, J., Ferrer, J.L. and García, D. (2005) 'Microscopic traffic simulation: A tool for the design, analysis and evaluation of intelligent transport systems', *Journal of Intelligent and Robotic Systems*, 41(2-3), pp. 173-203.

Barlovic, R., Esser, J., Froese, K., Knospe, W., Neubert, L., Schreckenberg, M. and Wahle, J. (1999) 'Online traffic simulation with cellular automata' *Traffic and Mobility* Springer, pp. 117-134.

Barrios, J.M., Hochberg, Y.V. and Yi, H. (2019) 'The cost of convenience: Ridesharing and traffic fatalities', https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3259965,

Ben-David, A. (2008) *About the relationship between ROC curves and Cohen's kappa*.

Bird, S., Klein, E. and Loper, E. (2009) *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc."

Blayney, E. (2016) *Louisville Leverages Crowdsourcing for Civic Good*. Available at: <https://datasmart.ash.harvard.edu/news/article/louisville-leverages-crowdsourcing-for-civic-good-919> (Accessed: Jan 31, 2018).

Boehm, B.W. (1988) 'A spiral model of software development and enhancement', *Computer*, (5), pp. 61-72.

Box, G.E. and Jenkins, G.M. (1976) *Time series analysis: forecasting and control*. Holden-Day.

Boxill, S.A. and Yu, L. (2000) 'An evaluation of traffic simulation models for supporting its', *Houston, TX: Development Centre for Transportation Training and Research, Texas Southern University*.

Brownlee, J. (2016) 'Overfitting and Underfitting With Machine Learning Algorithms', *Machine Learning Mastery*. Available at: <https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/> (Accessed: Mar 16, 2019).

Burghout, W., Koutsopoulos, H. and Andreasson, I. (2005) 'Hybrid mesoscopic-microscopic traffic simulation', *Transportation Research Record: Journal of the Transportation Research Board*, (1934), pp. 218-255.

Carpenter, T. and Way, T. (2012) *Tracking Sentiment Analysis through Twitter*. Citeseer.

Castro-Neto, M., Jeong, Y., Jeong, M. and Han, L.D. (2009) 'Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions', *Expert Systems with Applications*, 36(3), pp. 6164-6173.

- Chakraborty, P., Adu-Gyamfi, Y.O., Poddar, S., Ahsani, V., Sharma, A. and Sarkar, S. (2018) 'Traffic Congestion Detection from Camera Images using Deep Convolution Neural Networks', *Transportation Research Record*.
- Chawla, N.V., Japkowicz, N. and Kotcz, A. (2004) 'Special issue on learning from imbalanced data sets', *ACM Sigkdd Explorations Newsletter*, 6(1), pp. 1-6.
- Chen, B. and Cheng, H.H. (2010) 'A review of the applications of agent technology in traffic and transportation systems', *IEEE Transactions on intelligent transportation systems*, 11(2), pp. 485-497.
- Chen, X., Wei, Z., Li, Z., Liang, J., Cai, Y. and Zhang, B. (2017) 'Ensemble correlation-based low-rank matrix completion with applications to traffic data imputation', *Knowledge-Based Systems*, 132, pp. 249-262.
- Chester, M. and Athwall, A.K. (2002) *Basic information systems analysis and design*. McGraw-Hill Education London.
- Chiou, J., Zhang, Y., Chen, W. and Chang, C. (2014) 'A functional data approach to missing value imputation and outlier detection for traffic flow data', *Transportmetrica B: Transport Dynamics*, 2(2), pp. 106-129.
- Chowdhury, D., Santen, L. and Schadschneider, A. (2000) 'Statistical physics of vehicular traffic and some related systems', *Physics Reports*, 329(4-6), pp. 199-329.
- Clewlow, R.R. and Mishra, G.S. (2017) 'Disruptive transportation: The adoption, utilization, and impacts of ride-hailing in the United States'.
- Cohen, B. (2012) 'What exactly is a smart city?', September 19,. Available at: <http://www.fastcoexist.com/1680538/what-exactly-is-a-smart-city>
- Cohen, J. (1960) 'A coefficient of agreement for nominal scales', *Educational and psychological measurement*, 20(1), pp. 37-46.
- Comito, C., Forestiero, A. and Pizzuti, C. (2019) 'Bursty event detection in Twitter streams', *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(4), pp. 1-28.
- Cookson, G. and Pishue, B. (2017) 'Inrix global traffic scorecard', *INRIX Research, February*.
- Cortes, C. and Vapnik, V. (1995) 'Support-vector networks', *Machine Learning*, 20(3), pp. 273-297.
- D. Billings and J. Yang (2006) *Application of the ARIMA Models to Urban Roadway Travel Time Prediction - A Case Study*. pp. 2529.

Dalla Chiara, B., Deflorio, F. and Pinna, I. (2010) 'A comparing analysis of traffic monitoring systems', *EUROTRANSPORT*, (4).

D'Andrea, E. and Marcelloni, F. (2017) 'Detection of traffic congestion and incidents from GPS trace analysis', *Expert Systems with Applications*, 73, pp. 43-56. doi: 10.1016/j.eswa.2016.12.018.

D'Andrea, E., Ducange, P., Lazzerini, B. and Marcelloni, F. (2015) 'Real-Time Detection of Traffic From Twitter Stream Analysis', *IEEE Transactions on Intelligent Transportation Systems*, 16(4), pp. 2269-2283. doi: 10.1109/TITS.2015.2404431.

De Souza, A.M., Brennand, C.A., Yokoyama, R.S., Donato, E.A., Madeira, E.R. and Villas, L.A. (2017) 'Traffic management systems: A classification, review, challenges, and future perspectives', *International Journal of Distributed Sensor Networks*, 13(4), pp. 1550147716683612.

Debnath, A.K., Chin, H.C., Haque, M.M. and Yuen, B. (2014) *A methodological framework for benchmarking smart transport cities*.

Dia, H. (2001) 'An object-oriented neural network approach to short-term traffic forecasting', *European Journal of Operational Research*, 131(2), pp. 253-261. doi: //dx.doi.org/10.1016/S0377-2217(00)00125-9.

Dia, H. and Thomas, K. (2011) 'Development and evaluation of arterial incident detection models using fusion of simulated probe vehicle and loop detector data', *Information Fusion*, 12(1), pp. 20-27. doi: 10.1016/j.inffus.2010.01.001.

Dia, H., Gondwe, W. and Panwai, S., 2006. A traffic simulation approach to evaluating the benefits of incident management programs. *ATRF*.

Djahel, S., Doolan, R., Muntean, G. and Murphy, J. (2015) 'A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches', *IEEE Communications Surveys & Tutorials*, 17.

Dogru, N. and Subasi, A. (2018) *Traffic accident detection using random forest classifier*. IEEE, pp. 40.

Drew, D.R. (1968) 'Traffic flow theory and control', *Traffic flow theory and control*, .

Duan, Y., Lv, Y., Liu, Y. and Wang, F. (2016) *An efficient realization of deep learning for traffic data imputation*.

E. D'Andrea, P. Ducange, B. Lazzerini and F. Marcelloni (2015) 'Real-Time Detection of Traffic From Twitter Stream Analysis', *IEEE Transactions on Intelligent Transportation Systems*, 16(4), pp. 2269-2283. doi: 10.1109/TITS.2015.2404431.

Edie, L.C. (1963) *Discussion of traffic stream measurements and definitions*. Port of New York Authority.

Ehlert, P.A. and Rothkrantz, L.J. (2001) *Microscopic traffic simulation with reactive driving agents*. pp. 861.

Elliott, D., Keen, W. and Miao, L. (2019) *Recent advances in connected and automated vehicles*.

European Commission (2013) *Horizon 2020*. Available at: <https://ec.europa.eu/programmes/horizon2020/en/h2020-section/smart-green-and-integrated-transport> (Accessed: 15 October 2019).

European Commission (2019a) *EU Transport in Figures: Statistical Pocketbook 2019*. Available at: https://ec.europa.eu/transport/facts-fundings/statistics/pocketbook-2019_en (Accessed: 15 October 2019).

European Commission (2019b) *Passenger transport statistics*. Available at: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Passenger_transport_statistics&oldid=404334 (Accessed: 10 October 2019).

European Commission (2019c) *Road safety statistics*. Available at: https://ec.europa.eu/commission/news/road-safety-2019-apr-04_en (Accessed: 16 October 2019).

Evans, D. (2011) 'The internet of things: How the next evolution of the internet is changing everything', *CISCO white paper*, 1(2011), pp. 1-11.

Ezzat, A.A., Farouk, H.A., El-Kilany, K.S. and Abdelmoneim, A.F. (2014) *Development of a Stochastic Genetic Algorithm for Traffic Signal Timings Optimisation*.

Farkas, K., Feher, G., Benczur, A. and Sidlo, C. (2015) 'Crowdsending based public transport information service in smart cities', *IEEE Communications Magazine*, 53(8), pp. 158-165.

FHWA (2004) *Traffic incident management*. Available at: https://ops.fhwa.dot.gov/aboutus/one_pagers/tim.htm

FHWA (2018) *2017 Urban Congestion Trends: Measuring, Managing, and Improving Operations in the 21st Century*. Available at: <https://ops.fhwa.dot.gov/publications/fhwahop18025/index.htm>

Figueiredo, L., Jesus, I., Machado, J.T., Ferreira, J. and de Carvalho, J.M. (2001) *Towards the development of intelligent transportation systems*. pp. 1206.

- Fonseca, D.J., Brumback, T., Moynihan, G. and Fernandes, H. (2009) *Development of a Multi-Algorithmic Platform for Traffic Incident Detection*. International Conference on Industry, Engineering, & Management Systems, pp. 42.
- Friedman, N., Geiger, D. and Goldszmidt, M. (1997) 'Bayesian network classifiers', *Machine Learning*, 29(2-3), pp. 131-163.
- Garcia Esparza, S., O'Mahony, M.P. and Smyth, B. (2012) *Mining the real-time web: A novel approach to product recommendation*.
- Gardner, M.W. and Dorling, S.R. (1998) *Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences*.
- Gerlough, D.L. and Huber, M.J. (1976) *Traffic flow theory*.
- Gettman, D. and Head, L. (2003) 'Surrogate safety measures from traffic simulation models', *Transportation Research Record: Journal of the Transportation Research Board*, (1840), pp. 104-115.
- Ghaisani, A.P., Handayani, P.W. and Munajat, Q. (2017) *Users' Motivation in Sharing Information on Social Media*.
- Gharaibeh, A., Salahuddin, M.A., Hussini, S.J., Khreishah, A., Khalil, I., Guizani, M. and Al-Fuqaha, A. (2017) 'Smart cities: A survey on data management, security, and enabling technologies', *IEEE Communications Surveys & Tutorials*, 19(4), pp. 2456-2501.
- Ghosh, B., Basu, B. and O'Mahony, M. (2007) 'Bayesian time-series model for short-term traffic flow forecasting', *Journal of Transportation Engineering*, 133(3), pp. 180-189.
- Goves, C., North, R., Johnston, R. and Fletcher, G. (2016) *Short Term Traffic Prediction on the UK Motorway Network Using Neural Networks*.
- Gregor, S. and Hevner, A.R. (2013) 'Positioning and Presenting Design Science-Types of Knowledge in Design Science Research', *MIS Q*, 37(2), pp. 337-355.
- Gu, Y., Qian, Z. and Chen, F. (2016) 'From Twitter to detector: Real-time traffic incident detection using social media data', *Transportation Research Part C: Emerging Technologies*, 67, pp. 321-342. doi: 10.1016/j.trc.2016.02.011.
- Guerrero-Ibañez, J., Zeadally, S. and Contreras-Castillo, J. (2018) 'Sensor technologies for intelligent transportation systems', *Sensors*, 18(4), pp. 1212.
- Guo, F. (2013) *Short-term traffic prediction under normal and abnormal conditions*. Imperial College London.

- Gutierrez, C., Figuerias, P., Oliveira, P., Costa, R. and Jardim-Goncalves, R. (2015) 'Twitter mining for traffic events detection', *Proceedings of the 2015 Science and Information Conference, SAI 2015*, , pp. 371-378. doi: 10.1109/SAI.2015.7237170.
- Hadi, M., Sinha, P. and Wang, A. (2007) 'Modeling reductions in freeway capacity due to incidents in microscopic simulation models', *Transportation Research Record*, 1999(1), pp. 62-68.
- He, J., Shen, W., Divakaruni, P., Wynter, L. and Lawrence, R. (2013) Improving traffic prediction with tweet semantics. AAAI Press, pp. 1387.
- Hecht-Nielsen, R. (1988) 'Neurocomputing: picking the human brain', *IEEE Spectrum*, 25(3), pp. 36-41. doi: 10.1109/6.4520.
- Helman, D.L. (2004) 'Traffic incident management', *Public Roads*, 68(3).
- Hevner, A. and Chatterjee, S. (2010) 'Design science research in information systems' *Design research in information systems* Springer, pp. 9-22.
- Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004) 'Design Science in Information Systems Research', *MIS Quarterly*, 28(1), pp. 75-105.
- Hoadley, S. (2018) *Road Vehicle Automation and cities and regions*. Brussels. Available at: https://www.polisnetwork.eu/uploads/Modules/PublicDocuments/polis_discussion_paper_automated_vehicles.pdf (Accessed: .
- Hoerl, A.E. and Kennard, R.W. (1970) 'Ridge Regression: Biased Estimation for Nonorthogonal Problems', *Technometrics*, 12(1), pp. 55-67. doi: 10.1080/00401706.1970.10488634.
- Hong, W. (2011) 'Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm', *Neurocomputing*, 74(12–13), pp. 2096-2107. doi: //dx.doi.org/10.1016/j.neucom.2010.12.032.
- J. He, L. Ding, L. Jiang and L. Ma (2014) *Kernel ridge regression classification*. pp. 2263.
- Java, A., Song, X., Finin, T. and Tseng, B. (2007) *Why we twitter: understanding microblogging usage and communities*. ACM, pp. 56.
- Jiang, X. and Adeli, H. (2005) 'Dynamic Wavelet Neural Network Model for Traffic Flow Forecasting', *Journal of Transportation Engineering*, 131(10), pp. 771-779. doi: 10(771).
- Jin, X., Cheu, R.L. and Srinivasan, D. (2002) *Development and adaptation of constructive probabilistic neural network in freeway incident detection*.

Jin, X., Zhang, Y. and Yao, D. (2007) *Simultaneously prediction of network traffic flow based on PCA-SVR*. Springer, pp. 1022.

Kamarianakis, Y. and Prastacos, P. (2003) 'Forecasting Traffic Flow Conditions in an Urban Network: Comparison of Multivariate and Univariate Approaches', *Transportation Research Record: Journal of the Transportation Research Board*, 1857, pp. 74-84. doi: 10.3141/1857-09.

Karlaftis, M.G. and Vlahogianni, E.I. (2011) *Statistical methods versus neural networks in transportation research: Differences, similarities and some insights*.

Karlaftis, M.G., Latoski, S.P., Richards, N.J. and Sinha, K.C. (1999) 'ITS impacts on safety and traffic management: an investigation of secondary crash causes', *Journal of Intelligent Transportation Systems*, 5(1), pp. 39-52.

Kell, J.H., Fullerton, I.J. and Mills, M.K. (1990) *Traffic detector handbook*.

Khattak, A., Wang, X. and Zhang, H. (2012) 'Incident management integration tool: dynamically predicting incident durations, secondary incident occurrence and incident delays', *IET Intelligent Transport Systems*, 6(2), pp. 204-214.

Ki, Y., Heo, N., Choi, J., Ahn, G. and Park, K. (2018) *An incident detection algorithm using artificial neural networks and traffic information*. IEEE, pp. 1.

Kim, H., Golub, G.H. and Park, H. (2004) 'Missing value estimation for DNA microarray gene expression data: local least squares imputation', *Bioinformatics*, 21(2), pp. 187-198.

Kinoshita, A., Takasu, A. and Adachi, J. (2015) 'Real-time traffic incident detection using a probabilistic topic model', *Information Systems*, . doi: 10.1016/j.is.2015.07.002.

Kitschke, J.H. (2014) 'Investigation of the Convergence Behaviour of Different Optimisation Algorithms using Shape Optimisation in Flow Problems'.

Klein, L.A., Mills, M.K., Gibson, D. and Klein, L.A. (2006) *Traffic detector handbook: Volume IIIe*. United States. Federal Highway Administration.

Knight, W. (2016) *The Demographics of Social Media Users in 2016*. Available at: <http://www.thinkdigitalfirst.com/2016/01/04/the-demographics-of-social-media-users-in-2016/> (Accessed: Jun 25, 2017).

Kraft, D., 1987. *Optimal estimation with an introduction to stochastic control theory*: Frank L. Lewis.

Krstajic, M., Rohrdantz, C., Hund, M. and Weiler, A. (2012) 'Getting there first: Real-time detection of real-world incidents on twitter'.

Kumar, K., Parida, M. and Katiyar, V.K. (2013) *Short Term Traffic Flow Prediction for a Non Urban Highway Using Artificial Neural Network*.

Kwak, H., Lee, C., Park, H. and Moon, S. (2010) *What is Twitter, a social network or a news media?* ACM, pp. 591.

Lamos, V. (2012) 'Detecting events and patterns in large-scale user generated textual streams with statistical learning methods', *arXiv preprint arXiv:1208.2873*.

Landis, J.R. and Koch, G.G. (1977) 'The measurement of observer agreement for categorical data', *Biometrics*, , pp. 159-174.

Lantz, K.E. (1986) *The prototyping methodology*. Prentice-Hall, Inc.

Le Vine, S., Adamou, O. and Polak, J. (2014) 'Predicting new forms of activity/mobility patterns enabled by shared-mobility services through a needs-based stated-response method: Case study of grocery shopping', *Transport Policy*, 32, pp. 60-68.

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

Lee, S. and Fambro, D. (1999) 'Application of Subset Autoregressive Integrated Moving Average Model for Short-Term Freeway Traffic Volume Forecasting', *Transportation Research Record: Journal of the Transportation Research Board*, 1678, pp. 179-188. doi: 10.3141/1678-22.

Lee, S., Krammes, R.A. and Yen, J. (1998) 'Fuzzy-logic-based incident detection for signalized diamond interchanges', *Transportation Research Part C: Emerging Technologies*, 6(5-6), pp. 359-377.

Lewis, B.K. (2009) *Social media and strategic communication: Attitudes and perceptions among college students*. Oklahoma State University.

Li, Y., Li, Z. and Li, L. (2014) 'Missing traffic data: comparison of imputation methods', *IET Intelligent Transport Systems*, 8(1), pp. 51-57.

Li, Z., Al Hassan, R., Shahidehpour, M., Bahramirad, S. and Khodaei, A. (2017) 'A hierarchical framework for intelligent traffic management in smart cities', *IEEE Transactions on Smart Grid*, 10(1), pp. 691-701.

Li, Z., Shahidehpour, M., Bahramirad, S. and Khodaei, A. (2016) 'Optimizing traffic signal settings in smart cities', *IEEE Transactions on Smart Grid*, 8(5), pp. 2382-2393.

Lianyu Chu, Liu, H.X., Jun-Seok Oh and Recker, W. (2003) *A calibration procedure for microscopic traffic simulation*. IEEE.

- Lighthill, M.J. and Whitham, G.B. (1955) 'On kinematic waves II. A theory of traffic flow on long crowded roads', *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178), pp. 317-345.
- Lin, W., Wilson, T., Wiebe, J. and Hauptmann, A. (2006) *Which side are you on?: identifying perspectives at the document and sentence levels*. Association for Computational Linguistics, pp. 109.
- Liu, B. (2012) 'Sentiment analysis and opinion mining', *Synthesis lectures on human language technologies*, 5(1), pp. 1-167.
- Liu, Z., Sharma, S. and Datla, S. (2008) 'Imputation of missing traffic data during holiday periods', *Transportation Planning and Technology*, 31(5), pp. 525-544.
- Lopes, J., Bento, J., Huang, E., Antoniou, C. and Ben-Akiva, M. (2010) *Traffic and mobility data collection for real-time applications*. IEEE, pp. 216.
- Maerivoet, S. and De Moor, B. (2005) *Cellular automata models of road traffic*.
- Mahmassain, H.S., Haas, C., Zhou, S. and Peterman, J. (1999) *Evaluation of incident detection methodologies*. University of Texas at Austin. Center for Transportation Research. Available at: (Accessed: .
- Mahmud, S.M.S., Ferreira, L., Hoque, M.S. and Tavassoli, A. (2018) *Micro-simulation modelling for traffic safety: A review and potential application to heterogeneous traffic environment*.
- Mannering, F.L., Washburn, S.S. and Kilareski, W.P. (2009) *Principles of highway engineering and traffic analysis*. 4th edn. Wiley Hoboken, N.J.
- March, S.T. and Smith, G.F. (1995) *Design and natural science research on information technology*.
- Margineantu, D.D. and Dietterich, T.G. (2000) 'Bootstrap methods for the cost-sensitive evaluation of classifiers', .
- Masek, P., Masek, J., Frantik, P., Fujdiak, R., Ometov, A., Hosek, J., Andreev, S., Mlynek, P. and Misurec, J. (2016) 'A harmonized perspective on transportation management in smart cities: The novel IoT-driven environment for road traffic modeling', *Sensors*, 16(11), pp. 1872.
- Mathew, T. (2014) *Lectures notes in Traffic Engineering and Management* Indian Institute of Technology Bombay.
- McCallum, A. and Nigam, K. (1998) *A comparison of event models for naive bayes text classification*. Madison, WI, pp. 41.

- McCarthy, J., Minsky, M.L., Rochester, N. and Shannon, C.E. (2006) 'A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955', *AI magazine*, 27(4), pp. 12.
- Middleton, D., Gopalakrishna, D. and Raman, M. (2003) *Advances in traffic data collection and management*. Citeseer.
- Mimbela, L.E.Y. and Klein, L.A. (2007) 'Summary of vehicle detection and surveillance technologies used in intelligent transportation systems', .
- Minami, M., Kato, R., Hagiwara, T., Araki, K., Nagata, Y. and Takitani, K. (2008) 'Development of a visibility estimation model based on visibility information from road images captured in winter', .
- Motta, G., Sacco, D., Ma, T., You, L. and Liu, K. (2015) *Personal mobility service system in urban areas: the IRMA project*. IEEE, pp. 88.
- Muralidharan, A. and Horowitz, R. (2009) 'Imputation of ramp flow data for freeway traffic simulation', *Transportation Research Record: Journal of the Transportation Research Board*, (2009), pp. 58-64.
- Nelson, D. (2000) 'ITS subsystems and technologies in managing traffic, vehicles and systems., chapter 14', *Intelligent Transportation Primer*.
- Ni, D. and Leonard II, J.D. (2004) 'Markov Chain Monte Carlo Multiple Imputation for Incomplete ITS Data Using Bayesian Networks', *TRB 2005*.
- Nichols, J., Mahmud, J. and Drews, C. (2012) *Summarizing sporting events using twitter*. ACM, pp. 189.
- Nihan, N.L. and Holmesland, K.O. (1980) 'Use of the Box and Jenkins time series technique in traffic forecasting', *Transportation*, 9(2), pp. 125-143.
- Nikolaev, A.B., Sapego, Y.S., Jakubovich, A.N., Berner, L.I. and Stroganov, V.Y. (2016) 'Fuzzy Algorithm for the Detection of Incidents in the Transport System.', *International Journal of Environmental and Science Education*, 11(16), pp. 9039-9059.
- Osborne, M., Moran, S., McCreadie, R., Von Lunen, A., Sykora, M.D., Cano, E., Ireson, N., Macdonald, C., Ounis, I. and He, Y. (2014) 'Real-time detection, tracking, and monitoring of automatically discovered events in social media', .
- Pak, A. and Paroubek, P. (2010) *Twitter as a Corpus for Sentiment Analysis and Opinion Mining*.

- Panwai, S. and Dia, H. (2005) 'Comparative evaluation of microscopic car-following behavior', *IEEE Transactions on Intelligent Transportation Systems*, 6(3), pp. 314-325.
- Park, B. (2002) 'Hybrid neuro-fuzzy application in short-term freeway traffic volume forecasting', *Transportation Research Record: Journal of the Transportation Research Board*, (1802), pp. 190-196.
- Parkany, E. and Xie, C. (2005) *A complete review of incident detection algorithms & their deployment: what works and what doesn't*. Available at:(Accessed: .
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R. and Dubourg, V. (2011) 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, 12(Oct), pp. 2825-2830.
- Peppers, K., Tuunanen, T., Rothenberger, M.A. and Chatterjee, S. (2007) 'A Design Science Research Methodology for Information Systems Research', *Journal of Management Information Systems*, 24(3), pp. 45-77. doi: 10.2753/MIS0742-1222240302.
- Petrović, S., Osborne, M. and Lavrenko, V. (2010) *Streaming first story detection with application to twitter*. Association for Computational Linguistics, pp. 181.
- Popescu, A. and Pennacchiotti, M. (2010) *Detecting controversial events from twitter*. ACM, pp. 1873.
- Porter, M.F. (1980) 'An algorithm for suffix stripping', *Program*, 14(3), pp. 130-137.
- Porter, M.F. (2001) 'Snowball: A language for stemming algorithms', .
- Psallidas, F., Becker, H., Naaman, M. and Gravano, L. (2013) 'Effective Event Identification in Social Media.', *IEEE Data Eng.Bull.*, 36(3), pp. 42-50.
- PTV (2009) *VISSIM 5.20 User Manual*.
- Purao, S. (2002) 'Design research in the technology of information systems: Truth or dare', *GSU Department of CIS Working Paper*, , pp. 45-77.
- Qu, L., Li, L., Zhang, Y. and Hu, J. (2009a) 'PPCA-based missing data imputation for traffic flow volume: A systematical approach', *IEEE Transactions on intelligent transportation systems*, 10(3), pp. 512-522.
- Quadstone. (2003) *Quadstone Paramics V4.2: Analyser Reference Manual*
- Quek, C., Pasquier, M. and Lim, B. (2009) *A novel self-organizing fuzzy rule-based system for modelling traffic flow behaviour*.

- Reed, T. and Kidd, J. (2019) *Global Traffic Scorecard*.
- Rijkswaterstaat (2015) *Cooperative ITS Corridor: Probe Vehicle Data*. Available at: https://itscorridor.mett.nl/home+_eng/cits+corridor_eng/Project+details/Cooperative+services/Probe+Vehicle+Data/default.aspx
- Ritchie, S.G. and Cheu, R.L. (1993) *Simulation Of Freeway Incident Detection Using Artificial Neural Networks*.
- Rosen, A. (2017) 'Tweeting Made Easier', *Twitter*, November. Available at: https://blog.twitter.com/en_us/topics/product/2017/tweetingmadeeasier.html
- Rosenblatt, F. (1958) 'The perceptron: a probabilistic model for information storage and organization in the brain.', *Psychological review*, 65(6), pp. 386.
- Rosenblatt, F. (1961) *Principles of neurodynamics. Perceptrons and the theory of brain mechanisms*. CORNELL AERONAUTICAL LAB INC BUFFALO NY.
- Rossi, R., Gastaldi, M., Gecchele, G. and Barbaro, V. (2015) *Fuzzy Logic-based Incident Detection System using Loop Detectors Data*.
- Royce, W. (1970) *The software lifecycle model (Waterfall Model)*.
- Saidallah, M., El Fergougui, A. and Elalaoui, A.E. (2016) *A comparative study of urban road traffic simulators*. EDP Sciences, pp. 05002.
- Saka, A.A. (2000) *Introduction to Intelligent Transportation Systems*.
- Sakaki, T., Okazaki, M. and Matsuo, Y. (2010) *Earthquake shakes Twitter users: real-time event detection by social sensors*. ACM, pp. 851.
- Schulz, A., Ristoski, P. and Paulheim, H. (2013) *I See a Car Crash: Real-Time Detection of Small Scale Incidents in Microblogs*. Springer, Berlin, Heidelberg, pp. 22.
- Shaheen, S. and Chan, N. (2016) 'Mobility and the Sharing Economy: Potential to Overcome First-and Last-Mile Public Transit Connections'.
- Shang, Q., Yang, Z., Gao, S. and Tan, D. (2018) 'An Imputation Method for Missing Traffic Data Based on FCM Optimized by PSO-SVR', *Journal of Advanced Transportation*, 2018.
- Shen, C., Liu, F., Weng, F. and Li, T. (2013) *A participant-based approach for event summarization using twitter streams*. pp. 1152.
- Siripanpornchana, C., Panichpapiboon, S. and Chaovalit, P. (2016) *Incidents detection through mobile sensing*. IEEE.

Smith, B.L., Scherer, W.T. and Conklin, J.H. (2003) 'Exploring imputation techniques for missing data in transportation management systems', *Transportation Research Record*, 1836(1), pp. 132-142.

Smith, B.L., Williams, B.M. and Keith Oswald, R. (2002) 'Comparison of parametric and nonparametric models for traffic flow forecasting', *Transportation Research Part C: Emerging Technologies*, 10(4), pp. 303-321. doi: //dx.doi.org/10.1016/S0968-090X(02)00009-8.

Srinivasan, D., Sanyal, S. and Sharma, V. (2007) 'Freeway incident detection using hybrid fuzzy neural network', *IET Intelligent Transport Systems*, 1(4), pp. 249.

Srivastava, T. (2014) 'Classification Algorithm Support Vector Machine', *Analytics Vidhya*, -10-03T03:23:21+05:30. Available at: <https://www.analyticsvidhya.com/blog/2014/10/support-vector-machine-simplified/> (Accessed: Oct 27, 2017).

Starbird, K. and Palen, L. (2012) *(How) will the revolution be retweeted?: information diffusion and the 2011 Egyptian uprising*. ACM, pp. 7.

Stathopoulos, A., Dimitriou, L. and Tsekeris, T. (2008) 'Fuzzy Modeling Approach for Combined Forecasting of Urban Traffic Flow', *Computer-Aided Civil and Infrastructure Engineering*, 23(7), pp. 521-535. doi: 10.1111/j.1467-8667.2008.00558.x.

Stylios, G., Christodoulakis, D., Besharat, J., Vonitsanou, M., Kotrotsos, I., Koumpouri, A. and Stamou, S. (2010) 'Public opinion mining for governmental decisions', *Electronic Journal of e-Government*, 8(2), pp. 202.

T. Cover and P. Hart (1967) 'Nearest neighbor pattern classification', *IEEE Transactions on Information Theory*, 13(1), pp. 21-27. doi: 10.1109/TIT.1967.1053964.

Talebpour, A. and Mahmassani, H.S. (2016) *Influence of connected and autonomous vehicles on traffic flow stability and throughput*.

Tan, H., Feng, G., Feng, J., Wang, W., Zhang, Y. and Li, F. (2013) 'A tensor-based method for missing traffic data completion', *Transportation Research Part C: Emerging Technologies*, 28, pp. 15-27.

Tang, J., Liu, F., Zou, Y., Zhang, W. and Wang, Y. (2017) 'An improved fuzzy neural network for traffic speed prediction considering periodic characteristic', *IEEE Transactions on Intelligent Transportation Systems*, 18(9), pp. 2340-2350.

Thelwall, M. (2017) 'TensiStrength: Stress and relaxation magnitude detection for social media texts', *Information Processing & Management*, 53(1), pp. 106-121.

- Thelwall, M. (2017) 'The Heart and soul of the web? Sentiment strength detection in the social web with SentiStrength' *Cyberemotions* Springer, pp. 119-134.
- Thelwall, M., Buckley, K. and Paltoglou, G. (2012) 'Sentiment strength detection for the social web', *Journal of the American Society for Information Science and Technology*, 63(1), pp. 163-173.
- Thelwall, M., Buckley, K., Paltoglou, G., Cai, D. and Kappas, A. (2010) 'Sentiment strength detection in short informal text', *Journal of the American Society for Information Science and Technology*, 61(12), pp. 2544-2558.
- TSS—transport simulation systems. (2008) *Aimsun Version 6 User's Manual*
- Tumasjan, A., Sprenger, T.O., Sandner, P.G. and Welpe, I.M. (2010) 'Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment.', *ICWSM*, 10, pp. 178-185.
- Twitter (2016) . Available at: <https://business.twitter.com/basics/what-is-twitter/>
- UN-Habitat (2016) *World cities report 2016: Urbanization and development—emerging futures*. United Nations Human Settlements Programme Nairobi. Available at: <http://wcr.unhabitat.org/main-report/>
- United Nations (2019) *World Population Prospects 2019: Highlights*. Department of Economic and Social Affairs, Population Division. Available at: <https://www.un.org/en/sections/issues-depth/population/>
- Valacich, J., George, J. and Hoffer, J.A. (2013) *Essentials of Systems Analysis and Design*. 5th edn. Pearson Education Limited Harlow.
- Van Der Voort, M., Dougherty, M. and Watson, S. (1996) *Combining kohonen maps with arima time series models to forecast traffic flow*.
- Vlahogianni, E.I., Karlaftis, M.G. and Golias, J.C. (2005) 'Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach', *Transportation Research Part C: Emerging Technologies*, 13(3), pp. 211-234. doi: [//dx.doi.org/10.1016/j.trc.2005.04.007](https://doi.org/10.1016/j.trc.2005.04.007).
- Wadud, Z., MacKenzie, D. and Leiby, P. (2016) *Help or hindrance? The travel, energy and carbon impacts of highly automated vehicles*.
- Wang, J., Li, X., Liao, S.S. and Hua, Z. (2013) 'A Hybrid Approach for Automatic Incident Detection', *IEEE Transactions on Intelligent Transportation Systems*, 14(3), pp. 1176-1185. doi: 10.1109/TITS.2013.2255594.
- Wang, X. (2016) *Real-time Content Identification for Events and Sub-Events from Microblogs*. . Queen Mary University of London.

- Wanichayapong, N., Pruthipunyaskul, W., Pattara-Atikom, W. and Chaovalit, P. (2011) *Social-based traffic information extraction and classification*. pp. 107.
- Wanner, F., Stoffel, A., Jäckle, D., Kwon, B.C., Weiler, A., Keim, D.A., Isaacs, K.E., Giménez, A., Jusufi, I. and Gamblin, T. (2014) *State-of-the-Art Report of Visual Analysis for Event Detection in Text Data Streams*.
- Wardrop, J.G. (1952) 'Road paper. some theoretical aspects of road traffic research.', *Proceedings of the institution of civil engineers*, 1(3), pp. 325-362.
- Wen, H., Yang, Z., Jiang, G. and Shao, C. (2001) *A new algorithm of incident detection on freeways*. IEEE, pp. 197.
- Wener, R.E. and Evans, G.W. (2011) *Comparing stress of car and train commuters*.
- Weng, J. and Lee, B. (2011) 'Event Detection in Twitter.', *ICWSM*, 11, pp. 401-408.
- Wibisono, A., Sina, I., Ihsannuddin, M.A., Hafizh, A., Hardjono, B., Nurhadiyatna, A. and Jatmiko, W. (2012) *Traffic intelligent system architecture based on social media information*. IEEE, pp. 25.
- Wieringa, R.J. (2014) *Design science methodology for information systems and software engineering*. Springer.
- Williams, B. and Guin, A. (2007) 'Traffic Management Center Use of Incident Detection Algorithms: Findings of a Nationwide Survey', *IEEE Transactions on Intelligent Transportation Systems*, 8(2), pp. 351-358. doi: 10.1109/TITS.2007.894193.
- Williams, B., Durvasula, P. and Brown, D. (1998) 'Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models', *Transportation Research Record: Journal of the Transportation Research Board*, (1644), pp. 132-141.
- Williams, B.M. and Hoel, L.A. (2003) 'Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results', *Journal of Transportation Engineering*, 129(6), pp. 664-672.
- Xiao, H., Ambadipudi, R., Hourdakis, J. and Michalopoulos, P. (2005) 'Methodology for selecting microscopic simulators: Comparative evaluation of AIMSUN and VISSIM', .
- Xie, W., Zhu, F., Jiang, J., Lim, E. and Wang, K. (2016) 'Topicsketch: Real-time bursty topic detection from twitter', *IEEE Transactions on Knowledge and Data Engineering*, 28(8), pp. 2216-2229.

- Xu, C., Liu, P., Yang, B. and Wang, W. (2016) *Real-time estimation of secondary crash likelihood on freeways using high-resolution loop detector data.*
- Yang, H., Bartin, B. and Ozbay, K. (2013) 'Use of sensor data to identify secondary crashes on freeways', *Transportation Research Record: Journal of the Transportation Research Board*, (2396), pp. 82-92.
- Ye, L. and Yamamoto, T. (2018) *Modeling connected and autonomous vehicles in heterogeneous traffic flow.*
- Yin, H., Wong, S.C., Xu, J. and Wong, C.K. (2002) 'Urban traffic flow prediction using a fuzzy-neural approach', *Transportation Research Part C: Emerging Technologies*, 10(2), pp. 85-98. doi: //dx.doi.org/10.1016/S0968-090X(01)00004-3.
- Yin, W., Murray-Tuite, P. and Rakha, H. (2012) 'Imputing erroneous data of single-station loop detectors for nonincident conditions: Comparison between temporal and spatial methods', *Journal of Intelligent Transportation Systems*, 16(3), pp. 159-176.
- Young, W., Sobhani, A., Lenné, M.G. and Sarvi, M. (2014) *Simulation of safety: A review of the state of the art in road safety simulation modelling.*
- Yu, L., Yu, L., Wang, J., Yu, L., Qi, Y. and Wen, H. (2008) *Back-Propagation Neural Network for Traffic Incident Detection Based on Fusion of Loop Detector and Probe Vehicle Data.* pp. 116.
- Zadeh, L.A. (1996) 'Soft computing and fuzzy logic' *Fuzzy Sets, Fuzzy Logic, And Fuzzy Systems: Selected Papers by Lotfi A Zadeh* World Scientific, pp. 796-804.
- Zanella, A., Bui, N., Castellani, A., Vangelista, L. and Zorzi, M. (2014) 'Internet of things for smart cities', *IEEE Internet of Things journal*, 1(1), pp. 22-32.
- Zhang, C., Sun, S. and Yu, G. (2004) *A Bayesian network approach to time series forecasting of short-term traffic flows.* IEEE, pp. 216.
- Zhang, J. and Hounsell, N.B. (2010) 'A comparison study on environmental impacts caused by bus signal priority strategies'.
- Zhang, Z., He, Q., Gao, J. and Ni, M. (2018) *A deep learning approach for detecting traffic accidents from social media data.*
- Zhong, M., Sharma, S. and Lingras, P. (2004) 'Genetically designed models for accurate imputation of missing traffic counts', *Transportation Research Record*, 1879(1), pp. 71-79.
- Zhong, M., Sharma, S. and Liu, Z. (2005) 'Assessing robustness of imputation models based on data from different jurisdictions: examples of Alberta and Saskatchewan,

Canada', *Transportation Research Record: Journal of the Transportation Research Board*, (1917), pp. 116-126.

Zhou, Y., Dey, K.C., Chowdhury, M. and Wang, K. (2016) 'Process for evaluating the data transfer performance of wireless traffic sensors for real-time intelligent transportation systems applications', *IET Intelligent Transport Systems*, 11(1), pp. 18-27.

Zhu, L., Guo, F., Krishnan, R. and Polak, J.W. (2018) *The Use of Convolutional Neural Networks for Traffic Incident Detection at a Network Level*.

Zhu, L., Yu, F.R., Wang, Y., Ning, B. and Tang, T. (2018) 'Big data analytics in intelligent transportation systems: A survey', *IEEE Transactions on Intelligent Transportation Systems*, 20(1), pp. 383-398.

Appendices

Appendix 5-A

List of traffic related keywords

M1	delays	birmingham traffic
M2	delay	black country
M3	congestion	blake street
M4	congestions	blakest newtn
M5	roadworks	blakest queslet
M6	roadwork	blke street
M9	road work	blossomfield road
M40	road works	bloxwich lane
M42	traffic	blue lane
M69	jam	blue way
A34	traffic jam	booths farm
A38	accident	bordersley circus
A41	accidents	brandon road
A45	crash	brdsly circ
A47	car crash	brdwy nth
A428	frankley	brewery road
A446	a41 solihull	brighton street
A453	a41 warwick	bristol road
A454	a45 coventry	bristol street
A456	a46 northbound	brm road
A491	a4601 m6	bromley street
A4123	alcester road	burnt tree
A4400	aston lane	bypass lode
Queensway	atherstone west	chester road
Gravelly hill	barr stechford	cotteridge bearwood
interchange	barr yardley	cotteridge stechford
Wolverhampton ring	bell lane	cotteridge yardly
road	bescot road	country route
wolverhampton road	birmingham accident	coventry road
wood bearwd	birmingham	coventry road
wood stechfd	congestion	cw bearwood
wordsworth road	birmingham delay	cw cotteridge
wrentham street	birmingham delays	dudley road
wy alcester	birmingham	foxoak street
yardley wood	roadworks	frankley services
yardly wood	birmingham roadwork	hagley road

hampton lane
herbert austin
hessian close
high street
hilton park
holyhead road
jockey road
junction b4116
kbry road
kings road
kingsbury road
knps hghwy/grn
ladywood middleway

lichfield road
spring hill
springfield lane
stafford road
station road
stbdg road
stechford lane
stechford road
stratford a34
strfd road
stuck traffic
stuck in traffic
stuck road

TextBTP
toll northbound
toll southbound
Traffeek
venture way
vicarage road
walsal road
walsall road
walsall street
walsallrd pbeeches
warwick road

Appendix 5-B

Regular expressions pre-processing chain

```
emoticons_str = r""""
(?:
  [:=;] # Eyes
  [oO\~]? # Nose (optional)
  [D\)\]\(\)\[\]\OpP] # Mouth
)""""

regex_str = [
  emoticons_str,
  r'<[^>]+>', # HTML tags
  r'(?:@[\w_]+)', # @-mentions
  r'(?:\#[\w_]+[\w'\_\-]*[\w_]+)', # hash-tags
  r'http[s]?://(?:[a-z]|[0-9]|[$_-@.&+]|[*\(\),]|(?:%[0-9a-f][0-9a-f]))+', # URLs
  r'(?:(?:\d+)?(?:\.\d+)?)', # numbers
  r'(?:[a-z][a-z'\_\-]+[a-z])', # words with - and '
  r'(?:[\w_]+)', # other words
  r'(?:\S)' # anything else
]

tokens_re = re.compile(r'('+'.join(regex_str)+)', re.VERBOSE | re.IGNORECASE)
emoticon_re = re.compile(r'^'+emoticons_str+'$', re.VERBOSE | re.IGNORECASE)
punctuation = list(string.punctuation)
stop = punctuation + ['rt', 'via', 'amp', 'retweet', 'please retweet', 'retweet please']
#stopwords.words('english') +

def tokenize(s):
    return tokens_re.findall(s)

def preprocess(s, lowercase=False):
    tokens = tokenize(s)
    if lowercase:
        tokens = [token if emoticon_re.search(token) else token.lower() for token in tokens]
    preprocessed_tokens = []
    for tok in tokens:
        if not bool(re.search(r'([\.,;\\'"<>=/\@!])', tok)):
            if bool(re.search(r'#', tok)):
                preprocessed_tokens.append(tok[1:])
            else:
                preprocessed_tokens.append(tok)
    return preprocessed_tokens
```

Appendix 5-C

Data used for training the Stanford NER

motorists	traffic	update	
hit	how	m3	LOC
by	much	southbound	LOC
m56	longer	just	
delays	are	after	
following	the	m25	LOC
flooding	residents	gridlocked	
cheshire	of	i'm	
crash	gladstone	4	
involving	way	cars	
bmw	hawarden	back	
and	going	from	
mercedes	to	a	
blocks	be	totally	
busy	inconvenienced	smashed	
road	by	up	
in	your	astra	
cheltenham	dangerous	that's	
gloucestershire	roadworks	blocking	
	how	the	
minor	much	motorway	
delays	longer	what	
high	are	is	
st	the	causing	
flore	roadworks	the	
due	in	huge	
to	marford	congestion	
a	and	at	
broken	gresford	j19	LOC
down	going	a556	LOC
horsebox	to	45	
long	blight	am	
delays	the	from	
dunton	villages	mrs	
bassett	seems	white	
crossroads	never	sat	
temp	ending	there	
lights	accidentm	over	
in	3	40	
use	at	mins	
some	least	on	
already	an	the	
stuck	hour	m6	LOC
in	delay	nbound	LOC
m40	traffic	pls	

kingstanding	LOC	bar		bridge	
road	LOC	street	LOC	at	
kingstanding	LOC	in		aston	LOC
traffic		both		station	LOC
jam		directions		birmingham	LOC
before		at		thulston	LOC
the		b4035	LOC	roundabout	LOC
days		west		motorbike	
of		bar		crash	
heavy		street		on	
traffic		high		a6	LOC
as		street		leaves	
we		banbury		one	
know		cross		person	
it		who		derbyshire	
traffic		is		travel	
through		the		a40	LOC
pinhoe		silly		golden	
back		person		valley	
up		responsible		bypass	
to		for		inbound	
the		the		before	
m5	LOC	traffic		the	
motorway	LOC	light		elmbridge	LOC
bridge		timings		court	LOC
complete		a556	LOC	roundabout	LOC
stand		junction	LOC	in	
still		19	LOC	gloucester	
a		m6	LOC	is	
lot		it's		partially	
of		havoc		blocked	
traffic		each		due	
heading		morning		to	
out		changeit		an	
broadclyst		day		accident	
four		72		a534	LOC
vehicle		gravellyhill	LOC	heavy	
crash		chilly		traffic	
in		delays		standstill	
a623	LOC	someone's		at	
peak		hit		broxton	LOC
forest		the		roundabout	LOC
sparks		bridge		suspect	
derbyshire		again		accident	
police		ampd		one	
derbyshire		rush		of	
traffic		hour		them	
delays		delays		new	
on		as		buses	
a361	LOC	lorry		has	
south	LOC	strikes		broken	

down			bit		and	
at			of		oil	
crowles	LOC		a		spillage	
east	LOC		mare		on	
bound	LOC		any		a5	LOC
holding			chance		between	
up			stuck		shrewsbury	LOC
traffic			on		and	
accident			m1	LOC	telford	LOC
in			due		good	
kingstanding	LOC		to		morning	
on			accident		4	
a453	LOC		in		way	
college	LOC		tailback		traffic	
road	LOC		traffic		lights	
involving			completely		on	
12			stopped		willow	LOC
year			and		stcastle	LOC
old			backed		st	
boy			up		causing	
and			to		tail	
car			a5	LOC	backs	
has			preston	LOC	holes	
closed			island	LOC	fill	
road			shrewsbury	LOC	in	
at			before		just	
brackenbury	LOC		it		waiting	
road	LOC		turns		for	
another			into		a	
bloody			m54	LOC	scab	
traffic			will		of	
jam			be		tar	
m25	LOC		queueing		white	
motorway	LOC		back		bmw	
roadwork			to		in	
lights			emstrey		stenson	
outside			soon		canal	
new			road		near	
co			closed		derbu	
op			after		after	
in			serious		crash	
horsley	LOC		accident		with	
woodhouse	LOC		in		two	
stuck			clayton	LOC	people	
on			staffordshire	LOC	derbyshire	LOC
red			motorists		traffic	
both			face		obviously	
ways			delays		a41	LOC
de76ax			following		tushingham	LOC
causing			lorry		moving	
a			fire		again	

bad
accident
on
east
lancs
a580 LOC
liverpool LOC
bound
junction
at
catchdale LOC
moss LOC
avoid
huge
tailbacks
on
m25 LOC
after
rush

Appendix 6-A

Users' perception of the transport network (Full table 6-8)

Tweet	Issue
Once again we see the effect of a pedestrian crossing within 20 yards of A45 Kenilworth Road junction, accident waiting to happen!	Pedestrian crossing
Matrix sign A14 w/b at M6/M1 junction not advising of M1 n/b closure. Drivers could avoid if diverted on to M6 for M69	Matrix sign not updated
why don't somebody turn the lights off on the roundabout on the A5-A42 island and let the traffic flow freely ?????	Traffic lights timing
Usual awful traffic management between 23 and 25 of the M1 - nobody working!	Roadworks
M1 sth jcn 11-10 Matrix showing Ln 1 closed, all fully open. Satellite delay again??	Satellite delay
So the M6 has roadworks on it and 40 min delays. When do you find this out? When you get to it.	Roadworks
Traffic lights out on Binley Road at the cross junction, drive safely everyone, nearly got took out by a bus	Traffic lights not working
Who is responsible for the signaling on M1 J24 A453 island? This is a complete joke	Signaling
A453 at junction 24 of m1. Traffic lights out of sync. Miles of queues on a453 southbound. Can you sort please	Traffic lights
how about you find some government money and sort out this joke of a junction at j10 M42 and the A5	Junction delays
Stuck in standstill traffic with no explanation on the A1. Any ideas?	Unrecorded traffic event
#M6 J10 closed roadworks once again. No signage during the build up to tonight's work	No signage
Disappointed to see that the light on the sign at Pirehill is still flashing. Enough flashing lights on the A34 at the moment!	Traffic light not working
WTF is happening at Junction 9 M42? Moved 300 yards in 2 hours!	Unexplained traffic event
Traffic chaos on Chapel Lane in Sally Oak, Brum. The cause? A gaggle of geese crossing the roundabout.	Unrecorded traffic event
why standing traffic before junction 14 please, Nothing moving north	Unexplained traffic event

When are you going to sort out the awful traffic light system at the crossing over of Kenilworth road to the a45	Traffic lights not working
So why nothing about m6 southbound currently at standstill all 3 lanes ???	Unexplained traffic event
Please can you report pedestrian crossing Kenilworth Road, Balsall Common by Heart of England school, traffic light not working well	Traffic lights not working
M42 SB at a standstill 37over6 How come?	Unexplained traffic event
There is a broken down van on Five Ways roundabout currently causing traffic chaos!	Traffic incident
Anyone else sick of wolves road works? It's a fucking joke driving anywhere.	Roadworks
Please AVOID COMPTON this evening. Four way traffic lights at bottom of Holloway and traffic gridlock	Traffic lights not working
Matrix sign at j15 M1 says j13 20 mins, google and Waze saying M1 blocked at j14 southbound with 57min delay	Matrix sign not updated
Very Slow moving Traffic East-Bound on A46 from HobbyHorse Roundabout past A6	Unexplained traffic event
A kind person left their car on the J3 M42 overbridge after it broke down	Traffic event
Hi team. Trying to get from Birmingham to Worcester but there seems to be no diversion from M42 - M5. Can you help?	Matrix sign not updated
A45 at Irchester is at a standstill eastbound !	Traffic event
traffic standstill A46 towards M1 south, just before Kirby Muxloe junction	Traffic event
what is the thinking behind allowing 2 sets of traffic lgts on the main diversion route for the closed Farthinghoe road?	

Appendix 7-A

Automated methodology for locating traffic sensors in the model

```
def calculateDistanceSquared2DBetweenPoints(v, w):
    return (v.x - w.x) ** 2 + (v.y - w.y) ** 2

def getVectorBetweenPoints(p1, p2):
    v= GKPoint()
    v.x = p1.x - p2.x
    v.y = p1.y - p2.y
    v.z = p1.z - p2.z
    return v

def calculateClosestPointOnSegment(p1, p2, pTest):
    v = getVectorBetweenPoints(p2, p1)
    if math.fabs(v.x) + math.fabs(v.y)>0:
        u = ((pTest.x - p1.x) * v.x + (pTest.y - p1.y) * v.y) / (v.x * v.x + v.y * v.y)
        if u < 0:
            return p1
        elif u > 1:
            return p2
        else:
            n=GKPoint()
            n.x = round(p1.x + u * v.x)
            n.y = round(p1.y + u * v.y)
            return n
    return p1

def getClosestSectionPointFromPoint(p):
    sections = model.getCatalog().getObjectsByType( model.getType( "GKSection" ) )
    section = None
    distance_to_section = 50000
    closest_segment= None
    Closest_point= None
    dist= 50000
    if len(sections) > 0:
        for s in sections.itervalues():
            d = s.getPoints().distToPoint2D(p)
            if d < distance_to_section:
                distance_to_section = d
                section = s
        if section != None:
            points=len(section.getPoints())
            for i in range(0,(points-1)):
                segment = [section.getPoint(i), section.getPoint(i+1)]
                sectionPoint =
    calculateClosestPointOnSegment(section.getPoint(i), section.getPoint(i+1), p)
    dt = math.sqrt(calculateDistanceSquared2DBetweenPoints(p,
    sectionPoint))
```

```

        if dt < dist:
            dist = dt
            closest_segment = segment
            closest_point = sectionPoint
    return section, closest_point

```

```

query = {"loc": {"$within": {"$box": [[-1.9367252, 52.5504715], [-1.9274452,
52.5540443]]}}}

```

```

for doc in collection.find(query):

```

```

    latlong=doc['loc']['coordinates']
    lat=latlong[1]
    lon=latlong[0]
    coord= utm.from_latlon(lat, lon) #Converts to WGS84 (Aimsun coordinates)
    p=GKPoint()
    p.x=float(coord[0])
    p.y=float(coord[1])
    p.z=0
    name=str(doc['measurementId'])
    section, point= getClosestSectionPointFromPoint(p)
    if section != None:
        position = section.posAtPoint(point)
        nbLane = section.getNbLanesAtPos(position)
        detector = model.newObject('GKDetector')
        detector.setPosition(position)
        detector.setLanes(0, nbLane - 1)
        detector.setLength(1.8)
        detector.setName(name)
        detector.adjust()
        section.addTopObject(detector)
        layer = model.getGeoModel().getActiveLayer()
        model.getGeoModel().add(layer, detector)

```

Appendix 7-B

Automated methodology for importing traffic data into the model

```
def DistribError(a,b,c):
    if a == 0:
        a = c - b
        if a < 0:
            a = abs(a)
            error = c - (a+b)
            dist = abs(error/3)
            if error > 0:
                a = dist + a
                b = dist + b
                c = a + b
            if error < 0:
                a = a - dist
                b = b - dist
                c = a + b
    elif b == 0:
        b = c - a
        if b < 0:
            b = abs(b)
            error = c - (a+b)
            dist = abs(error/3)
            if error > 0:
                a = dist + a
                b = dist + b
                c = a + b
            if error < 0:
                a = a - dist
                b = b - dist
                c = a + b
    elif c == 0:
        c = a + b
        if c < 0:
            c = abs(c)
            error = c - (a+b)
            dist = abs(error/3)
            if error > 0:
                a = dist + a
                b = dist + b
                c = a + b
            if error < 0:
                a = a - dist
                b = b - dist
                c = a + b
    else:
        error = c - (a+b)
        dist = abs(error/3)
```

```

        if error > 0:
            a = dist + a
            b = dist + b
            c = a + b
        if error < 0:
            a = a - dist
            b = b - dist
            c = a + b
    return (a,b,c)

def TurnPercentage(a,b,c):
    a = float(a)
    b = float(b)
    c = float(c)
    perc1 = ((a/c)*100)
    perc2 = ((b/c)*100)
    return (perc1, perc2)

def getStateFolder( model ):
    folderName = "GKModel::trafficStates"
    folder = model.getCreateRootFolder().findFolder( folderName )
    if folder == None:
        folder = GKSystem.getSystem().createFolder( model.getCreateRootFolder(),
folderName )
    return folder

def findSection( model, entry ):
    section = model.getCatalog().find( int(entry) )
    if section.isA( "GKSection" ) == False:
        section = None
    return section

def importStateFlow( model, state, id, value):

    section = findSection( model, id )
    # Set the value if the section is valid
    if section != None:
        flow = float(value)
        state.setEntranceFlow( section, None, flow )

def importStateTurns( model, state, origin, destin, value):

    section = findSection( model, origin )

    toSection = findSection( model, destin )

    turnflow = float(value)

    if section != None and toSection != None:
        state.setTurningPercentage( section, toSection, None, turnflow )

```

```
def SectionCount(sectionid):  
    s =findSection(model, sectionid)  
    column= model.getColumn("DYNAMIC::SRC_GKSection_flow_0")  
    timeseries= s.getDataValueTS(column)  
    value = timeseries.getAggregatedValue()  
    #print sectionid  
    #print value  
    return value
```

Appendix 7-C

Optimisation-based imputation methodology

```
def MAPE(real,simulated):
```

```
    mape = abs ((real-simulated)/real) * 100
    #print mape
    return mape
```

```
def datetime_range(start, end, delta):
```

```
    current = start
    while current < end:
        yield current
        current += delta
```

```
def objective(x):
```

```
    return ((flowDib[16]-x[31])**2 + (flowDib[23]-x[32])**2 + (flowDib[8]-x[45])**2 +
(flowDib[4]-x[46])**2)
```

```
def constraint1(x):
```

```
    return x[0]-x[15]-x[16]
```

```
def constraint28(x):
```

```
    return x[6]-x[17]-x[18]
```

```
def constraint2(x):
```

```
    return x[31]-x[15]-x[17]
```

```
def constraint3(x):
```

```
    return x[7]-x[16]-x[18]
```

```
def constraint4(x):
```

```
    return x[7]-x[19]-x[20]
```

```
def constraint5(x):
```

```
    return flowDib[14]-x[21]-x[22]
```

```
def constraint6(x):
```

```
    return x[4]-x[19]-x[21]
```

```
def constraint7(x):
```

```
    return x[8]-x[20]-x[22]
```

```
def constraint8(x):
```

```
    return x[8]-x[9]-x[10]
```

```
def constraint9(x):
```

```
    return x[11]-x[9]-x[3]
```

```

def constraint10(x):
    return x[10]-x[23]-x[24]

def constraint11(x):
    return x[2]-x[25]-x[26]

def constraint12(x):
    return x[32]-x[23]-x[25]

def constraint13(x):
    return x[12]-x[24]-x[26]

def constraint14(x):
    return flowDib[19]-x[14]-x[13]

def constraint15(x):
    return x[5]-x[14]-x[12]

def constraint16(x):
    return x[13]-x[27]-x[28]

def constraint17(x):
    return x[11]-x[29]-x[30]

def constraint18(x):
    return x[1]-x[27]-x[29]

def constraint19(x):
    return x[6]-x[28]-x[30]

def constraint20(x):
    return x[33]+x[37]-x[38]

def constraint21(x):
    return x[38]-x[45]-x[39]

def constraint22(x):
    return x[40]-x[39]-flowDib[10]

def constraint23(x):
    return x[40]-x[35]-x[41]

def constraint24(x):
    return x[41]+x[36]-x[42]

def constraint25(x):
    return x[42]-x[46]-x[43]

def constraint26(x):
    return x[43]-x[44]+ flowDib[1]

```

```

def constraint27(x):
    return x[44]-x[37]-x[34]

def optimise(n):
    x0 = np.random.uniform(low=50, high=600, size=(n,))
    #print x0
    b = [50, None]

    con1 = {'type': 'eq', 'fun': constraint1}
    con2 = {'type': 'eq', 'fun': constraint2}
    con3 = {'type': 'eq', 'fun': constraint3}
    con4 = {'type': 'eq', 'fun': constraint4}
    con5 = {'type': 'eq', 'fun': constraint5}
    con6 = {'type': 'eq', 'fun': constraint6}
    con7 = {'type': 'eq', 'fun': constraint7}
    con8 = {'type': 'eq', 'fun': constraint8}
    con9 = {'type': 'eq', 'fun': constraint9}
    con10 = {'type': 'eq', 'fun': constraint10}
    con11 = {'type': 'eq', 'fun': constraint11}
    con12 = {'type': 'eq', 'fun': constraint12}
    con13 = {'type': 'eq', 'fun': constraint13}
    con14 = {'type': 'eq', 'fun': constraint14}
    con15 = {'type': 'eq', 'fun': constraint15}
    con16 = {'type': 'eq', 'fun': constraint16}
    con17 = {'type': 'eq', 'fun': constraint17}
    con18 = {'type': 'eq', 'fun': constraint18}
    con19 = {'type': 'eq', 'fun': constraint19}
    con20 = {'type': 'eq', 'fun': constraint20}
    con21 = {'type': 'eq', 'fun': constraint21}
    con22 = {'type': 'eq', 'fun': constraint22}
    con23 = {'type': 'eq', 'fun': constraint23}
    con24 = {'type': 'eq', 'fun': constraint24}
    con25 = {'type': 'eq', 'fun': constraint25}
    con26 = {'type': 'eq', 'fun': constraint26}
    con27 = {'type': 'eq', 'fun': constraint27}
    con28 = {'type': 'eq', 'fun': constraint28}

    cons =
    ([con1,con2,con3,con4,con5,con6,con7,con8,con9,con10,con11,con12,con13,con14,con15,co
n16,con17,con18,con19,con20,con21,con22,con23,con24,con25,con26,con27,con28])

    sol = minimise(objective,x0,method='SLSQP',\
                    bounds=bnds,constraints=cons)
    #print sol
    x = sol.x
    sollist2= x.tolist()
    #x = sol.x
    success = sol['success']
    print success
    pru = x[6]-x[17]-x[18]

```

```
pru2 = x[0]-x[15]-x[16]  
print pru, pru2  
return sollist2, success
```

Appendix 8-A

Results from real-time implementation of the imputation methodology (Full table 8-1)

Time interval			MAPE				Iteration	Time
			C ₂	C ₄	C ₅	C ₇		
06	06:15	06:30	2.86%	8.89%	8.57%	8.45%	9	109.88
	06:45	07:00	4.46%	7.59%	8.50%	8.26%	7	109.4
07	07:00	07:15	3.57%	3.53%	9.66%	0.84%	1	16.38
	07:15	07:30	9.63%	2.86%	6.28%	9.05%	2	39.30
	07:30	07:45	8.63%	9.09%	9.84%	4.49%	5	91.99
	07:45	08:00	9.30%	1.61%	6.13%	7.81%	1	18.75
08	08:00	08:15	5.86%	4.76%	3.65%	9.52%	9	165.6
	08:15	08:30	9.31%	8.25%	8.88%	9.64%	3	56.02
	08:30	08:45	5.71%	2.74%	1.26%	5.56%	42	744.8
	08:45	09:00	5.94%	1.23%	2.42%	1.27%	1	17.68
09	09:00	09:15	2.22%	1.85%	0.00%	1.82%	9	145.6
	09:15	09:30	1.09%	4.44%	8.72%	9.86%	2	32.84
	09:30	09:45	2.15%	1.28%	6.90%	5.69%	1	16.14
	09:45	09:00	0.61%	2.38%	6.25%	4.00%	14	215.8
10	10:00	10:15	3.39%	8.51%	7.02%	0.00%	11	146.9
	10:15	10:30	2.21%	6.25%	9.65%	9.20%	7	95.28
	10:30	10:45	5.46%	0.00%	7.74%	0.45%	3	42.06
	10:45	11:00	6.35%	2.22%	2.04%	6.52%	7	96.13
11	11:00	11:15	0.57%	7.89%	7.14%	5.69%	54	767.4
	11:15	11:30	0.00%	8.33%	8.54%	4.08%	6	82.28
	11:30	11:45	4.44%	7.50%	9.89%	3.53%	12	169.3
	11:45	12:00	2.56%	0.00%	4.72%	7.69%	6	82.15
12	12:15	12:30	2.50%	0.59%	6.07%	2.72%	36	494.1
	12:30	12:45	0.50%	2.50%	6.37%	9.26%	41	586.8
	12:45	13:00	1.46%	8.51%	3.60%	5.56%	25	352.1
13	13:00	13:15	0.00%	9.30%	4.44%	0.77%	177	2745
	13:15	13:30	0.00%	3.92%	5.93%	3.92%	2	31.69
	13:30	13:45	5.85%	7.50%	5.83%	6.19%	32	538.8
	13:45	14:00	1.80%	2.17%	9.52%	6.21%	1	17.38
	14:00	14:15	2.34%	2.44%	5.43%	8.18%	1	16.29
14	14:15	14:30	2.79%	2.94%	5.30%	8.76%	3	52.96
	14:30	14:45	2.03%	0.00%	9.77%	4.03%	12	205.1
	14:45	15:00	3.83%	7.50%	4.17%	0.83%	26	448.7
	10:00	10:15	0.45%	3.70%	8.40%	5.77%	3	53.03
15	10:15	10:30	9.38%	8.16%	4.13%	1.86%	19	355.2
	10:30	10:45	4.37%	0.98%	2.07%	6.67%	16	290.3
	10:45	11:00	1.05%	2.63%	1.98%	9.26%	37	764.5
	16:00	16:15	3.31%	7.69%	4.80%	1.43%	8	175.4
16	16:15	16:30	4.32%	4.49%	6.25%	9.45%	45	172
	16:30	16:45	5.16%	7.14%	6.88%	8.78%	2	45.5
17	17:15	17:30	8.74%	7.69%	5.69%	9.76%	9	163.30

	17:30	17:45	6.92%	1.74%	9.52%	8.75%	1	18.20
	17:45	18:00	8.53%	1.01%	3.94%	6.97%	13	235.53
18	18:00	18:15	1.75%	4.72%	2.48%	2.70%	13	210.4
	18:15	18:30	0.99%	6.49%	3.25%	7.21%	2	31.79
	18:30	18:45	6.90%	9.80%	9.63%	5.88%	16	267.1
	18:45	19:00	3.35%	1.64%	0.66%	2.38%	1	15.5
19	19:00	19:15	0.76%	8.11%	0.00%	5.56%	1	12.05
	19:15	19:30	5.08%	2.33%	1.74%	6.67%	2	22.57
	19:30	19:45	4.12%	3.23%	9.28%	4.55%	12	128.1
	19:45	20:00	4.30%	7.41%	7.50%	8.11%	7	68.80
20	20:00	20:15	10.00%	4.76%	9.21%	0.00%	17	173
	20:15	20:30	7.88%	5.56%	8.62%	8.89%	9	87.40
	20:45	21:00	7.86%	5.88%	6.82%	4.35%	65	596.1
21	21:15	21:30	8.75%	6.67%	8.11%	3.75%	81	797.7
	21:30	21:45	3.12%	4.00%	4.44%	8.33%	6	50.38
	21:45	22:00	8.33%	7.41%	8.00%	3.24%	4	33.81